MINISTRY OF EDUCATION AND SCIENCE OF UKRAINE

M. E. Zhukovsky National Aerospace University
"Kharkiv Aviation Institute"

Faculty of Software Engineering and Business

Department of Software Engineering

Explanatory note

to the diploma project

(type of qualification work)

bachelor's degree
(educational degree)

on the topic "Image preprocessing software for PROMPT determination based on the CLIP model"

Khai.603.641p.24V.121.206315 PZ

Completed by: 4th year student, group 641p
Field of knowledge 12 Information technology
(code and name)
Specialty 121 – Software Engineering
(code and name)
Educational program Software Engineering
(name)

Pisatsky D.V.
(surname and initials of the student(s))
Head: Vdovitchenko O.V.
(surname and initials)
Reviewer: Kantemir I. V. (last name and initials)

Kharkiv – 2024

Ministry of Education and Science of Ukraine
M. E. Zhukovsky National Aerospace University "Kharkiv Aviation Institute"

Faculty of Software Engineering and Business
(full name)

Department of Software Engineering
(full name)

First level of higher education (bachelor's degree)

Area of knowledge 12 – information technology
(code and name)

Specialty 121 – Software Engineering
(code and name)

Software Engineering Educational Program

APPROVE

Head of the Department
_____ _Igor TURKIN____
(signature) (initials and last name)
"___" _____ 202_ year

TASK
FOR QUALIFICATION WORK

_____To Denis Viktorovych Pisotsky_____
(surname, first name and patronymic)

1. Topic of the qualification work "Image pre-processing software for determining PROMPT based on the CLIP model"

Head of qualification work Vdovitchenko Oleksandr Valeriyovych, Candidate of Technical Sciences, docent
(surname, first name, patronymic, academic degree, academic title)

approved by the University order No. __570____ of "April 25" 2024

2. The deadline for the applicant to submit the qualification work is 05/29/2024.

3. Initial data for the work, customer requirements for the application.

4. Contents of the explanatory note (list of tasks to be solved) 1) analysis of requirements for image preprocessing software for determining PROMPT based on the CLIP model;

2) design of image preprocessing software for PROMPT determination based on the CLIP model;

3) testing image pre-processing software to determine PROMPT based on the CLIP model;

4) economic justification for the development of image preprocessing software for determining PROMPT based on the CLIP model.

5. List of graphic material: use case diagram, class diagram, package diagram, function tree, interaction flowcharts, business model diagrams, database diagrams, screen form mockups

6. Consultants of qualification work sections

Section	Consultant's last name, initials and position	Signature, date	
		task issued	task accepted
1	Vdovitchenko O.V., Associate Professor, Department 603		
2	Vdovitchenko O.V., Associate Professor, Department 603		
3	Vdovitchenko O.V., Associate Professor, Department 603		
4	Kupriyanova V.S., associate professor, department 601		

Norm control _____ Tetyana DEGTYAROVA «_12__»__06____2024
(signature) (initials and last name)

7. Date of issue of the task "2" October 2023

CALENDAR PLAN

No. salary	Name of the stages of qualification work	Deadline for completion of qualification work stages	Note
1	Obtaining and approving the topic of the qualification works	01.07.2023 – 12.10.2023 Completed	
2	Domain analysis 3 Software	03.10.2023 – 22.10.2023 Completed	
	requirements analysis image preprocessing to determine PROMPT based on the CLIP model	10/28/2023 – 12/20/2023 Completed	
4	Software design image preprocessing to determine PROMPT based on the CLIP model	01.01.2024 – 01.03.2024 Completed	
5	Testing image preprocessing software for PROMPT detection based on the CLIP model	08.03.2024 – 01.05.2024 Completed	
6	Economic justification of software development image preprocessing to determine PROMPT based on the CLIP model	08.05.2024 – 18.05.2024 Completed	
7	Preparation of an explanatory note 8 Pre-	05/20/2024 – 05/25/2024 Completed	
	defense of the qualification thesis 9 Defense	05/27/2024 - 06/7/2024 Completed	
	of the qualification thesis	05/27/2024 - 06/7/2024 Completed	

Getter Denis Pisotsky
(signature) (initials and last name)

Head of qualification work

_____ Olexander Vdovitchenko
signature) (initials and last name)

ABSTRACT

Explanatory note to the bachelor's thesis project: 167 pages,
60 figures, 52 tables, 69 references to literary sources.

To improve and accelerate the work of companies engaged in image processing for PROMPT determination based on the CLIP model, it is planned to create ImgProPlus software, which will be a web application for image preprocessing. This application is designed to automate the process of preparing images for PROMPT determination using the CLIP model. The application will accelerate the exchange of information about image processing between users and the system. This product will allow you to move from manual image processing to an automated process. The web application is a convenient way to preprocess images from any device that has access to the Internet. The goal of the diploma project is to automate, simplify, and reduce human resource costs for the process of preparing images for PROMPT determination based on the CLIP model.

In the process of implementing the bachelor's thesis project, an analysis of the requirements for image preprocessing software was conducted, process and data models were developed, algorithms for software implementation were developed, modules were developed, software testing was conducted, and software development was economically justified.

Software design was performed in the Visual Studio Code development environment, and software development was implemented in the Python programming language.

To test an online food sales and delivery store, plans for system (41), integration (14) and stand-alone (13) testing were developed, specifications for tests were developed, and 9 stand-alone tests, 6 integration tests and 31 system tests were performed.

At the stage of economic justification of software development
providing pre-processing of images for determining PROMPT based on the CLIP model, the development cost was determined, which amounted to 182,663.42 UAH, the selling price of all subscriptions including VAT, which amounted to 526,848 UAH, with the profit being 188.42% of the cost, which corresponds to 344,184.57 UAH.

Keywords: *CLIP, Interrogator, transformer, artificial intelligence, application, PROMPT, image.*

ABSTRACT

Explanatory note to the bachelor's diploma project: 167 pages,
60 figures, 52 tables, 69 references to literary sources.

In order to improve and speed up the work of image processing firms to determine PROMPT based on the CLIP model, it is planned to create the ImgProPlus software which will be a web application for image preprocessing.
This application is designed to automate the process of image preparation for PROMPT definition using the CLIP model. The application will speed up the exchange of image processing information between users and the system. This product will allow you to move from manual image processing to an automated process. The web application is a convenient way to pre-process images from any device that has access to the Internet. The goal of the diploma project is to automate, simplify, and reduce the cost of human resources for the process of image preparation for determining PROMPT based on the CLIP model.

In the process of completing the bachelor's diploma project, an analysis of requirements for image preprocessing software was carried out, process and data models were developed, algorithms for software implementation were developed, modules were developed, software testing was carried out, and economically justified software development.

Software design was performed in the Visual development environment Studio Code and software development was implemented in the Python programming language.

To test the online store for food sales and delivery, plans for system (41), integration (14) and autonomous (13) testing were developed, test specifications were developed, 9 autonomous tests, 6 integration tests and 31 system testing tests were also performed.

At the economic substantiation stage of the development of image preprocessing software for determining PROMPT based on the CLIP model, the development cost was determined, which amounted to UAH 182,663.42, the selling price of all subscriptions including VAT, which amounted to UAH 526,848 while the profit is 188.42% of the cost price, which corresponds to UAH
344,184.57 .

Keywords: *CLIP, Interrogator, transformer, artificial intelligence, application, PROMPT, image.*

CONTENT

LIST OF SYMBOLS, SYMBOLS, UNITS OF MEASUREMENT OF PHYSICAL QUANTITIES, ABBREVIATIONS AND TERMS

3D – Three-Dimensional;
API – Application Programming Interface;
CLIP – Contrastive Language-Image Pretraining;
CPU – Central Processing Unit;
ER – Entity-Relationship;
GPU – Graphical Processing Unit;
HTTPS – Hypertext Transfer Protocol Secure;
JSON – JavaScript Object Notation;
PROMPT – Programmed Retrieval Operations for Machine Processing and Translation;
SQL – Structured Query Language;
SSL – Secure Sockets Layer;
TLS – Transport Layer Security;
UI – User Interface;
DB – database;
OOP – object-oriented programming; OS – operating system; SW – software; DBMS – database management system; AI – Artificial Intelligence.

INTRODUCTION

In today's information society and computer science, technology is developing rapidly, providing new opportunities in many aspects of human life. One of the key areas is image processing and text recognition, which play an important role in areas such as computer vision, medicine, autonomous vehicles, multimedia, and the gaming industry. [1]

In this study, we consider the topic "Image preprocessing software for query definition based on the CLIP model" [2] and justify its relevance, significance, and degree of novelty.

Tasks related to text and image recognition are becoming increasingly important in the context of modern technologies. The relevance of this topic is due to a number of factors.

First, the use of the CLIP model is very promising in the context of image processing. The CLIP model allows us to understand the semantics of images using text descriptions and provides the possibility of a wide range of applications. It can be used to improve the accuracy of image search and indexing, as well as to automatically generate text descriptions for images, which is important in multimedia systems. [3]

Secondly, the gaming industry is one of the industries where text recognition and image processing play a key role. From realistic game worlds to virtual scenery, games require the development of technologies that provide high-quality visualization and interaction. The possibilities of creating realistic objects and scenarios in games are very significant. [4]

Thirdly, “Generative from text to 3D” [5] artificial intelligence models are shaping the future. This means that based on text descriptions
it will be possible to generate 3D models of objects, and this will become promising direction for solving problems in the gaming industry, making scenery
and many other areas. This approach can significantly save time and resources, which is important in a competitive world.

Image preprocessing software also has a number of important advantages that make it attractive to a wide range of users. Among the features of this software are the following:

Efficiency. Image processing and PROMPT definition do not require time-consuming manual processing or complex procedures. The software provides convenient and fast tools to perform these tasks, allowing users to use their time effectively.

Convenience: Using image preprocessing software allows you to work with images anywhere, anytime, as long as you have access to a computer or mobile device. This provides convenience and flexibility in working with images.

Automation: The software provides the ability to automate many routine image processing operations, which significantly saves users time and effort. [6]

Accuracy: Using the CLIP model to determine PROMPT ensures high accuracy of results. The software helps to avoid errors and ensures reliability in determining results.

This technology is used by both image processing professionals and ordinary users who need a fast and reliable tool for working with images.

1 ANALYSIS OF REQUIREMENTS FOR IMAGE PRE-PROCESSING SOFTWARE FOR DETERMINATION PROMPT BASED ON CLIP MODEL

1.1 Business requirements

1.1.1 General description of the software product and business requirements

"ImgProPlus" is a new web application that provides an innovative platform for image preprocessing and PROMPT detection based on the CLIP model. The goal of the development is to create a system that will allow users to effectively use advanced technologies for image analysis and processing.

The customer of the project is the company "Image Intelligence Solutions" with the aim of improving the quality of image processing and expanding the possibilities of using the CLIP model in various industries, such as photo processing, 3D modeling, etc.

Business requirements related to the software product "ImgProPlus", are given in table 1.1. (see table 1.1)

Table 1.1 – Business requirements for the application.

ID	Description of business requirements
BR-01	Providing efficient image preprocessing for users with different levels of expertise.
BR-02	The opportunity for Image Intelligence Solutions to generate revenue through the provision of advanced features and services available for a fee.

1.1.2 Glossary development

Application – an interactive software product that used via the user's computer to define PROMPT.

Pre-training: The process by which a model learns universal representations of images and text before solving a specific task. [7]

Vector Representation: A unique numerical representation of an object or concept that is used for further analysis and comparison.

Transformer: A neural network architecture that
uses a self-attention mechanism to efficiently process sequences.

Prompt Engineering: The process of finding the best textual prompts to achieve the desired output in the PROMPT algorithm.

Contrastive Pre-training: A technique where a model learns to distinguish between similar and dissimilar objects to obtain more informative representations.

Language-Visual Representations: Vector representations that combine textual and visual concepts to enhance the algorithm's ability to understand the relationships between them.

Adaptive Algorithms: Algorithms that can automatically change their behavior and parameters according to new data and conditions for optimal performance. [8]

Automated Image Assessment: Using artificial intelligence algorithms to automatically evaluate and analyze images using the CLIP model.

Self-Attention Mechanism: Part of the transformer architecture that allows a model to interact with different parts of the input signal at different levels of representation.

Natural Language Processing Technologies: Tools and techniques used to analyze and understand human language, [9] which improve interaction with textual information in the context of learning English using CLIP.

Artificial Intelligence: A branch of computer science that studies the creation of systems capable of performing tasks that typically require human intelligence, such as pattern recognition, language, and decision-making.

Image Processing: The process of applying various techniques and algorithms to optimize and enhance images before further analysis.

Preprocessing Module: Part of the software product responsible for performing image processing and preparation operations for further use in the CLIP model.

Task Representation: The process of converting input tasks or questions into a format that is understandable by the CLIP model for efficient PROMPT definition.

Vector Representation: Converting image information into vector format for further use in a model.

Training and Calibration: The process of using training data to optimize the parameters of a CLIP model and adapt it to a specific task.

Term: A conventional designation for a word or concept that can be analyzed and defined using the CLIP model.

Active User: A person who uses the software for image preprocessing and PROMPT definition.

1.1.3 Review and analysis of existing software analogues

ImageToPrompt (see Fig. 1.1) is an electronic platform for image processing and PROMPT definition based on the CLIP model, which is offered as a free service. The service is designed to provide users with the ability to create and optimize PROMPT responses for use in artificial intelligence.

Unlike other services, this one is completely free and works in a matter of seconds.

The problems with this service are that it only allows you to process images up to 4 MB, and one at a time. Another problem is that the site is slow, although the PROMPT search algorithm works quickly, the site itself takes a very long time to open and is slow. It is also worth noting that PROMPT is not always of high quality, which leads to problems in further use.

Unlike imagetoprompt.com, our application will allow you to use all the capabilities of your computer to create PROMPT for multiple images at the same time, also since our application is a local version, users will not experience the problem of waiting for PROMPT or that the site is down.

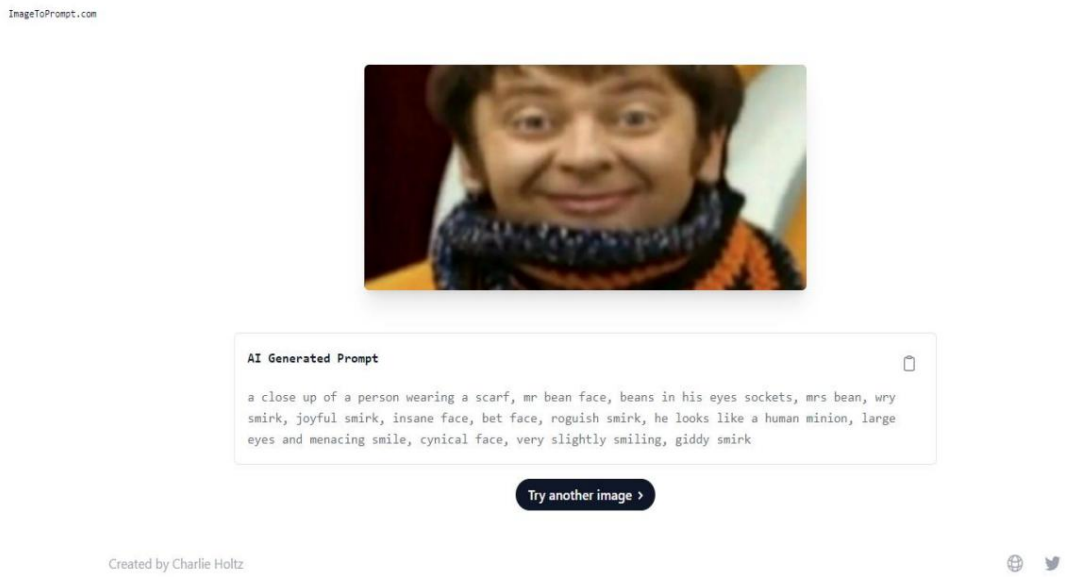

Figure 1.1 – ImageToPrompt website view

PromptoMania (see Fig. 1.2) is an innovative platform for image processing and PROMPT generation based on the CLIP model. Specialized in learning and optimizing PROMPT responses, PromptoMania provides unique tools for customizing and refining your prompts according to the chosen artificial intelligence model.

PromptoMania users can not only interact with the intuitive interface, but also create their own unique PROMPTs, using their own images and adapting them to their needs. The platform also allows you to select and customize PROMPTs according to the specifications of the selected model, providing greater flexibility in training and using artificial intelligence. [10]

PromptoMania offers two versions: free and paid. The free version includes basic features, but has limitations on processing large numbers of images and customizing PROMPT. The paid version, in turn, provides more capabilities, works more efficiently and faster, and also gives access to advanced features.

The downsides of "promptomania.com" are the limitations of the free version, which include a limit on the number of processed images and limited customization options for PROMPT. Also, free version users may be given limited access to some advanced features.

Unlike other analogues, PromptoMania provides a flexible toolkit for creating PROMPTs.

Our application will partially implement the customization function PROMPT and the ability to receive additional features for an additional fee.

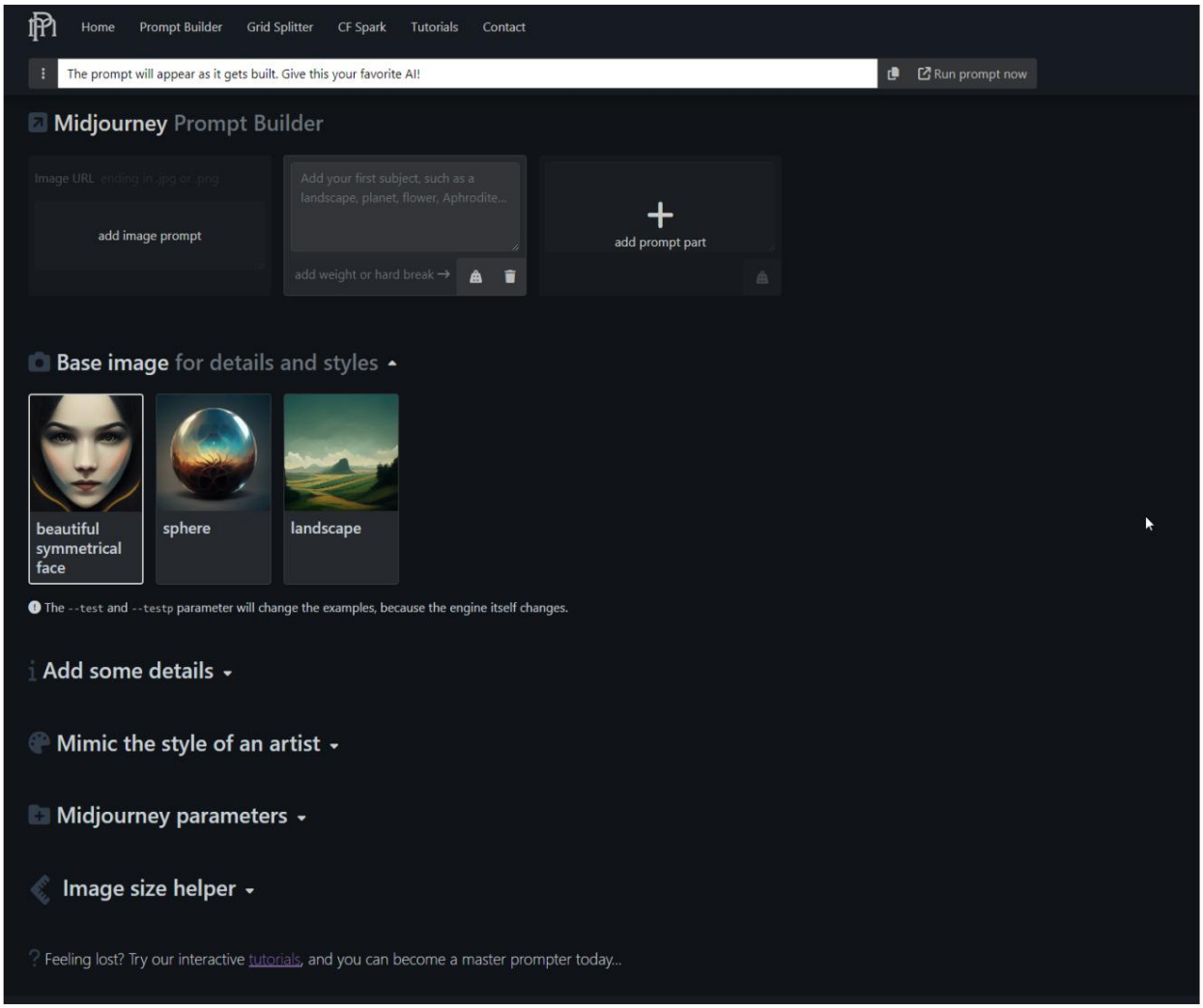

Figure 1.2 – Appearance of the PromptoMania website.

1.1.4 Software product concept and project scope.

ImgProPlus is a PC software for Image Intelligence Solutions, a company that aims to improve image processing and improve the quality of its services using CLIP technology. It automates image processing and management. Unlike existing programs that the company currently uses, ImgProPlus will be integrated with leading technology solutions and automates the entire image processing process.

In addition, it provides tools for analyzing and improving customer service, as well as a predictive and resource management system to maximize the use of processing power.

Functions that will be implemented:

– Ability to download and programmatically process images for creating improved PROMPTs.

– Visualization and analysis of results obtained from the CLIP model.

– Using your own images and the ability to adapt them to your needs user.

– Study and optimize work with PROMPT through the rules and documentation section.

Functions that will not be implemented:

– Customization according to the selected model.

1.1.5 Business rules

Business rules that apply to the ImgProPlus software are listed in the table. (see Table 1.2)

Table 1.2– Business Rules

Identifier Rule definition		Type regulations	Source (origin)
BRL-1	Using images and pre-processing them to create PROMPT	Fact	Methods definition PROMPT
BRL-2	Use image analysis and processing can when creating PROMPT.	Fact	Methods definition PROMPT
BRL-3	During the PROMPT definition process, it is necessary to use rules and algorithms based on the CLIP model.	Fact	Methods definition PROMPT

End of table 1.2

Identifier Rule definition Rule type			Source (origin)
BRL-4	When using the program, adapt the can processing parameters for achievement optimal results.	Fact	Methods definition PROMPT
BRL-5	Using dictionary and documentation for of definitions of terms and concepts in the context of the PROMPT definition	Fact	Methods definition PROMPT
BRL-6	Software control over over the efficiency of use PROMPT by inclusion mechanisms for evaluating and tracking results	Fact	Methods definition PROMPT

1.2 User requirements

1.2.1 Characteristics of user classes

General product characteristics:

The product is an image preprocessing software for PROMPT determination based on the CLIP model. The main functions include loading, image processing and

PROMPT definition using the CLIP model.

The product has an interactive user interface and integration with the CLIP model for optimal PROMPT definition.

Table 1.3 – User classes.

User classes Unauthorized	Description
user	Unauthorized user – a user who has just logged into the application and who must log in to obtain the appropriate role.
Subscription without user	A non-subscriber user is an application user who can use the application's features if they use the resources of their machine. In the case of requests to the server, the number of processing is limited.
Subscribed with user	A user with a subscription is an application user who has access to all new updates, algorithms, and has unlimited access to the server for image processing.
Administrator	An administrator is a person who has a special version of the program that allows them to interact with the database and, in accordance with user or developer requests, provide the necessary information (the developer is automatically an administrator).

1.2.2 Usage Class Diagram

According to the software, a usage diagram (see Fig. 1.3) was developed, which involved 4 people (unauthorized user, authorized user without subscription, authorized user with subscription, and administrator), 1 API, and 7 processes. More details about the usage algorithms will be described in the next section.

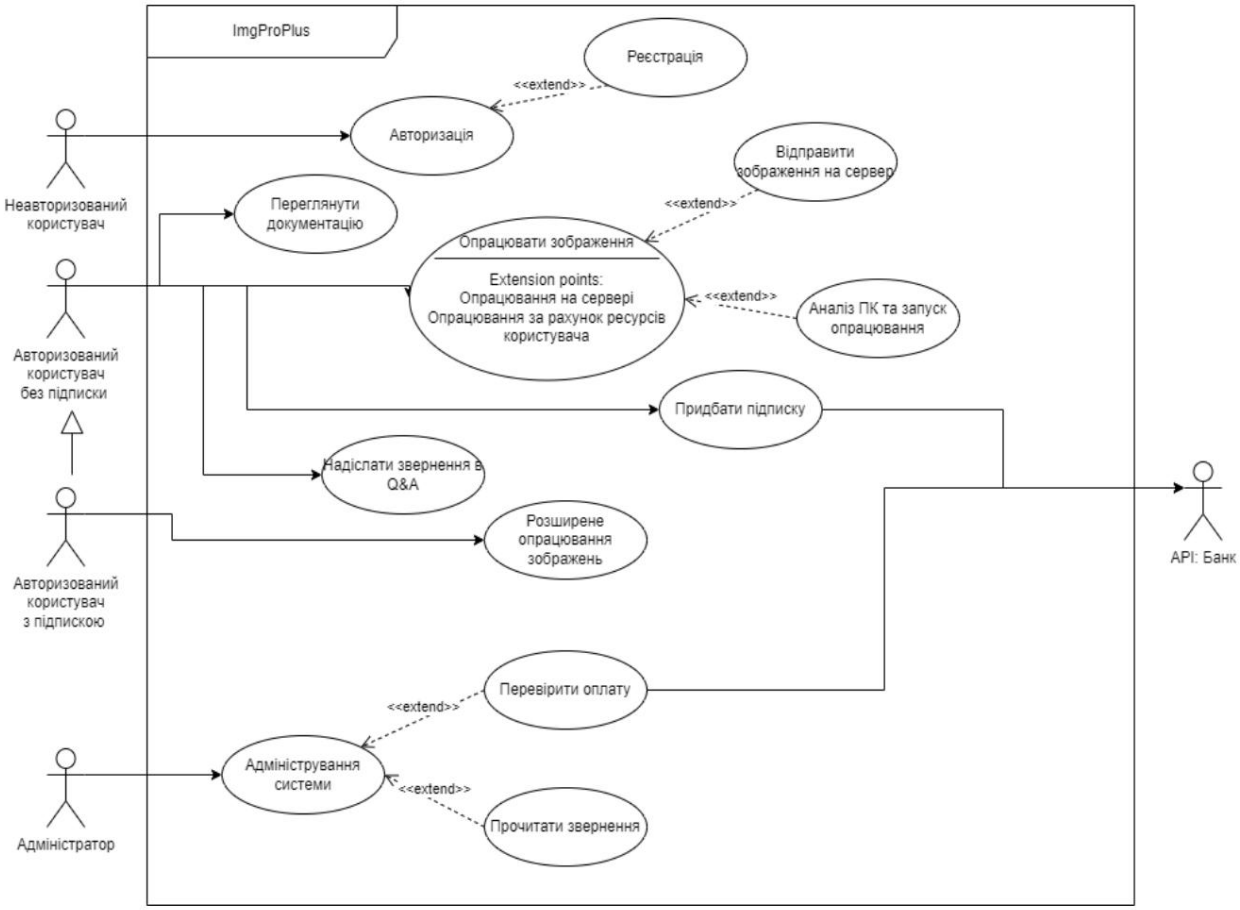

Figure 1.3 – Diagram of using image preprocessing software to determine PROMPT based on the CLIP model

1.2.3 Use case scenarios

Unauthorized user usage scenarios. (Authorization)

Table 1.4 – Usage option “Authorization”

ID and	UC-1 Authorization
name: Main actor Unauthorized user Additional actors –	
Trigger: Program activation Prerequisites:	
	–
Normal direction of use case development:	1.0 Log in 1. The user enters his/her login (email) and password. 2. If he/she is not registered, the user must register (see 1.1) 3. When you click login, the application sends a request to the server. 4. The response from the server has been received, the user has logged into the application. (see 1.0.E1)
Alternative directions for development of the use case:	1.1 Registration 1. User enters email and password 2. The user receives an email confirmation code 3. The user enters the code and successfully registers in additional.
Exceptions:	1.0.E1 Server is down 1a. After requesting authorization, the user receives a message stating that authorization is not possible and a request to try again later.
Initial conditions:	POST-1. The user has logged in.
Business Rules:	–
Non-functional requirements:	1. The functionality must be available 24/7. 2. The authorization functionality must be protected from unauthorized access to user data.

Table 1.5 – Option to use “Process Image”

ID and name:	UC-2 Process image
Main actor Authorized user with or without subscription	
Additional actors –	

End of table 1.5

ID and name:	UC-2 Process image
Trigger: User clicked upload image.	
Prerequisites: PRE-1. The uploaded image meets the requirements and the user has selected a processing method (local or server-side) (see 2.0.E1)	
Normal direction of use case development:	2.0 Process images 1. User uploads image to the application. 2. User selects server-side processing option (see 2.1) 3. The user selects the local processing option (see 2.2) 4. The user received the result in the form of prompts.
Alternative directions for development of the use case:	2.1 Process images on the server 1. The application sends an image to the server, which in turn sends a processed PROMPT. (see 2.1.E1) 2. The user receives the necessary PROMPT in the application. 2.2 Process images locally 1. The application processes the user's image and issues the necessary PROMPT.
Exceptions:	2.0.E1. The user is not subscribed and their free "credits" have expired 1a. The user receives a notification about the shortage and an offer to subscribe 2a. The image processing request can be transferred to a local version so that the user can process on their PC 2.1.E1. The server could not process the image 1a. The user receives a message that the image cannot be processed on the server. 2a. The user is provided with a "mini-report" with the error, and the ability to send it to the developers at the click of a button.
Initial conditions:	POST-1. The user processed the sent image and received a PROMPT from it. POST-2. In the case of processing on the server, a user who did not have a subscription loses 1 "credit".
Business rules: BRL-2, BRL-3, BRL-4	
Non-functional requirements:	1. Functionality must be available 24/7. 2. The processing functionality must be protected from unauthorized access to user data (from interception of images).

Table 1.6 – “Purchase Subscription” Usage Option

ID and name:	UC-3 Purchase subscription
Main actor Authorized user	
Additional API actors: Bank	
Trigger: Authorized user clicked the buy subscription button	
Prerequisites:	PRE-1. User has logged in to the application
Normal direction of use case development:	3.0 Purchase a subscription 1. The user receives information about available subscriptions and their cost. 2. The user selects a subscription and its term. The system calculates the cost. 3. The user enters bank details, or uses the built-in system (GooglePay, ApplePay). (see 3.0.E1) 4. The user receives subscriber status for a month (see 3.0.E2)
Alternative directions for developing the use case:	–
Exceptions:	3.0.E1 The user entered incorrect data. 1a. The user will receive additional notifications about incorrect data via the Bank's API 3.0.E2 The user does not have sufficient funds and receives a notification about this in additional messages.
Initial conditions:	POST-1. User processed received subscription POST-2. A user with a subscription has appeared in the database.
Business Rules:	BRL-5
Non-functional requirements:	1. The functionality must be available 24/7. 2. The purchase functionality must be protected from unauthorized access to user data (card number).

1.3 Software requirements

1.3.1 Functional requirements for software

I have constructed a corresponding function tree for “Image preprocessing software for determining PROMPT based on the CLIP model”. I have identified 4 corresponding functions: administrative, informative, image processing and payment. This division allows for a convenient and efficient distribution of software functions. [11]

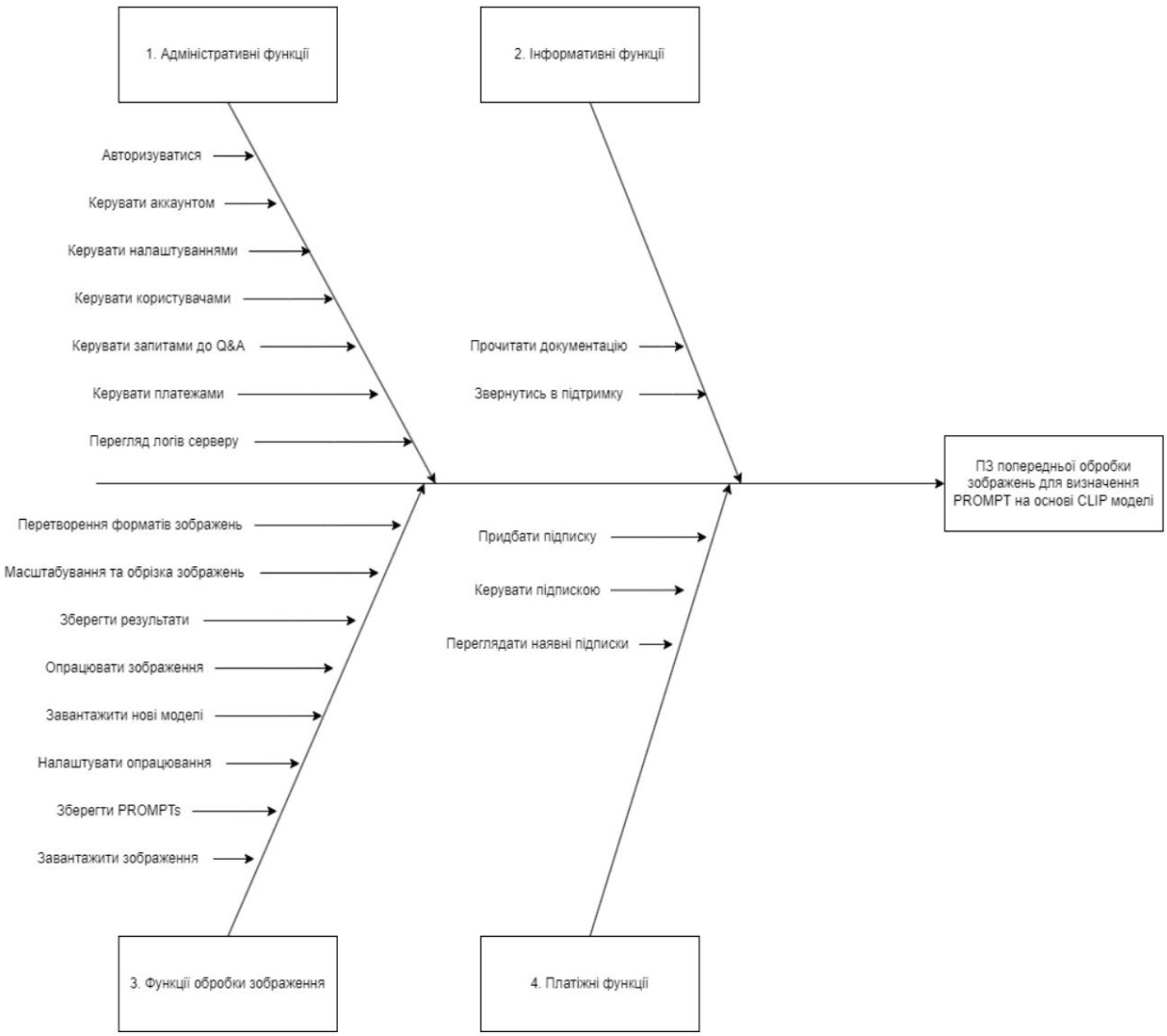

Figure 1.4 – Software function tree

1.3.1.1 Administrative functions

The "Administrative Functions" group of functions includes functions that deal with the management and configuration of various aspects of the system. These functions enable users, including administrators and regular users, to effectively manage various system parameters. (listed in Table 1.5) [12]

Table 1.5 – Functional requirements of group FR1

ID	Description of requirements
FR1.01	The system must be able to authorize users
FR1.01.01	When entering the application, the user has the option to authorize.
FR1.01.02	The system must allow the user to register using an email address and password. The user must receive a confirmation code.
FR1.01.03	The user should receive a message or receive a visual error (marked in red) when data is entered incorrectly.
FR1.02	The system should provide user account management.
FR1.03	The system should provide the ability for users to customize their account settings.
FR1.04	The system should allow administrators to manage users and their accounts
FR1.04.01	The administrator must have the ability to create a “test” user and view a list of existing users with the ability to edit and delete.
FR1.04.02	When deleting a user by an administrator, the system must display a confirmation window with two buttons: "Confirm deletion" and "Cancel"
FR1.05	The system should allow administrators to manage questions and answers.
FR1.06	The system should allow administrators to manage payments.
FR1.06.01	The payment management page must include the ability to viewing and editing the payment list.
FR1.06.02	The payment details page must include information about the payment itself and all related data.
FR1.07	The system should allow administrators to view server logs.

1.3.1.2 Informative functions

The "Informative functions" function group includes functions that are concerned with providing information and documentation on various aspects of the system. These functions enable the user to obtain information and understand the main aspects of the system. The details of the functions of this group are given in the table below (see Table 1.6):

Table 1.6 – Informative functions

ID	Description of requirements
FR2.01	The system should enable users to read the application documentation, which includes information about the functionality, settings, and instructions for using the application.
FR2.02	The system should allow users to contact support with access to resources for assistance.

1.3.1.3 Image processing

The "Image Processing Functions" function group includes functions related to image processing and transformation. These functions enable users to effectively control various image processing parameters. The details of the functions of this group are given in the table below (see Table 1.7):

Table 1.7 – Image processing functions

ID	Description of requirements
FR3.01	The system must be able to convert different image formats when loading.
FR3.02	The system shall allow users to save the results of image processing
FR3.03	The system must be able to process images using loaded processing models (e.g. CLIP v1, v2, etc.).
FR3.04	The system must include the ability to select the type of processing "On the server" or "On the local machine".
FR3.04.01	When selecting "On the server", if there are not enough credits and there is no subscription, the system should display a message about this.

End of table 1.7

ID	Description of requirements
FR3.04.02	When selecting "On local machine", the user will be prompted to perform a power check on their PC, and then the results will be displayed.
FR3.05	The system should provide the ability to customize image processing. Customization options such as "Number of Steps", "Negative PROMPT", "Creativity" etc.
FR3.06	The system should provide the ability to save image processing results according to specified PROMPTs (instructions). The system should allow
FR3.07	loading of new image processing models (e.g. CLIP v1, v2, etc.).
FR3.07.01	The system should provide the ability to select a model from available models.
FR3.07.02	The system shall, if the user does not have a subscription, display a message when attempting to download models, explaining that this feature is only available with a subscription and offering to purchase one. The system shall provide the
FR3.08	ability to download images for further processing.

1.3.1.4 Payment functions

The "Payment Functions" function group includes functions related to the management and configuration of various aspects of the payment system. These functions enable users to effectively manage various parameters of the payment system. The details of the functions of this group are given in the table below (see Table 1.8): [13]

Table 1.8 – Payment functions

ID	Description of requirements
FR4.01	The system should allow users to purchase a subscription from a list of possible subscriptions.
FR4.02	The system should allow users to view their existing subscriptions. The user should be presented with a list of available subscriptions from which they can select. The prices and discounts (if any) should also be displayed.
FR4.03	The system should allow users to manage their subscription.

1.3.2 Non-functional software requirements

1.3.2.1 External interface requirements

1.3.2.1.1 User interfaces

According to "CLIP-based image preprocessing software model", a map of dialog boxes was constructed (see Fig. 1.5).

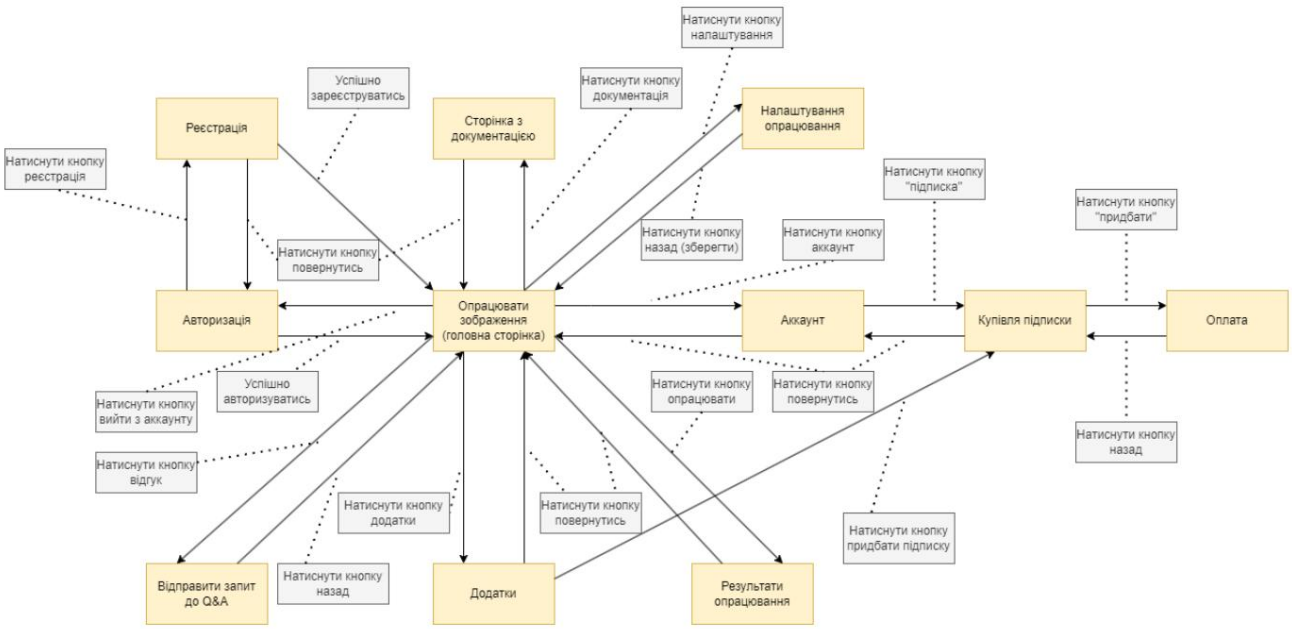

Figure 1.5 – Dialog box map

Disposable dialog box prototypes:

According to the constructed map of dialog boxes (see Fig. 1.5), a prototype was developed for each dialog box, which is presented below (see Fig. 1.6 – 1.16).

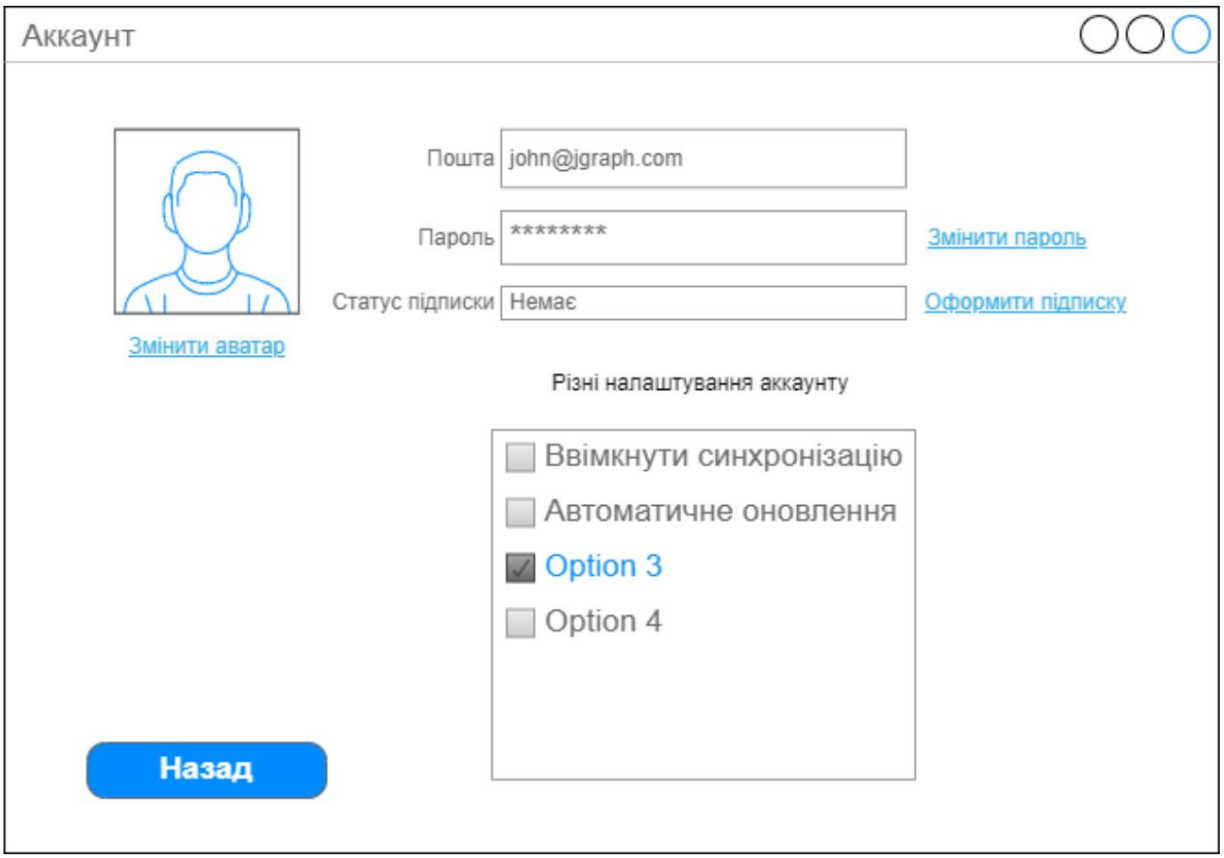

Figure 1.6 – Prototype of the Account dialog box

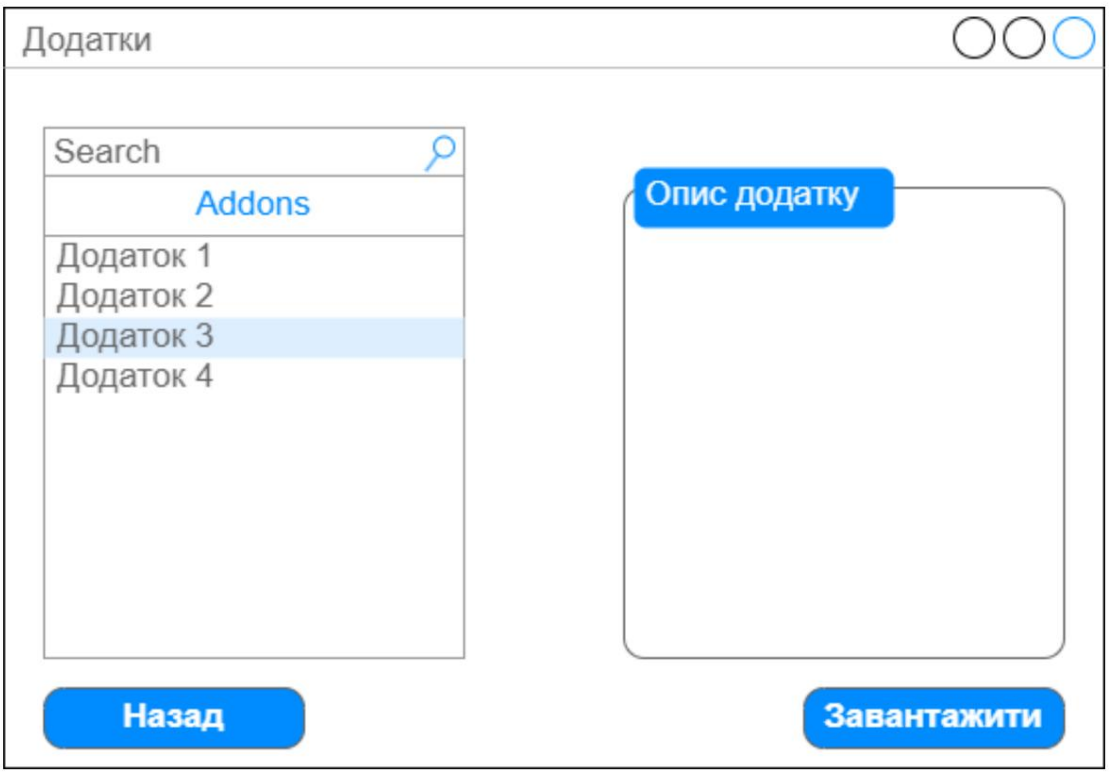

Figure 1.7 – Prototype of the “Add-ons” dialog box

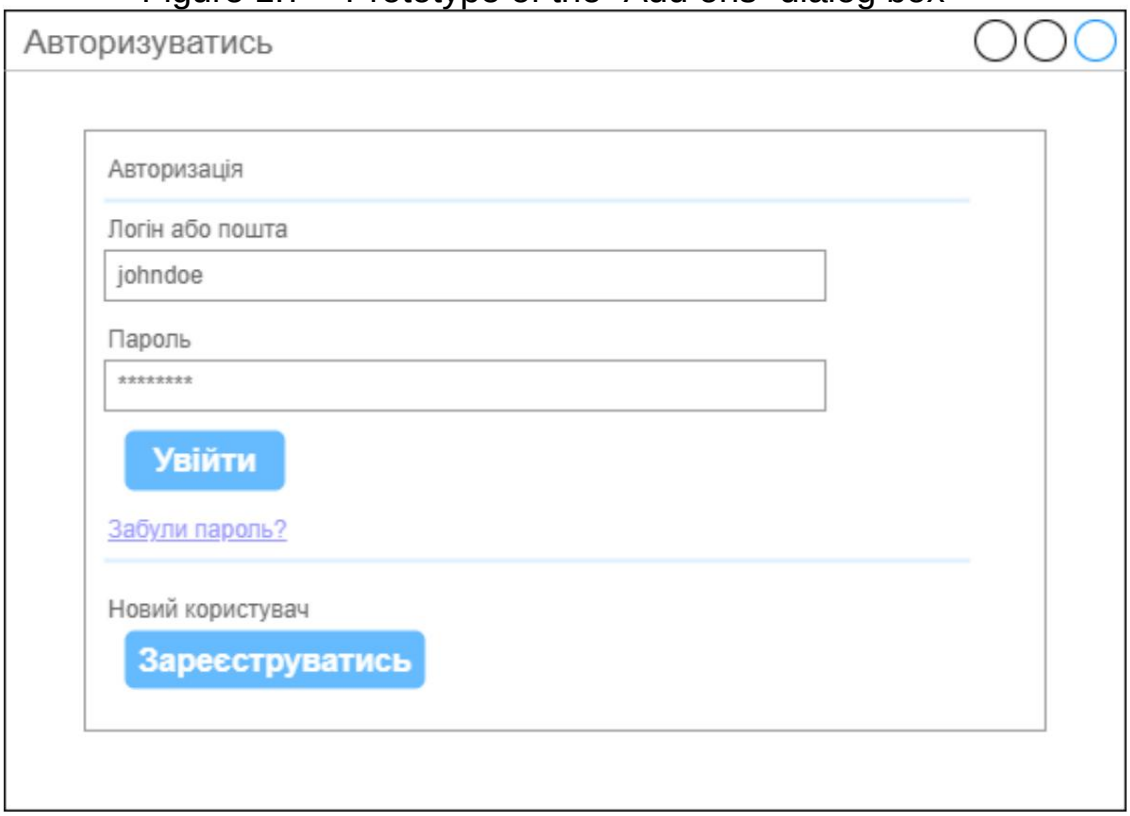

Figure 1.8 – Prototype of the “Authorization” dialog box

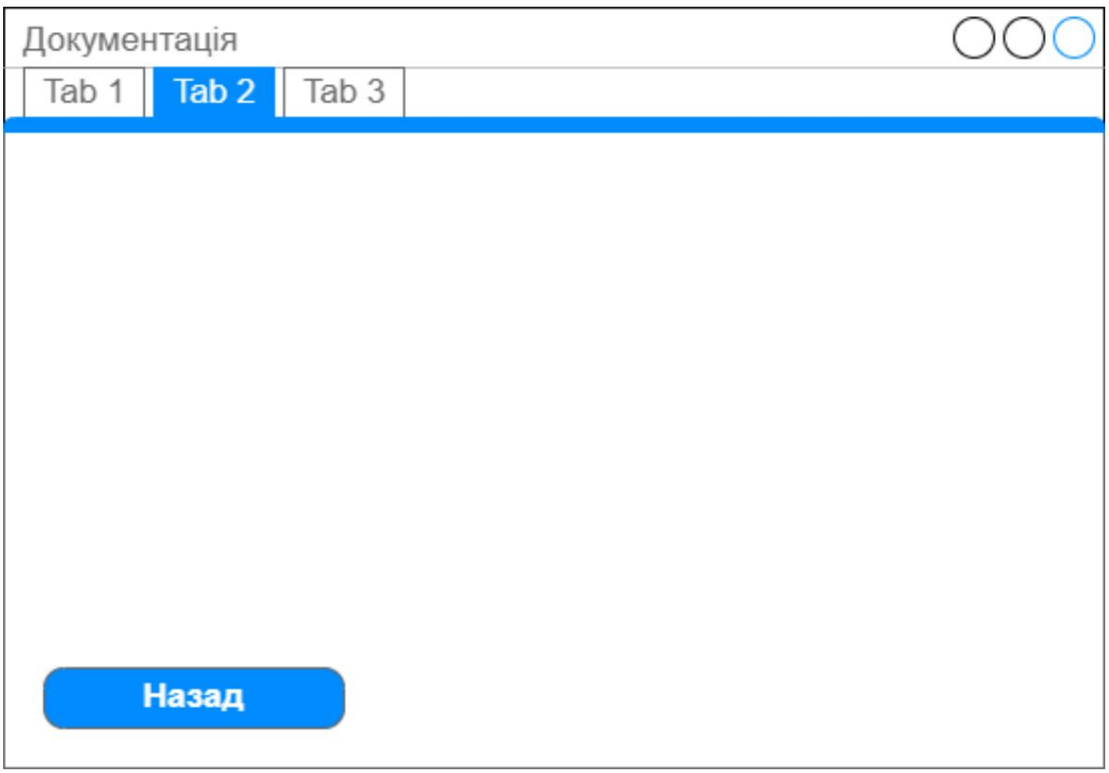

Figure 1.9 – Prototype of the “Documentation” dialog box

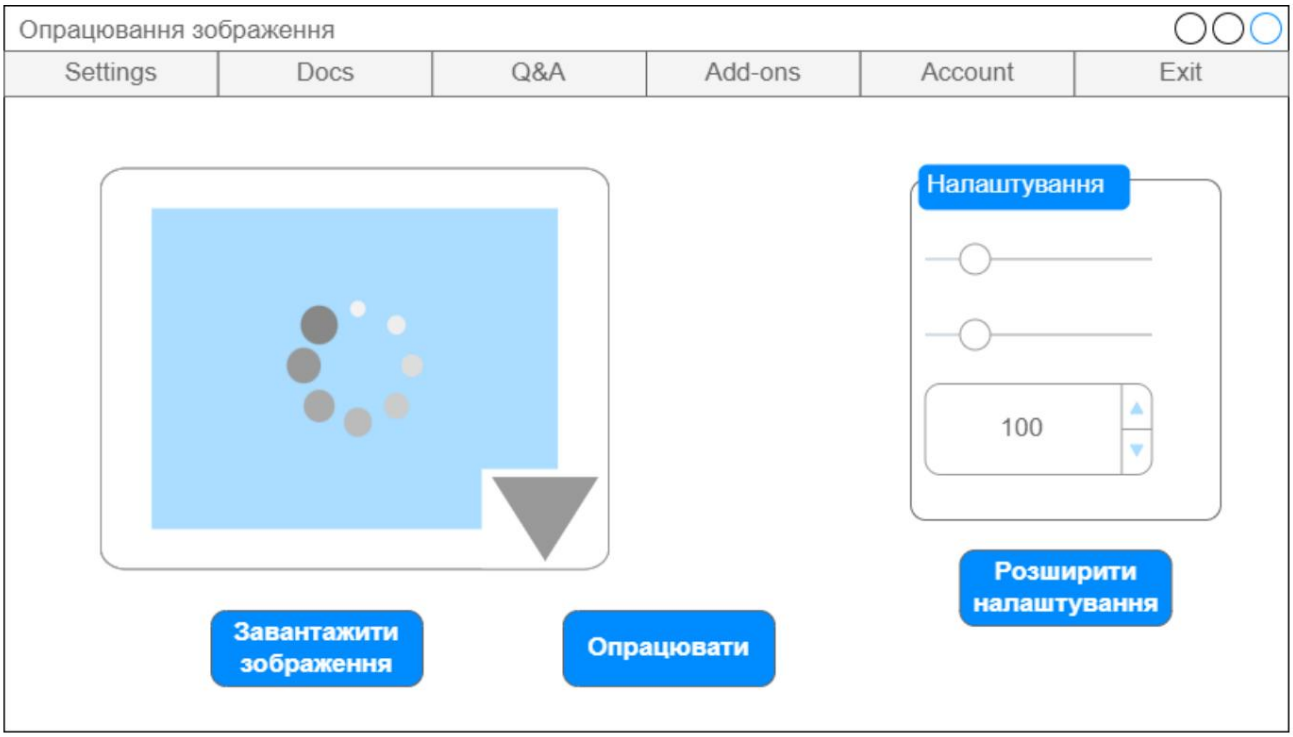

Figure 1.10 – Prototype of the “Image Processing” dialog box

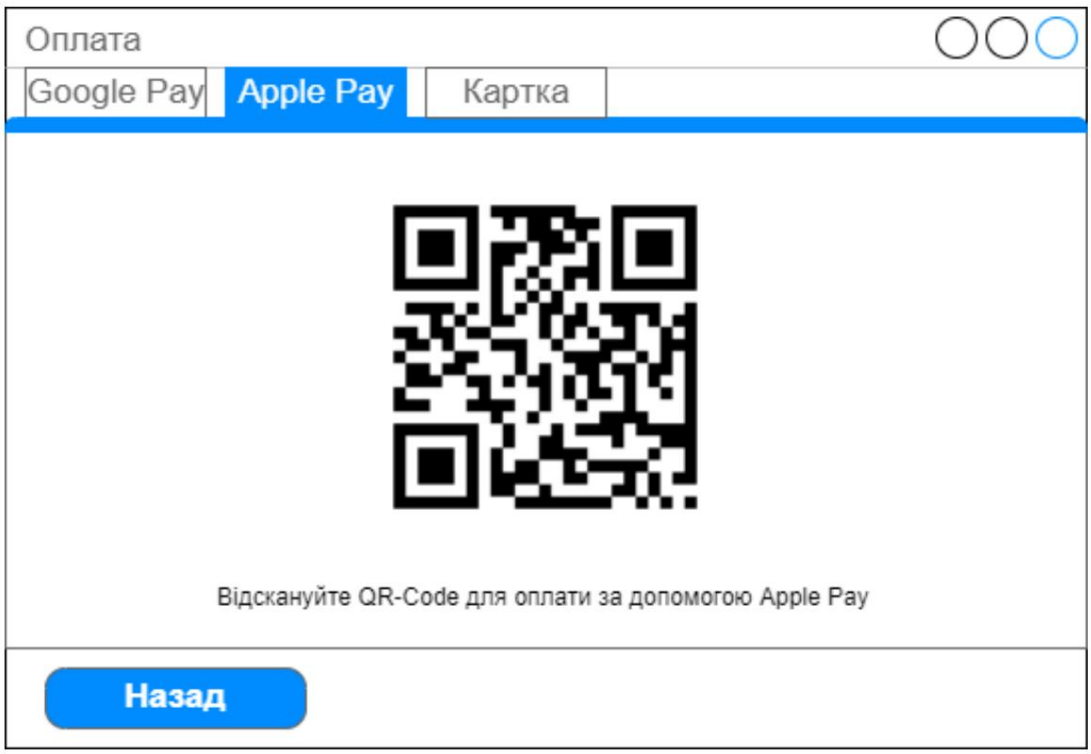

Figure 1.11 – Prototype of the “Payment” dialog box

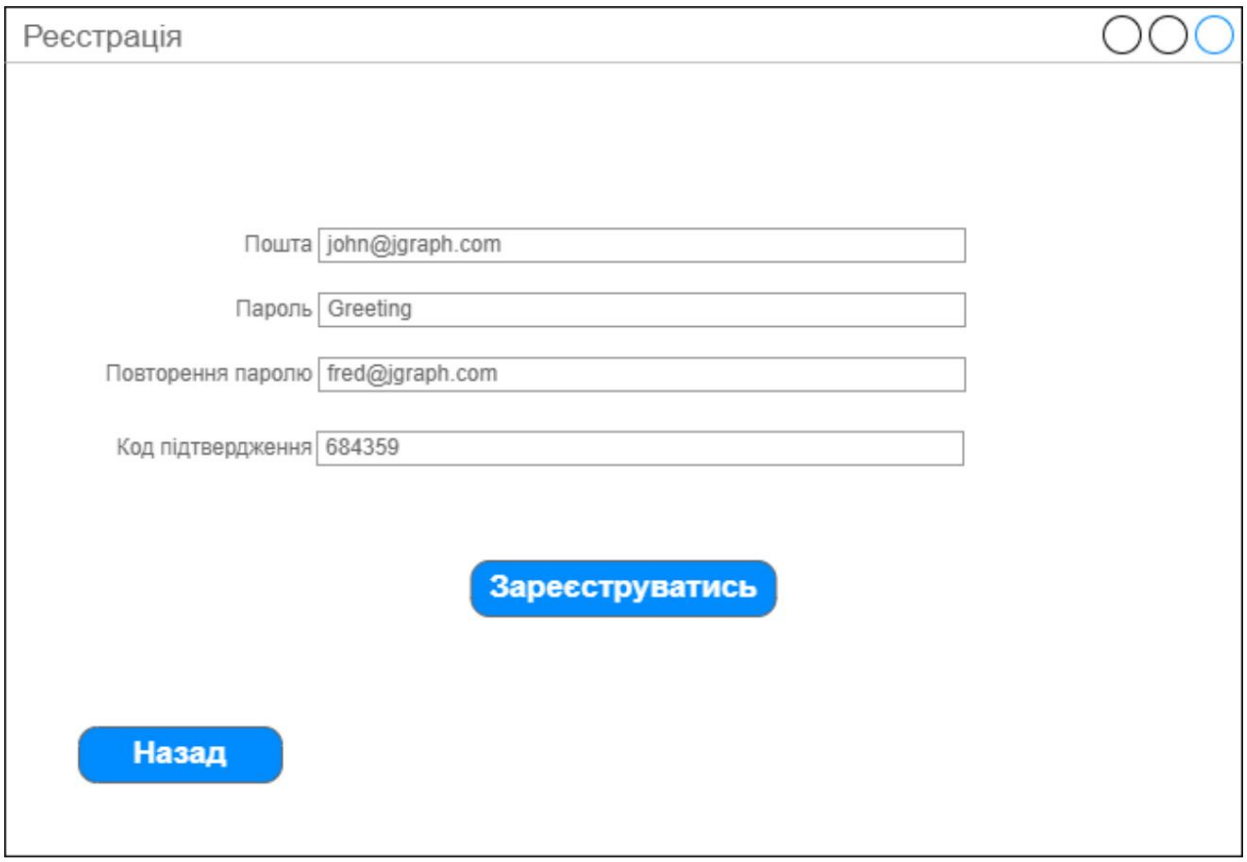

Figure 1.12 – Prototype of the “Registration” dialog box

Figure 1.13 – Prototype of the “Q&A Request” dialog box

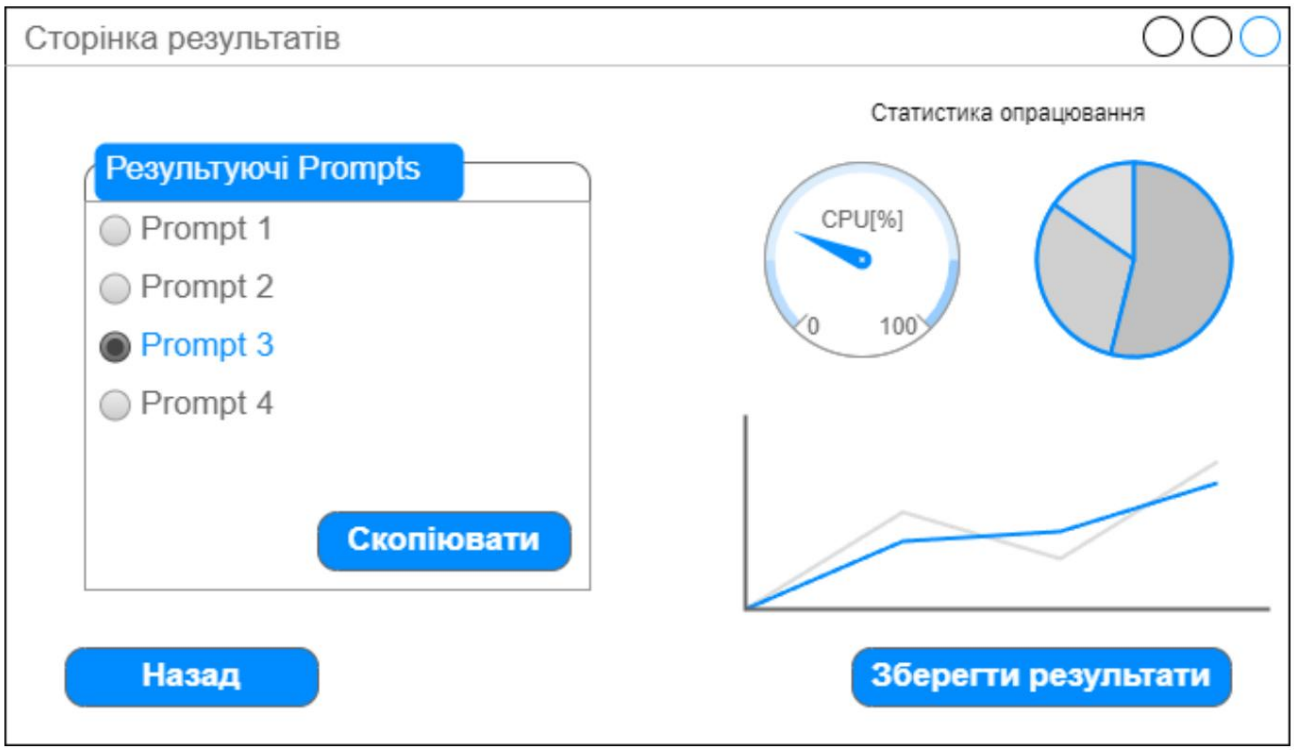

Figure 1.14 – Prototype of the Results Page dialog box

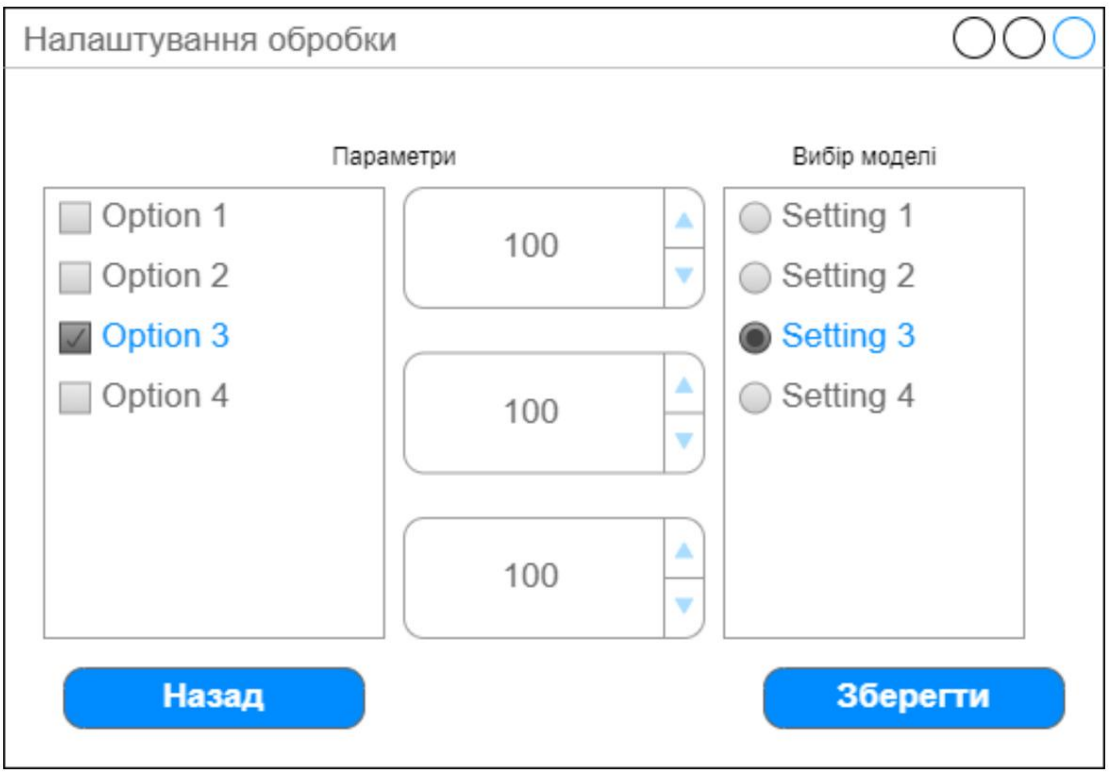

Figure 1.15 – Prototype of the Processing Settings dialog box

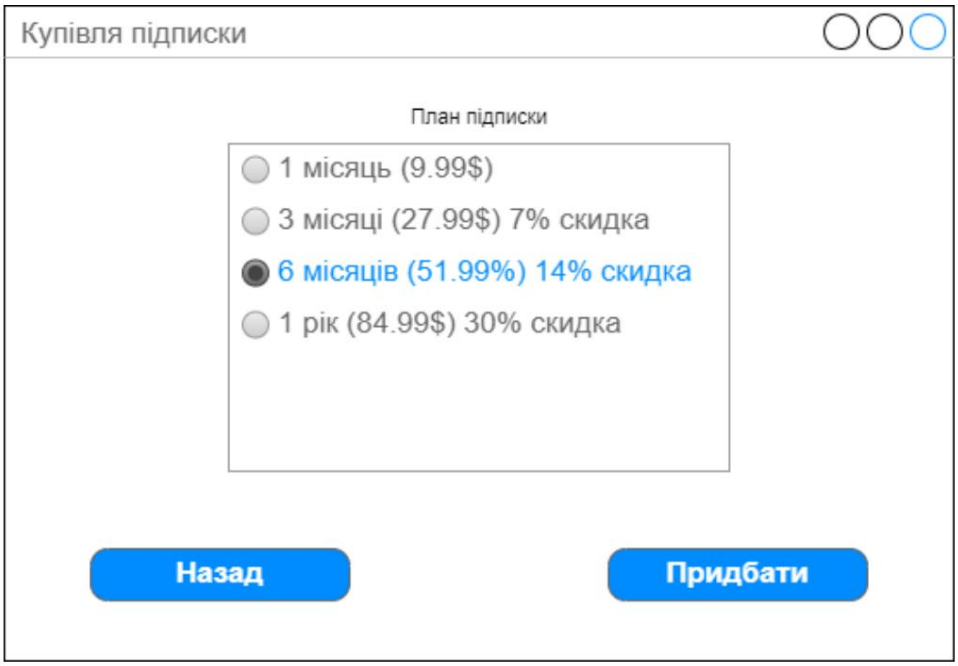

Figure 1.16 – Prototype of the “Purchase Subscription” dialog box

Along with the prototypes, user interface requirements were developed, which are listed in Table 1.9. (see Table 1.9)

Table 1.9 – Description of user interface requirements

ID	Description of requirements
UI-01	Each window related to image processing based on the CLIP model must have a name that clearly indicates its purpose and functionality.
UI-02	Each window has a close button in the upper right corner windows.
UI-03	The system must provide a link to help for each form explaining how to use that form.
UI-04	The system must provide a reference to help for each CLIP-based image processing function, explaining to users how to use the function.
UI-05	The system should allow users to easily access context-sensitive help, even when working with a specific image processing function based on the CLIP model.
UI-06	Each interface element that interacts with the CLIP model must display intuitive tooltips to help the user understand actions and options.
UI-07	The system should allow users to customize the visualization parameters of image processing results based on the CLIP model for better understanding of the output data.
UI-08	The "Authorization" window should contain the "Login" and "Password" fields, the "Login" and "Register" buttons, and a link to recover your password.
UI-09	The "Registration" field should contain the fields "Email", "Login", "Password", "Repeat password", "Confirmation code", "Back" and "Register" buttons.
UI-10	The "Processing Settings" window should contain a corresponding "Parameters" block for user editing, and a "Model Selection" block. The window should also contain "Back" and "Save" buttons.
UI-11	The "Purchase Subscription" window should contain "Back" and "Purchase" buttons. It should also contain a corresponding "Subscription Plan" block, which will display a list of available subscriptions in a user-friendly format.
UI-12	The "Results Page" window should contain a list of "Result PROMPTs", a "Processing Statistics" block, which will display the processing results in a visual format. It should also contain the "Back" and "Save Results" buttons.
UI-13	The "Q&A Request" window should contain a "Message" field, "Send" and "Back" buttons.

End of table 1.9

ID	Description of requirements
UI-14	The "Payment" window should contain the "Payment Methods" menu and display the available information (QR-Codes, etc.) according to the payment method, as well as the "Back" button.
UI-15	The "Image Processing" window should contain the "Image" field, the "Settings" block, the "Extend Settings" buttons, "Load Image", "Process" buttons, and a row of "Navigation" menus at the top of the window. When loading an image, the user will see a (transparent) button on the image to edit it (zoom and crop).
UI-16	The "Documentation" window should contain all the necessary information for working with the application, and links to relevant topics, articles, or documents on the Internet.
UI-17	The "Account" window should contain the fields "Mail", "Login", "Subscription availability", and buttons for navigating "Purchase subscription", "Back", "Change password".
UI-18	The "Apps" window should contain a block with available new models and algorithms, a "Search" field, a "App Information" block, and "Download" and "Back" buttons.

1.3.2.1.2 Software interfaces

Requirements according to external software are given in table 1.10. (see table 1.10)

Table 1.10 – Description of requirements for software interfaces

ID	External software	Description of requirements
SI 1-01	API Bank	The system must send a request using the bank's API and check the user's ability to pay for the subscription, and accordingly, upon successful payment, the system issues the user with subscriber status.
SI 1-02	API Bank	After receiving a successful payment, the system must send all payment data to the system so that it is stored in the system.

1.3.2.1.3 Communication interfaces

The software will also use certain communication interfaces to resolve issues with the user and verify their account. All requirements are listed in Table 1.11. (see Table 1.11)

Table 1.11 – Description of requirements for software communication interfaces

ID	Description of requirements
CI-01	The system should send the user a confirmation code to the specified email address upon registration, which will be valid for several hours. [14]
CI-02	After successfully purchasing a subscription, the system should send the user a check and a text describing all subscription options to the email address provided during registration.
CI-03	The system should send the user a link to change the password to the email address specified during registration.
CI-04	If a user has sent a request to Q&A, all responses from there should be sent to the user's email address specified during registration.

1.3.2.2 Requirements for quality indicators

Requirements for quality indicators can be divided into external, which are detected during execution, and internal, which are not detected. After analyzing the software, non-functional requirements were constructed and formulated. (see Table 1.12)

Table 1.12 – Description of non-functional requirements

ID	Attribute	Description of requirements
NFR-01	Availability	The system must be available at all times.
NFR-02	Scalability	The system must be scalable enough to support the upload of 10,000 images simultaneously.

End of table 1.12

ID Attribute	Description of requirements
Scalability - importance	The system must be scalable so that NFR-03 support simultaneous processing of 1000 images.
NFR- Portability	The program should work on different operating systems 04 systems, ensuring stable operation on Windows, macOS, and Linux platforms. [15]
NFR-05 Reliability	The system must operate without failures in 97% of usage cases during a month.
NFR-06 Reliability	The mean time to recovery (MTTRS) after a failure should not exceed 10 minutes. [16]
NFR-07 Security	The system must be protected from unauthorized access.
NFR-08 Productivity	The system should provide fast and predictable image processing based on the CLIP model, with a response time of no more than 5 seconds for standard images (Full-HD).
NFR-09 Ease of use	The level of user errors when interacting with the interface of an image processing system based on the CLIP model should not exceed 5%.
NFR-10 Ease of use	The system should have a simple and intuitive user interface that allows users to easily install and use the program without additional instructions.
NFR-11 Integrity	The system must guarantee the integrity of image processing, preventing data loss or exposure during processing and storage of results.
NFR-12 Verifiability and testability	Developers and testers must be able to effectively verify and test the image processing system, providing rapid confirmation of correct implementation and detection errors.
NFR-13 Protection	The system must have measures to protect against data and software corruption due to operation and interaction with other systems. [17]
NFR-14 Localization	The system must be able to satisfy the linguistic, cultural and other local requirements of users in different regions (Ukrainian, English). [18]

1.3.2.3 Limitations

Since the software has a specific idea and implementation, constraints have been developed that characterize the required tools and technologies. The corresponding constraints are given in Table 1.13. (see Table 1.13)

Table 1.13 – Software implementation limitations

ID	Description of requirements
LI-01	The system must use the Python or Rust programming language [19] to implement the client side of the application and image processing using the CLIP system.
LI-02	To work with the database, you must use MySQL.
LI-03	The implementation of the server side of the application can be performed using the Python or Node.js programming language, taking into account optimality and efficiency for specific tasks.
LI-04	To implement an image processing system using CLIP, it is recommended to use the OpenCV library [20] for working with images and PyTorch for integration with the CLIP model.
LI-05	Initial development of the application involves optimization for Windows 10 and 11 operating systems, and after successful implementation, porting to Linux and macOS platforms will be carried out.
LI-06	To interact with the banking API, use the monobank API or LiqPay to ensure efficient and secure exchange of financial information.
LI-07	To interact with the banking API (monobank or LiqPay), it is necessary to adhere to security protocols and standards, in particular, use the HTTPS protocol for secure data transmission.
LI-08	To ensure system security and stability, it is recommended to regularly update and use current versions of the libraries and frameworks used.
LI-09	The implementation of image processing must take into account the capabilities of the hardware platform and ensure optimal use of resources, in particular, the use of optimized algorithms for working with graphic data.

End of table 1.13

ID	Description of requirements
LI-10	When developing, it is necessary to take into account the possibility of integration with other systems in the future, so the code should be written taking into account open source software standards and RESTful API.
LI-11	Ensure modularity and extensibility of the design so that new functionality and components can be added without the need for significant changes to existing code.
LI-12	Use standardized approaches to code design and commenting to facilitate software understanding and maintenance.

1.4 Requirements Tracking Matrix

In accordance with the formulated rules, a requirements traceability matrix was constructed to determine the relationships between use cases (user requirements) and software requirements (see Table 1.14)

Table 1.14 – Traceability Matrix (Requirements Tracing)

Function linen requirements	Business Rules Usage Options												
	U C-01	U C-02	U C-03	U C-04	U C-05	U C-06	U C-07	BR L-01	BR L-02	BR L-03	BR L-04	BR L-05	BR L-06
FR1.01	+												
FR1.01.01 +													
FR1.01.02 +													
FR1.01.03 +													
FR1.02							+						
FR1.03							+						
FR1.04							+						
FR1.04.01							+						
FR1.04.02							+						
FR1.05							+						
FR1.06							+				+		
FR1.06.01							+				+		
FR1.06.02							+				+		
FR1.07							+						
FR2.01					+							+	
FR2.02		+						+					
FR3.01		+											
FR3.02		+											
FR3.03		+											
FR3.04		+				+				+			
FR3.04.01		+			+				+				
FR3.04.02					+ +								
FR3.05		+											
FR3.06		+											
FR3.07						+							
FR3.07.01		+											

End of table 1.14

Functions on-all requirements	Usage options							Business rules					
	U C-01	U C-02	U C-03	U C-04	U C-05	U C-06	U C-07	BR L-01	BR L-02	BR L-03	BR L-04	BR L-05	BR L-06
FR3.07.02						+							
FR3.08		+											
FR3.08.01		+											
FR3.08.02		+											
FR3.09		+											
FR4.01			+						+				
FR4.02			+						+				
FR4.03			+						+				
NFR-01		+			+							+	
NFR-02		+											
NFR-03									+				
NFR-04												+	
NFR-05										+			
NFR-06											+		
NFR-07 +	+				+								
NFR-08										+			
NFR-09		+											
NFR-10 +					+								
NFR-11		+				+							
NFR-12										+			
NFR-13		+ +											
NFR-14 +	+ + +	+ + +	+										+

Conclusions to Chapter 1

This section developed business requirements, user requirements, software requirements, non-functional software requirements, and a requirements tracking matrix for the project "PROMPT preprocessing software based on the CLIP model."

2 SOFTWARE DESIGN PRE-PROCESSING OF IMAGES FOR DETERMINATION PROMPT BASED ON CLIP MODEL

2.1 Architectural design

2.1.1. Software development methodology

We chose the Feature-Driven Development (FDD) methodology to manage the development process. This method allows us to focus on creating specific functional components. Feature-Driven Development (FDD) Methodology

An agile software development methodology, Feature-Driven Development (FDD), focuses on the continuous creation and improvement of individual functional components of a system. Developing a general model, building a feature list, feature planning, feature design, and implementation are the main stages of FDD. [21]

Development of a general model

In the first stage, we created the architecture of the system as a whole. The image preprocessing module, the text hint recognition module based on the CLIP model, and the user interface for interacting with the system are the main components. The overall model provides a basis for further detailing and development, and also helps to understand how the different components of the system interact with each other.

Building a list of functions

The list of system functions includes:

–Upload and store images
–Image pre-processing, including resizing, normalization, and feature extraction

–Integration with CLIP model for image analysis
–Generation of text prompts based on analysis results
–Displaying results to the user through a user-friendly interface

Each feature was detailed and included in the overall development plan.

Planning by function

Prioritization and sequencing are part of the planning for the implementation of the functions. Loading and storing images were the most important tasks for the further operation of the system. Development of the preprocessing module, integration of the CLIP model, and the function of generating and displaying text prompts were the next steps.

Design by function

Before implementing each feature, we developed a detailed technical specification. For the image preprocessing feature, we defined normalization and resizing algorithms, as well as parameters required for further analysis by the CLIP model. We also considered the possibility of scaling and adapting to different image types.

Implementation by functions

We developed a detailed technical specification before implementing each function. We defined the normalization and resizing algorithms for the image preprocessing function, as well as the parameters required for further analysis of the CLIP model. We also considered the possibilities of scaling and adaptation to different image types.

CLIP model integration

Programming for uploading and storing photos was the first step towards implementing the features. This was followed by pre-processing features, CLIP model integration, and text tooltip generation and display. Each feature was tested separately and together after it was integrated into the overall system.

Testing and validation

Software development depends on testing. We conduct testing with real users to evaluate the efficiency and usability of the system, integration tests to verify the interaction between different components of the system. We also planned for continuous improvement and optimization of the system based on user feedback and testing results.

2.1.2. General software architecture model/style

Client-server application software that uses the CLIP model for image processing can be described as a distributed system consisting of many major subsystems. The following describes the major subsystems and their relationships. [22]

Client Subsystem: The client subsystem resides in the CLIP model-based image processing software architecture and consists of many components that interact with the user and ensure that data is sent quickly to the server for processing.[23]
Let's talk about the details of this subsystem:

User interface (UI).

User Interface (UI): A graphical interface is a component of the client subsystem that allows users to interact with the system. [24] The user interface (UI) should be user-friendly and easy to understand so that users can easily use the functionality of the application.

Image upload form: You need to have elements for the user to upload an image. This could be a "Choose File" button or another UI element similar to this one.

Image Transfer Module: The data transfer mechanism is responsible for the efficient transfer of images from the user to the server. Distribution for efficient network utilization, data transfer optimization, etc. may be part of this.

Metadata delivery: The transmission module can transmit metadata such as user ID, sending time information, and others in addition to the image itself.

Interaction with other subsystems.

Interaction with the server subsystem: The client subsystem interacts with the server part, sending images for processing and receiving results. This communication occurs over the network using a protocol that corresponds to this, such as HTTP or HTTPS. [25]

Interacting with the customer database: You may need to interact with the database to retrieve or update user information.

Information protection.

Data encryption: Encryption can ensure the security and privacy of data transmission. This is especially important when images are transmitted over open networks such as the Internet. [26]

User authentication: an important component of security involves, that only authorized users can use the system.

Efficient data transfer for further processing on the server side and user convenience should be the main goals of developing the client subsystem.

The server subsystem for CLIP-based image processing software consists of many components that are responsible for receiving, processing, and analyzing images from users. Let's look at the main components that make up this subsystem:

Image reception and processing.

Image Receiver: This component receives images from the subsystem client. It must have methods for receiving and partially parsing the file.

Processing module: processes the image using the CLIP model. Preprocessing, further analysis, and output of results may be part of this stage.

Interaction with the database.

Saving results: Processing results, such as defined image-based queries, can be stored in a database for future analysis or use.

Customer database queries: If the system has the ability to store and retrieve information about users, the server can contact the database to retrieve the required information.

Interaction with the customer service subsystem includes a request processing module that receives and processes customer requests. Usage
This module involves transferring images for processing and sending the results back.

Interaction protocols.

To ensure efficient data exchange with clients, the server must use appropriate communication protocols, such as HTTP or HTTPS.

Information protection.

Connection protection: provides encryption and connection security to ensure prevent unauthorized access.

Authentication and Authorization: Ensures that users are truly identified and allows them to access only the resources that are necessary.

Event Monitoring and Recording: Event logging allows you to record events that occur on the server for further analysis and identification of potential problems.

Resource monitoring: a monitoring system that tracks usage server resources to quickly identify problems.

The development of the server subsystem is to ensure efficient and secure image processing, as well as reliable network interaction with clients.

Communications between subsystems: The client subsystem communicates with the server subsystem over the network, sending data for processing and receiving results.

The server subsystem interacts with the client database to collect and receive user data.

The image transfer module is responsible for the efficient transfer of images from the client to the server.

This architecture allows you to create an efficient and scalable client-server application for image processing based on the CLIP model, providing a user-friendly user interface and high-performance server-side processing.

2.1.3. Designing the data storage subsystem

2.1.3.1. Choosing a data model and approach to database design

Given the specifics of our project, it was decided to use relational data model. This decision is justified by several factors:

–Data structuring: Our system data is structured and consists of image information, processing results, metadata, and text prompts (PROMPT). Using tables and relationships between them, relational databases (RBDs) allow us to efficiently manage such structured data.

–Data Integrity: Relational databases maintain reliable data using foreign keys and integrity constraints. This is important for our system because we are dealing with interrelated data such as images and their textual clues.

The “Database first” approach was chosen for database design. This method involves creating a database with structures and tables, and then creating data models based on the existing database schema. The main advantages of using the “Database First” method in our project are: [27]

–Control over database schema: When we create a database, we have complete control over its structure. This allows us to carefully plan and optimize the schema to ensure high performance and efficient data storage.

–Ensuring compliance with business requirements: By first designing the database schema, we can ensure that the data meets the business requirements and that it has the correct structure that can be processed and analyzed.

–Ease of integration: Using a “Database First” approach makes it easier to integrate an existing database with other systems and tools; this can be important for extending functionality or integrating with others in the future.

Implementing a Database First approach

The process of implementing the "Database First" approach includes several key stages:

Database Schema Design: The database schema is now being created, including defining all the required tables, their attributes, and the relationships between them. For example, the Images table might contain fields to store data such as the date the file was uploaded and information about the file. To allow the analysis results to be stored, the Images table will be linked to the Text Prompts (PROMPT) table using a foreign key.

Database Development: Database Management System (DBMS) such as MySQL and PostgreSQL, is used to create the actual database based on the designed schema. Additionally, indexes, triggers, and other optimizations needed to ensure high performance can be found during this process. [28]

Data model generation: After the database is created, data models are generated to be used in the software. This can be done using ORM (Object-Relational Mapping) tools such as Entity Framework for C# or SQLAlchemy for Python. [29] These tools automatically generate model classes based on the existing database schema.

Software Integration: The generated data models are integrated with software to enable interaction with the database. This includes implementing methods for loading, storing, and processing data, as well as integration with the CLIP module for image analysis and textual cue generation.

2.1.3.2. Data model design and development

As a result of the analysis of software requirements in the previous paragraphs, The following generalized entities were identified, described below:

–Users
–Payments
–Subscription offers
–Models (CLIP models that can be downloaded)

After normalizing the database [29] and forming relationships between entities, a conceptual data model of the subject area “Image Processing Application” was built, which is shown in Figure 2.1.

Figure 2.1 – ER-model of subject domain data for the ImgProPlus software

Here is a description of the selected entities and their attributes in tabular form (see Table 2.1)

Table 2.1 – Description of ER model entities

Essence	Primary key	Attributes
Users – users systems	username– login user	password – password
Auth Tokens – verification tokens	username – login user	authority – role name
Payments	payment_id – payment code	username – user login, payment_date – payment date, promotion_id – subscription code
Confirmation Codes – codes confirmation	id – identification number	username – user login, code – confirmation code, expiration_date – date destruction

End of table 2.1

Essence	Primary key	Attributes
Models	model_name – model name	description – model description, premium – model availability to users
Credits – user credits	username – login user	credits – number of user credits without subscription
Promotions – offers	promotion_id – offer code	subscription_length – subscription length (in months), cost – subscription cost (in USD), description – offer description, subscription_length_ua – length of the subscription in Ukrainian, description_text_ua – description of the subscription in Ukrainian.
Subscribers – subscription status	username – login user	end_date – subscription end date
Requests – requests	request_id – request code	username – user login, request_text – request text

After evaluating the entities and attributes, a logical data model was developed, which is shown in Fig. 2.2. (see Fig. 2.2)

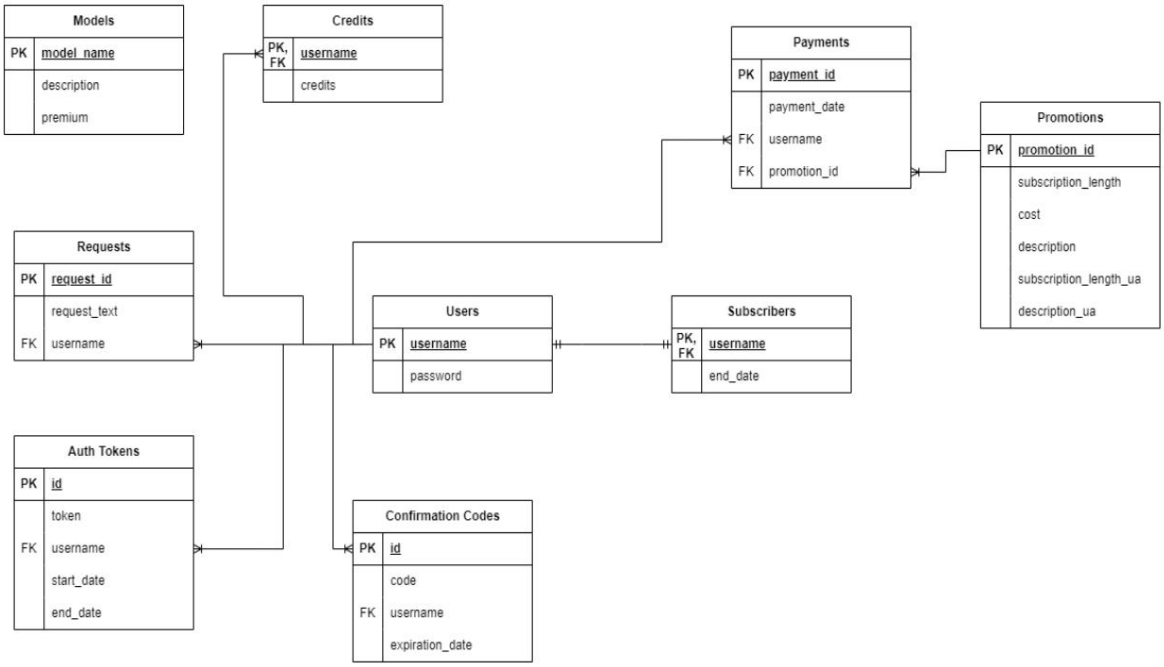

Figure 2.2 – Logical data model of the subject area “Application for image processing"

With the help of a carefully formed ER model and logical data model, we can form a physical data model, with a convenient distribution by data type, to save space on the hard disk. [30]

For convenience, the physical model is represented by an image generated in MySQL Workbench, which creates a diagram based on an existing database. The physical model is shown in Figure 2.3. (see Figure 2.3)

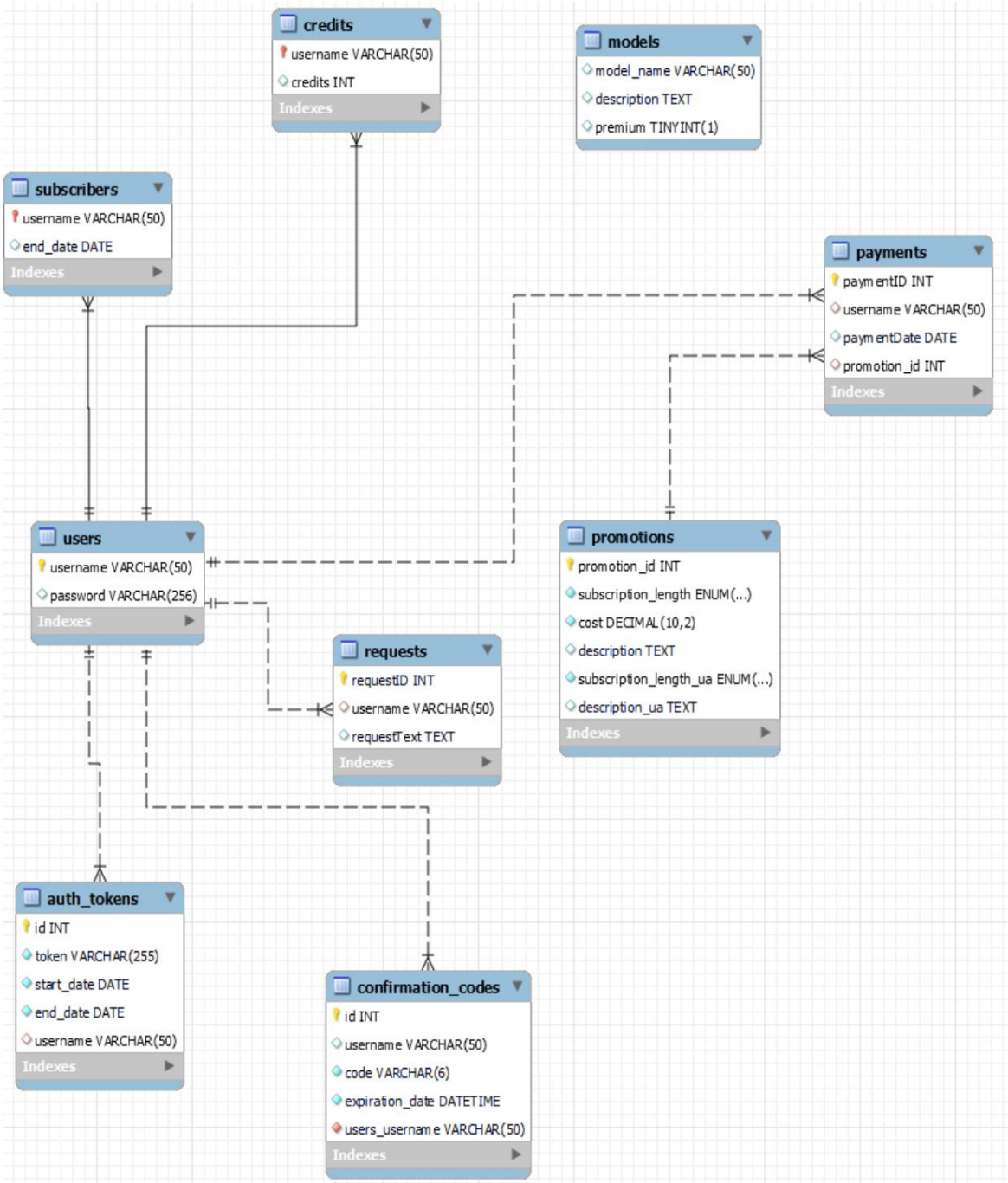

Figure 2.3 – Physical data model

2.1.3.3. Interaction with the database

The application will interact with the database using mysql.connector, a library written by Oracle, which allows you to quickly and easily process requests from the application.

The algorithm of the database operation (which will be written in the database.py class, and discussed in more detail in section 2.4) is presented in Figure 2.4 (see Figure 2.4) [31]

Figure 2.4 – Algorithm for processing a query in the database using mysql.connector

The configuration file will store data for connecting to the database. The json file format allows you to conveniently change the data as needed and work with it in the program.

To process the query, the execute_query function was written, which allows the developer to flexibly execute various calls to the database. The code for this function is shown in Figure 2.5. (see Figure 2.5.)

```python
def execute_query(self, query, params=None) -> list | bool:
    """
    Execute a MySQL query and return the results.

    Args:
        query (str): The SQL query to execute.
        params (tuple, optional): A tuple of parameters to substitute into the query.

    Returns:
        list: A list of tuples containing the rows returned by the query.
        bool: True if the query executed successfully, False otherwise.

    This function executes the provided SQL query with optional parameters and returns
    the results as a list of tuples. It also logs any errors that occur during execution.
    """

    cursor = self.connection.cursor()
    try:
        cursor.execute(query, params)
        results = cursor.fetchall()
        self.connection.commit()
        if query[:6] == "SELECT":
            return results if results else False
        return results if results else True

    except mysql.connector.Error as error:
        self.logger.error(f"Error executing query: {error}")
        return False

    finally:
        cursor.close()
```

Figure 2.5 – Code and documentation of execute_query

After returning the results, the developer interacts with them using other functions that are part of database.py and are listed in Appendix A (see Appendix A).

2.1.3.4. Developing database queries

Having created a physical data model and worked out how the application will interact with the database, in section 2.1.3.3, we can generate code and queries to the database. Only fragments of the code are given in this section, the full code is in Appendix A (see Appendix A).

A fragment of table creation is shown in Figure 2.6.

```
-- Create Requests table
CREATE TABLE Requests (
    requestID INT AUTO_INCREMENT PRIMARY KEY,
    username VARCHAR(50),
    requestText TEXT,
    FOREIGN KEY (username) REFERENCES Users(username)
);

-- Create auth_tokens table
CREATE TABLE auth_tokens (
    id INT AUTO_INCREMENT PRIMARY KEY,
    token VARCHAR(255) NOT NULL,
    start_date DATE NOT NULL,
    end_date DATE NOT NULL,
    username VARCHAR(50),
    FOREIGN KEY (username) REFERENCES Users(username)
);

-- Create confirmation_codes table
CREATE TABLE confirmation_codes (
    id INT AUTO_INCREMENT PRIMARY KEY,
    username VARCHAR(50),
    code varchar(6) NOT NULL,
    expiration_date DATETIME NOT NULL,
    UNIQUE KEY unique_code (code)
);
```

Figure 2.6 – Fragment of creating tables in the database

I chose these tables because I needed to create triggers (events) for them that would automatically delete stale data. [32] Figure 27 shows the code for the triggers.

```
-- EVENTS

-- Delete expired hashes event
DELIMITER $$
CREATE EVENT delete_expired_hashes
ON SCHEDULE
    EVERY 1 DAY
COMMENT 'Delete expired hash records from the auth_tokens table'
DO
BEGIN
    DELETE FROM auth_tokens
    WHERE CURRENT_DATE() NOT BETWEEN start_date AND end_date;
END$$
DELIMITER ;

-- Delete expired codes event
DELIMITER $$
CREATE EVENT delete_expired_codes
ON SCHEDULE EVERY 1 MINUTE
DO
  DELETE FROM confirmation_codes WHERE expiration_date < NOW();
DELIMITER ;
```

Figure 2.7 – Creating triggers to clear the auth_tokens and confirmation_codes

To interact with INSERT and SELECT commands, the execute_query function was written, which is discussed in the previous paragraph, but, for clarity, the call to this function will be demonstrated with the INSERT and SELECT commands, respectively. The functions and their documentation in which the call occurs are shown in Figures 2.8-2.9. (see Figures 2.8-2.9)

For example, the verify_confirmation_code function is given, which interacts with the confirmation_code table and retrieves the required code from there using the SELECT command. code.

```python
def verify_confirmation_code(self, username: str, code: str) -> bool:
    """
    Verify a confirmation code for a user.

    Args:
        username (str): The username of the user.
        code (str): The confirmation code to verify.

    Returns:
        bool: True if the confirmation code is valid for the user, False otherwise.

    This function verifies the provided confirmation code for the user specified by the
    username. It checks if the code exists in the 'confirmation_codes' table in the database
    for the given username and returns True if the code is valid, False otherwise.
    """
    try:
        query = "select code from confirmation_codes where username = %s and code = %s"
        results = self.execute_query(query, (username,code))
        if results:
            self.logger.info('verify_confirmation_code executed succesfully!')
            return True
        else: return False
    except Error as e:
        self.logger.error(f"Something went wrong during verify_confirmation_code function. Error - {e}")
```

Figure 2.8 – SELECT call in the verify_confirmation_code function

For the following example, the generate_confirmation_code function is given, which generates a temporary code and enters it into the database. (see Fig. 2.9)

```python
def generate_confirmation_code(self, username: str) -> str:
    """
    Generate a confirmation code for a user.

    Args:
        username (str): The username of the user.

    Returns:
        str: The generated confirmation code.

    This function generates a confirmation code for the user specified by the username.
    It inserts the code into the 'confirmation_codes' table in the database and returns
    the generated code.
    """
    try:
        code = str(randint(100000, 999999))
        query = "INSERT INTO confirmation_codes(username, code, expiration_date) VALUES (%s, %s, %s)"
        expiration_date = datetime.now() + timedelta(minutes=10)
        results = self.execute_query(query, (username, code, expiration_date))
        if results:
            self.logger.info('generate_confirmation_code executed succesfully!')
            return code
        else: return False
    except Error as e:
        self.logger.error(f"Something went wrong during generate_confirmation_code function. Error - {e}")
```

Figure 2.9 – INSERT call in the generate_confirmation_code function

UPDATE and REMOVE calls will not be discussed in this section.
They are available in the functions delete_user, reset_password, etc. (see Appendix A)

2.1.4 Designing interaction with external services

Since the application uses a client-server architecture, for
For convenience, our own communication model based on HTTPS was developed.

Using the built-in socket library for the Server class, the user sent data that was encrypted using SSL and a key.
[33] were encoded and transmitted securely over the Internet. The basic operation of the algorithm is shown in Figure 2.10. (fragments with encryption and obtaining keys were discarded, for the sake of visualizing the operation of the client-server system directly)

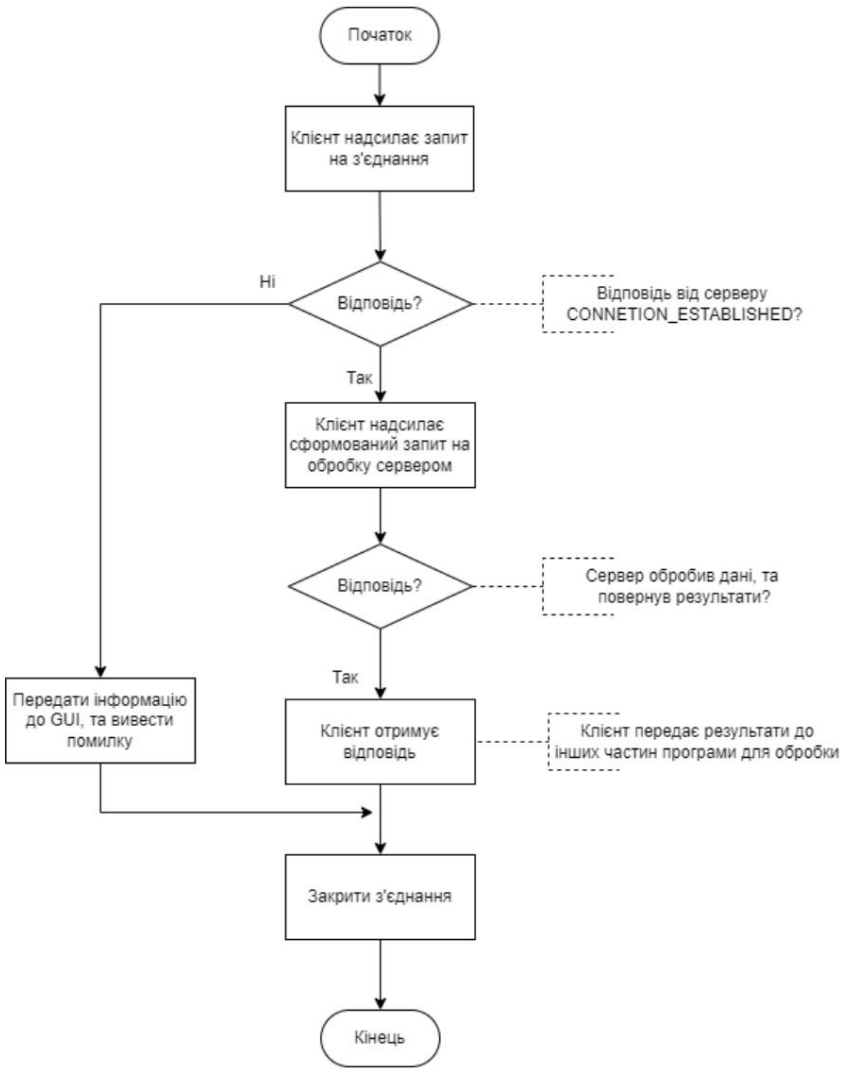

Figure 2.10 – Server-client interaction

Since it was decided not to use GET and POST calls, the request format was redesigned, and a custom one was developed, which is more convenient for deployment in this project. [34] What the call looks like is shown in Figure 2.11. (see Figure 2.11)

Figure 2.11 – Structure of a request to the server used in additional

Let's consider the components of this request. For convenience, the data will be entered into table 2.2. (see table 2.2)

Table 2.2 – Query components

Component Length		Description
HEADER 3 characters		Specifies the purpose of the call to the server, by which the server will distribute and understand what exactly the client wants. For example, "UPD" is a request to check for updates for the application.
<SEP>	1 character	Defines the end of one part of the query and the beginning of another. For this project, the "|" (vertical bar) was chosen.
DATA	N characters	Consists of data that the client sends for validation. May be empty in some requests. For example, "EXT", which means a request to get a list of available models, should not receive a DATA field.

Since HTTPS is used in addition, let's pay attention to how the TLS handshake works. [35] The algorithm for client-server interaction during HTTPS is shown in Figure 2.12. (see Figure 2.12)

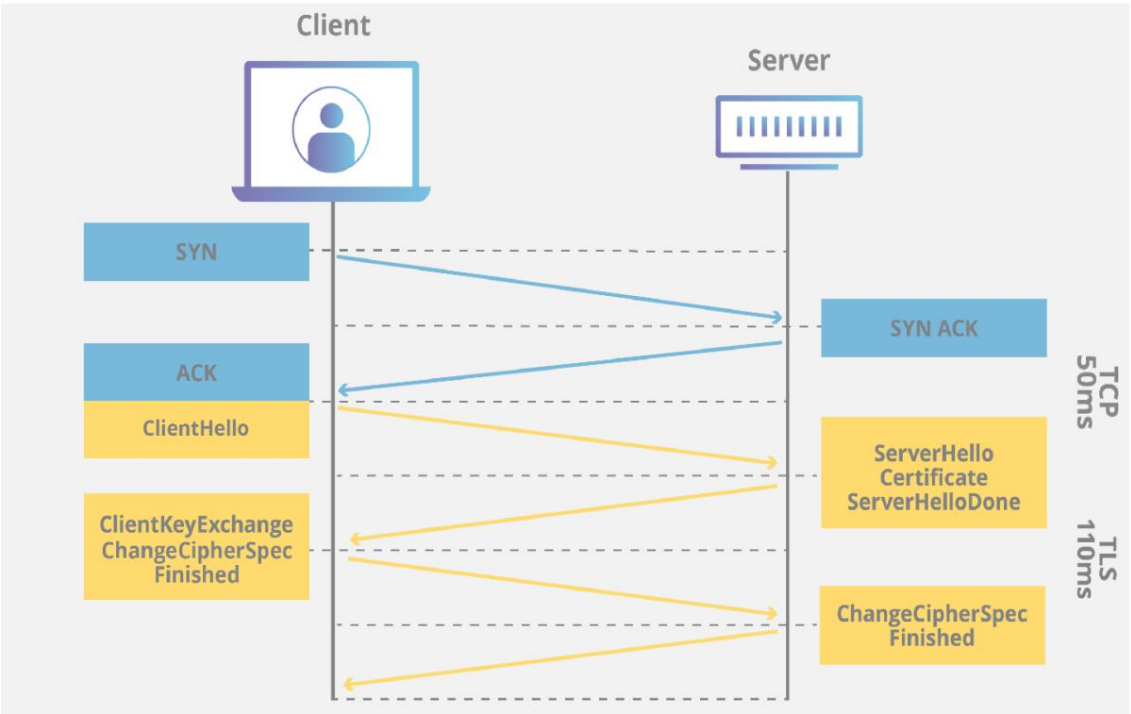

Figure 2.12 – TLS handshake algorithm

The client, when started, sends a "hello" message to the server, which contains data about the TLS certificate, etc. Also, in this message, a random set of bytes will be stored, called the "client random". In response, the server sends an SSL certificate. [36]

After receiving the certificate, the client checks this certificate in the certification center, confirming that it interacts with a server with a valid domain. (in our case, with the server of the application "ImgProPlus")

If the certificate is successfully verified, session keys are created to verify that the keys were generated correctly. After this stage, the server and client are ready to exchange data.

Throughout the algorithm, asymmetric cryptography (public and private keys) is used. Therefore, it is important to generate a key pair when deploying the server for further work with clients. Key generation is described in Appendix A.

Working with WebAPI (LiqPay service)

Since creating your own payment and transaction system is an incredibly complex and responsible job, it was decided to use an external financial system (this practice is used by companies such as Steam, OpenAI, X, etc.). The choice fell on the service "LiqPay", which is a Ukrainian payment service integrated with "PrivatBank". [37]

Only payment receipts will be stored directly in the database itself, so that in case of problems with receiving the premium, the administrator can resolve it and contact the user, acting as a middle-man, so to speak.

The API in LiqPay also uses asymmetric encryption, [38] which allows you to use the keys received in your own account to generate links to the necessary payment "mini-sites", where the user can pay using a convenient method. With the help of convenient documentation, this is quite easy to implement. Below is the interface of the ImgProPlus application
and how payment works. (see Fig. 2.13-2.14)

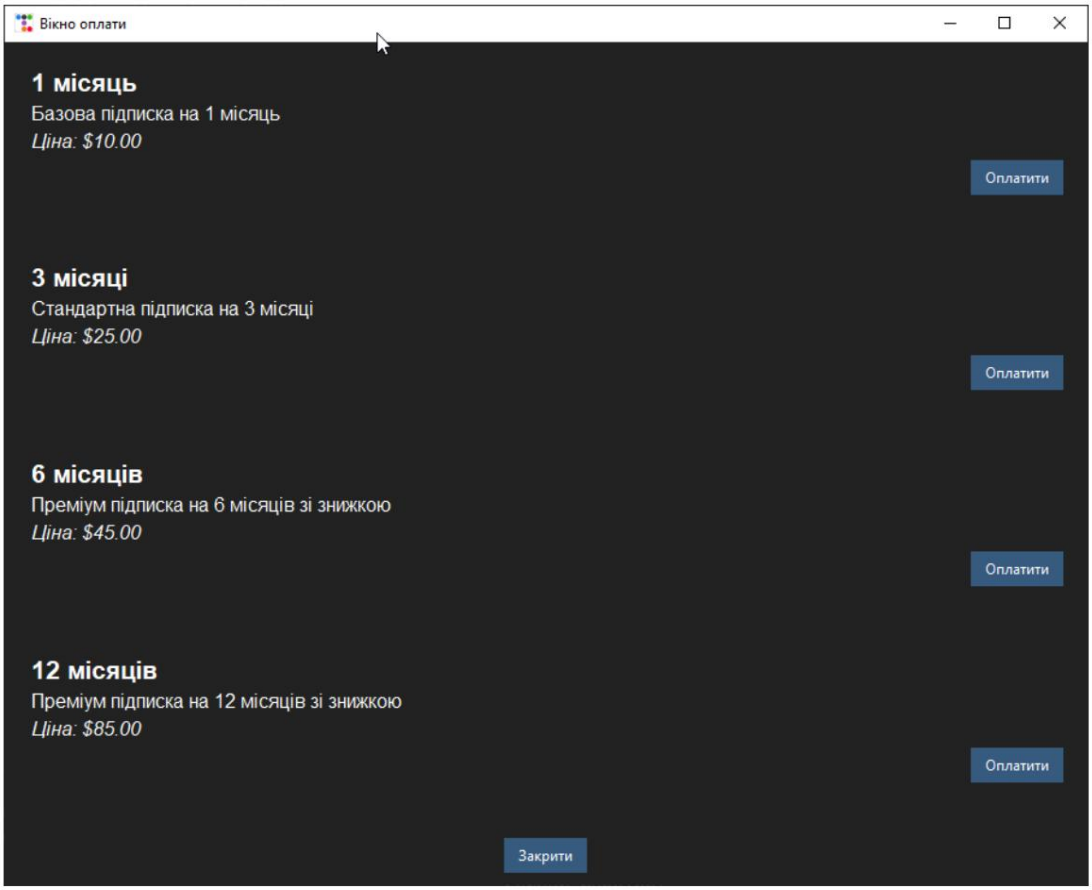

Figure 2.13 – Payment window

After clicking the "Pay" button, the user will see the following (see Fig. 2.14)

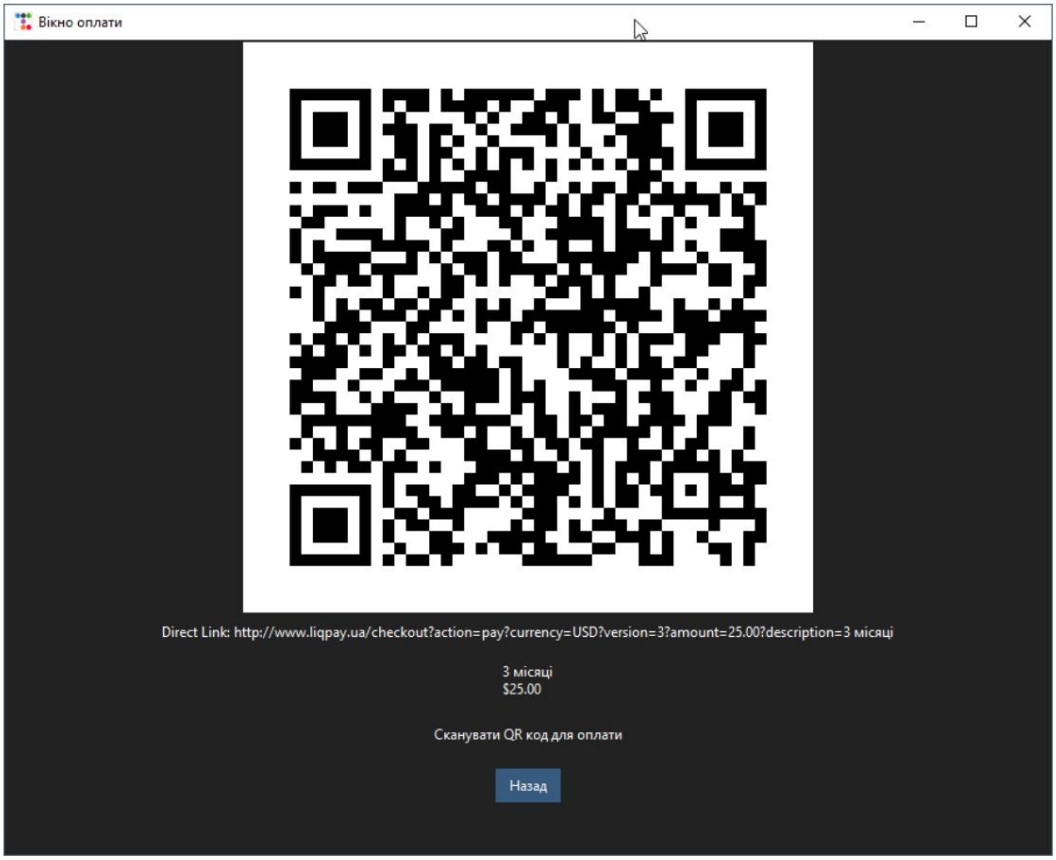

Figure 2.14 – generated code for three-month payment

The user can scan the code, or click on the link below, and go to the browser (if it is more convenient for the user to pay from a computer).

The code is generated as follows – using LiqPayAPI we generate a payment form (see Fig. 2.15), which uses the <form> tag in HTML, and transmit it to the user's browser or the user scans the QR code and pays from the phone. [38]

```
liqpay = LiqPay(public_key, private_key)
html = liqpay.cnb_form({
    'action': 'pay',
    'amount': '1',
    'currency': 'USD',
    'description': 'description text',
    'order_id': 'order_id_1',
    'version': '3'
})
```

Figure 2.15 – Example of using LiqPayAPI

With the speed and convenience of LiqPay, deploying an application has become much easier. (Not to mention that tax is automatically deducted from income, which makes the work of an individual entrepreneur easier).

2.2. Detailed design

Support service contact model

One of the business processes that will be used in addition is the ability to contact the support service. This process involves the user and the support service.

The user sends a request to the support team, and the support team responds; if the user is not satisfied with the response, the case is not closed until the problem is resolved. [39] If the user does not respond after 48 hours, the case is automatically closed.

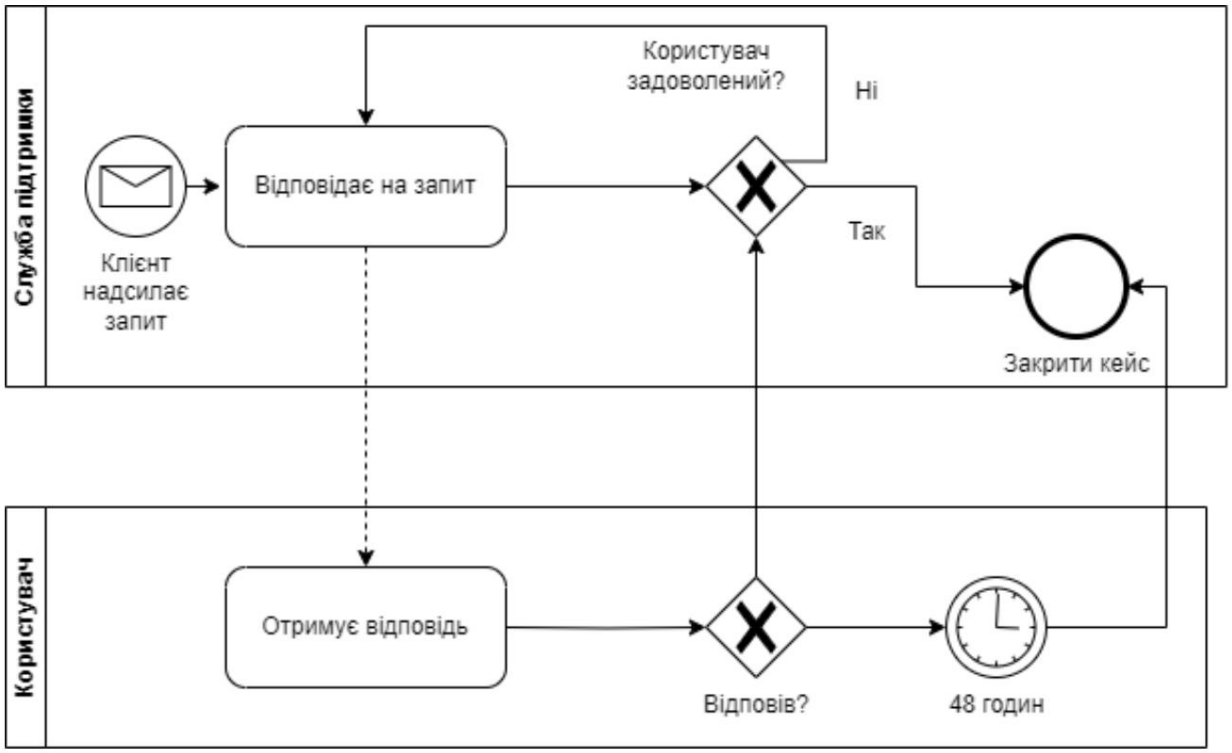

Figure 2.16 – Support service contact model

Subscription model

Another business process that will be used in addition is the ability to issue a subscription. This process involves the user, the application server, and the bank's API.

The user wants to purchase a subscription, the server provides a list of all available subscriptions, after selecting a subscription, they can buy or refuse it. When purchasing, the bank's API is also involved, which processes the payment and provides the relevant information to the application.

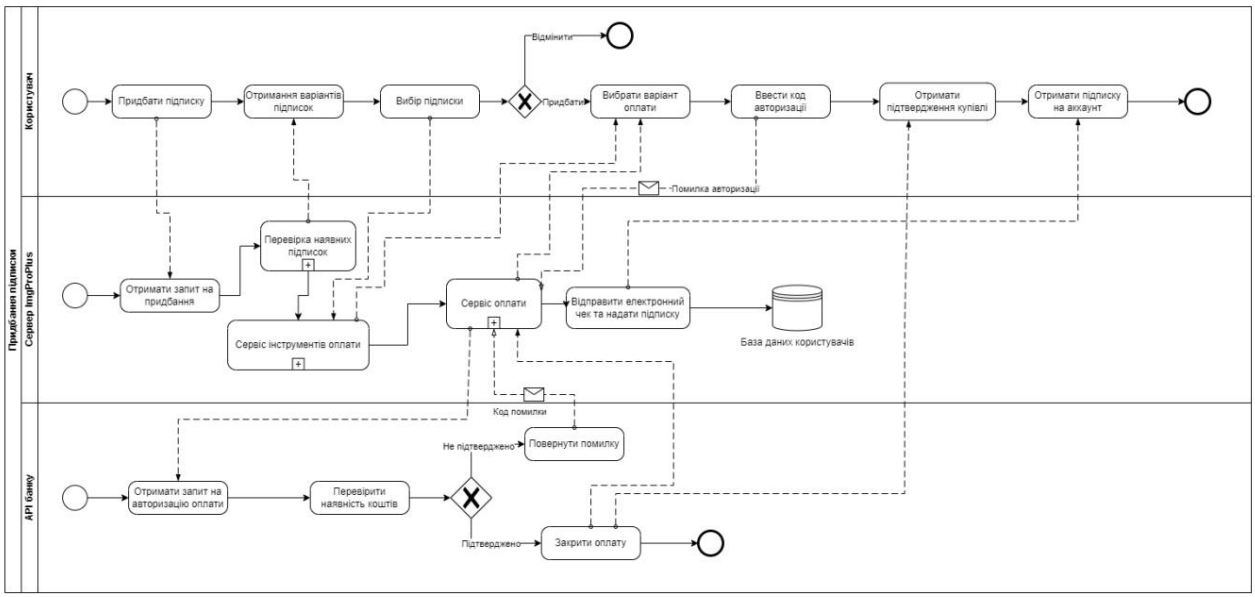

Figure 2.17 – Subscription purchase model

Registration model

In addition, the business process of user registration will also be used. The user and the server are involved in the process. The user tries to register and enters data, the server receives this request and responds (in different cases in different ways). As a result, if everything is correct, the user is registered and the server has saved his data in its database.

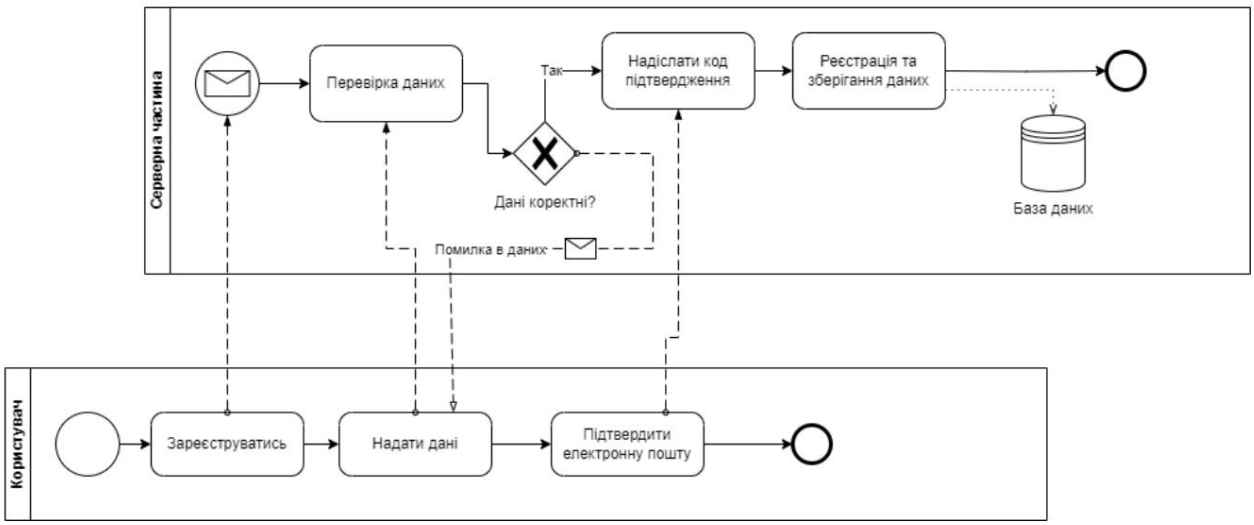

Figure 2.18 – Registration model

2.2.1. Structural models

According to the client-server model, the application consists of the app, server-side packages and the added clip package, which contains all the necessary utilities for image processing. [40] The structural model is presented in more detail in Figure 2.19. (see Figure 2.19)

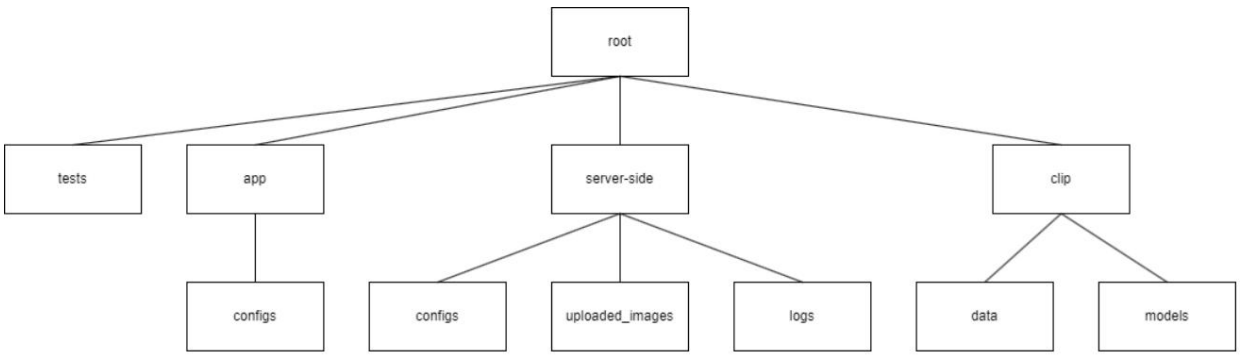

Figure 2.19 – Structural model of ImgProPlus application packages

The clip package contains the code files clip_interrogator.py (contents are given in Appendix A), and additional directories data and models. They store the data necessary for the CLIP models to work (.safetensors files and dictionaries, i.e. .txt files).

The server-side package contains the files for the server, i.e. server.py, database.py, and the application for administrators, admin_app.py. Additional

The configs, uploaded_images, and logs directories contain configuration files for the program and server, images uploaded by the user that are currently being processed, and server and database logs.

The app package contains the user application – app.py, client.py and other child files for the application to work. There is also a directory with configuration files for the application to work (user settings, token, cache, etc.).

The tests package contains tests for all other packages, but they are not part of the distributed software, as they are only used by developers to verify that the code is written correctly.

Here are UML diagrams for the clip_interrogator.py class. (Diagrams for GUI elements will not be provided in this section because the interaction between components is minimal, but they will be provided in section 2.3.) [41]

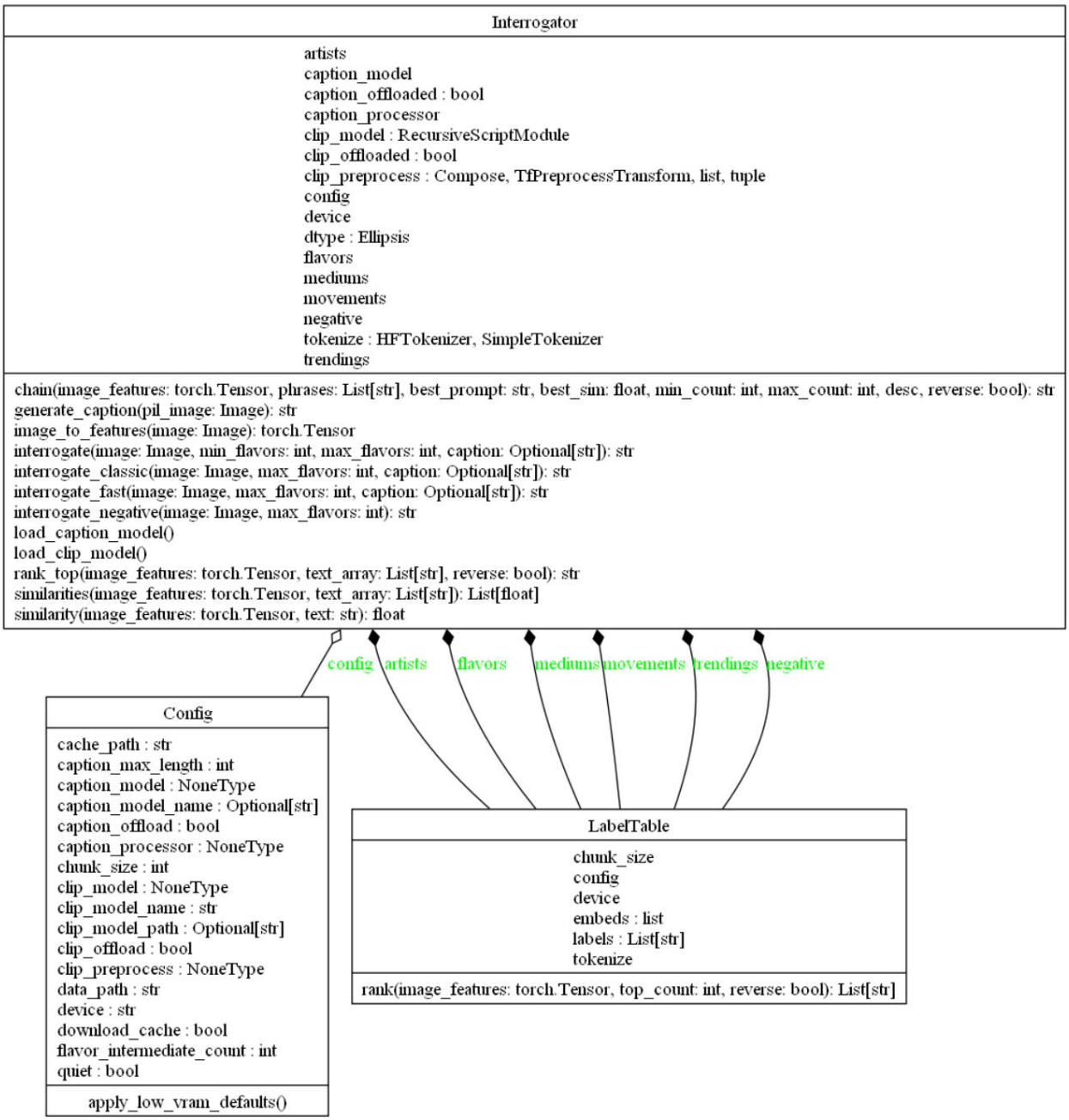

Figure 2.20 – UML diagram of clip_interrogator.py

2.2.2. Functionality and interaction modeling

The main functionality of the ImgProPlus application will be presented in the form of flowcharts that reflect the step-by-step logic and operation of the program. The key diagrams are presented in Figures 2.21 – 2.24.

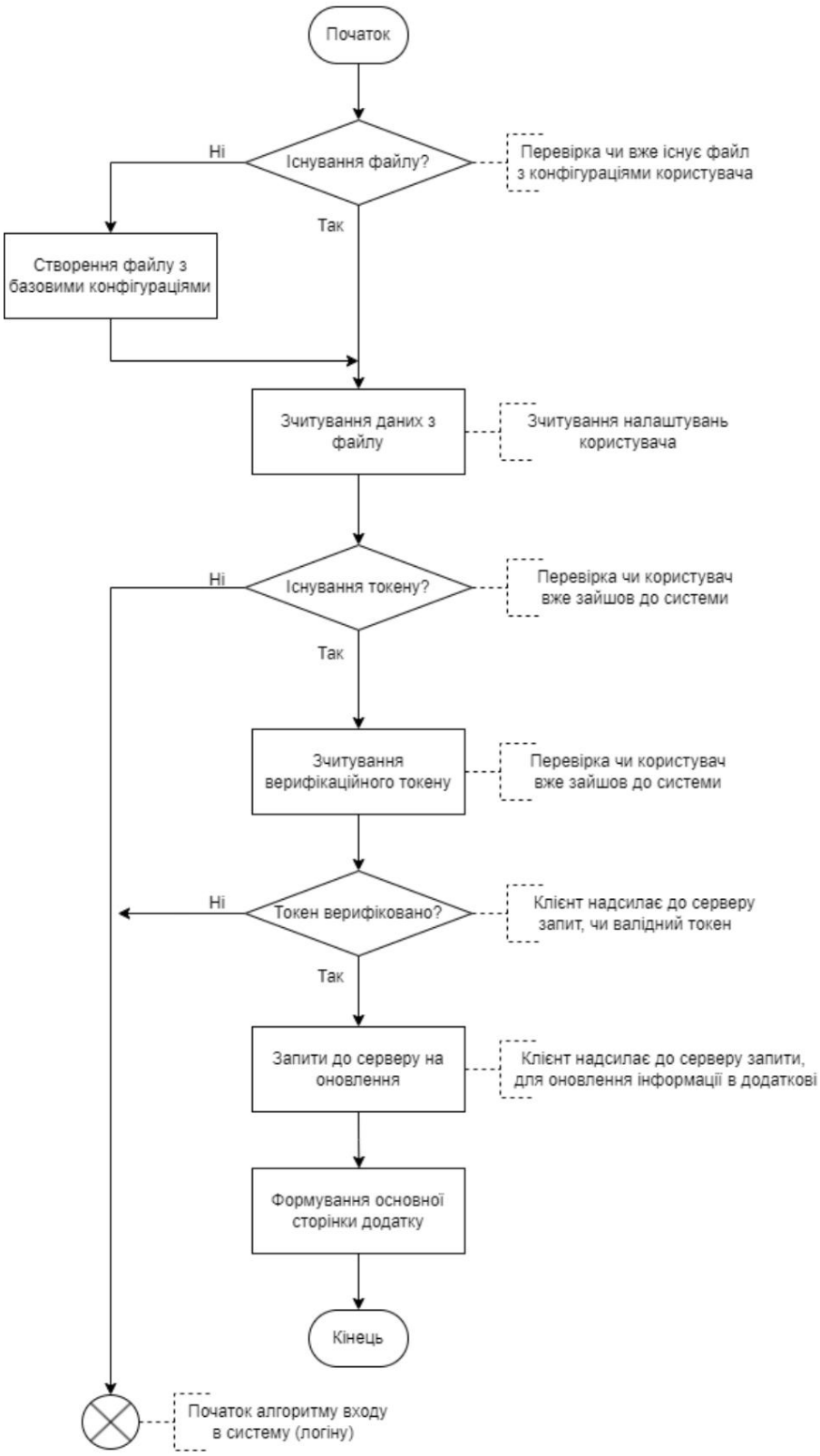

Figure 2.21 – Diagram of modeling the application's operation when the program is launched

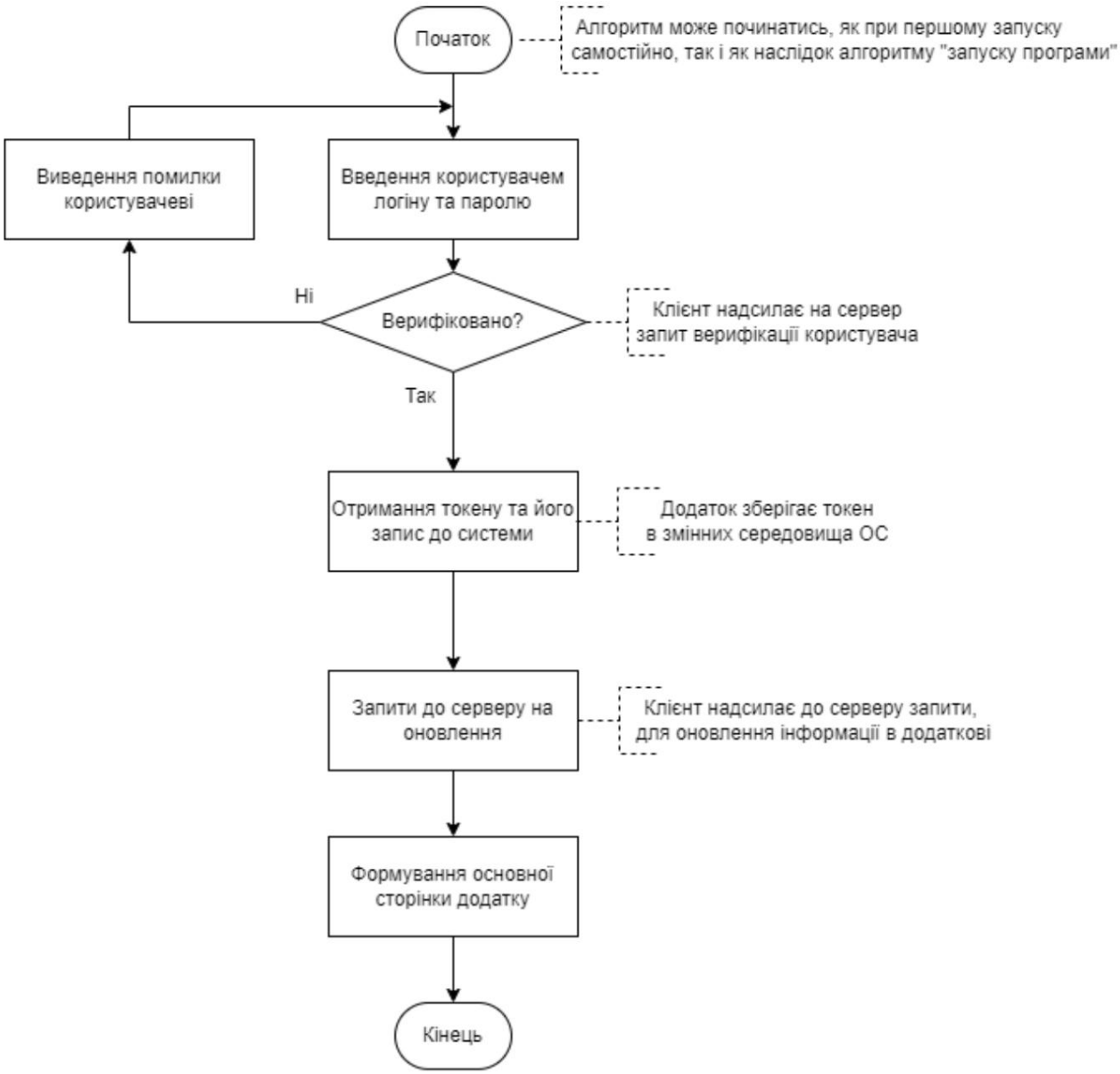

Figure 2.22 – Application login modeling diagram

Figure 2.23 – Image processing simulation diagram

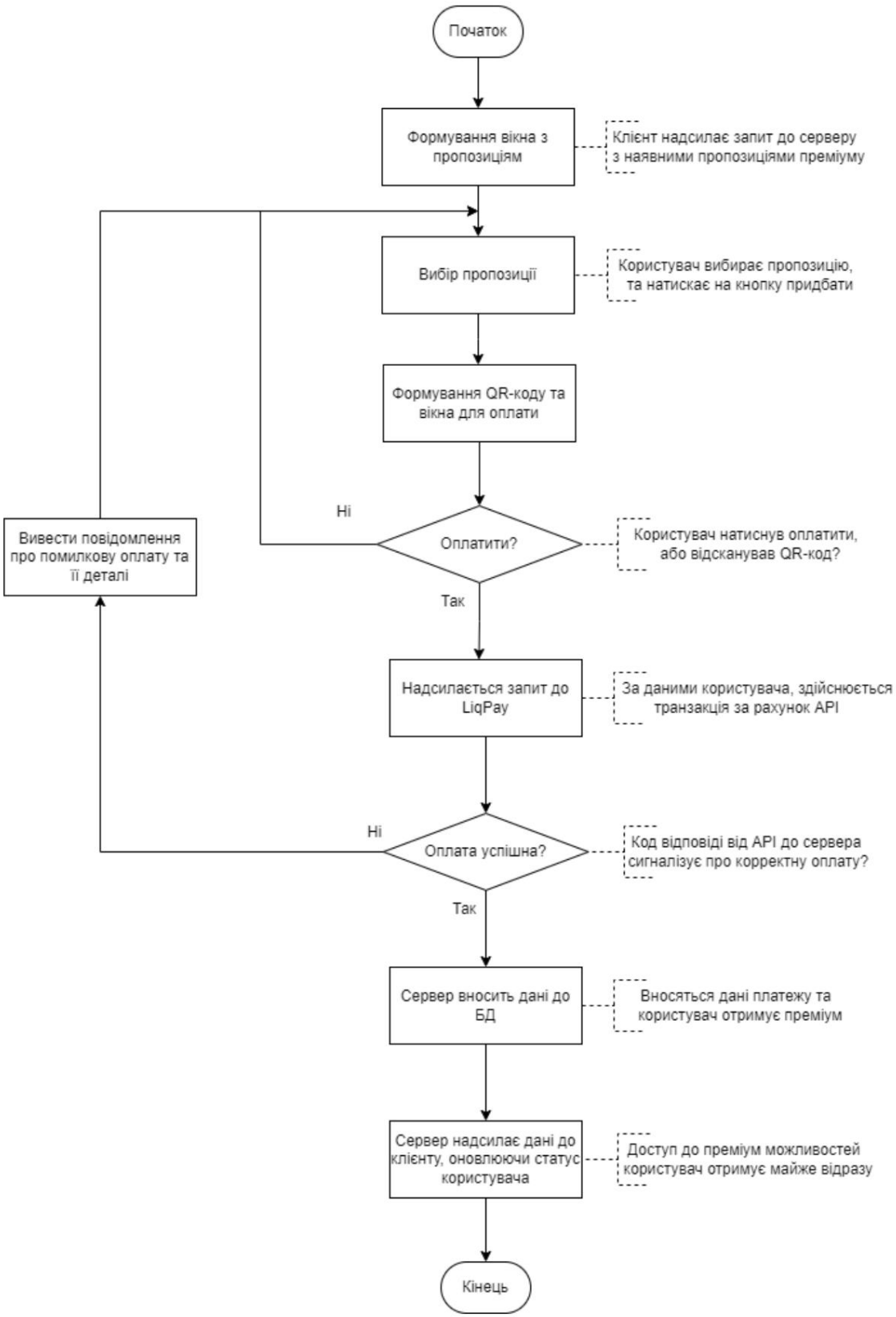

Figure 2.24 – Subscription Purchase Simulation Diagram

2.3. User interface design

According to clause 1.3.2.1.1, the user interface was written. The Transition Map [42] was implemented according to the diagram in Figure 1.5. To save space, it was decided to display the interface only in Ukrainian (the application supports English and Ukrainian languages, as specified in the requirements). The designed interfaces are presented in Figures 2.25 – 2.36.

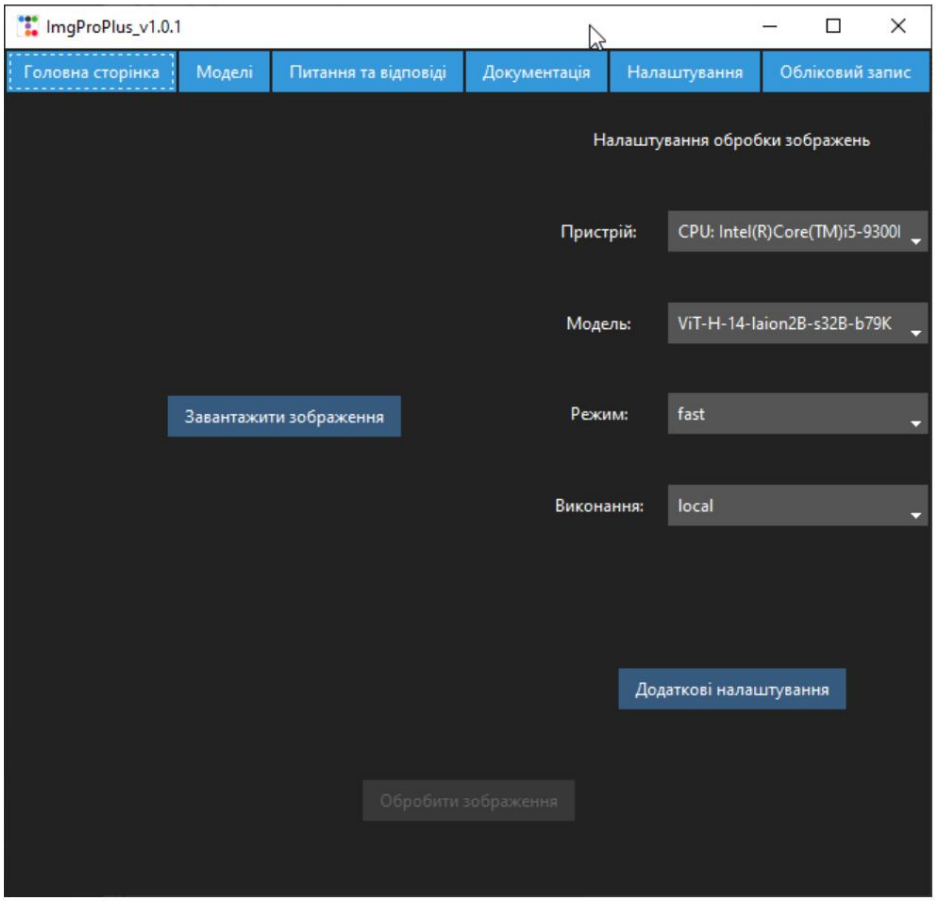

Figure 2.25 – Home window interface

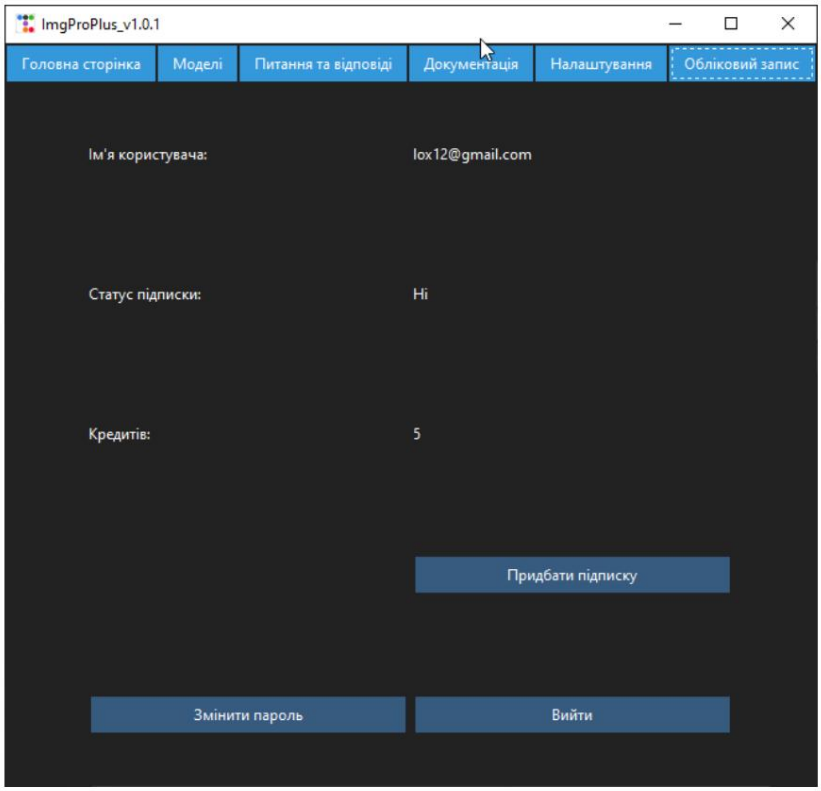

Figure 2.26 – Account window interface

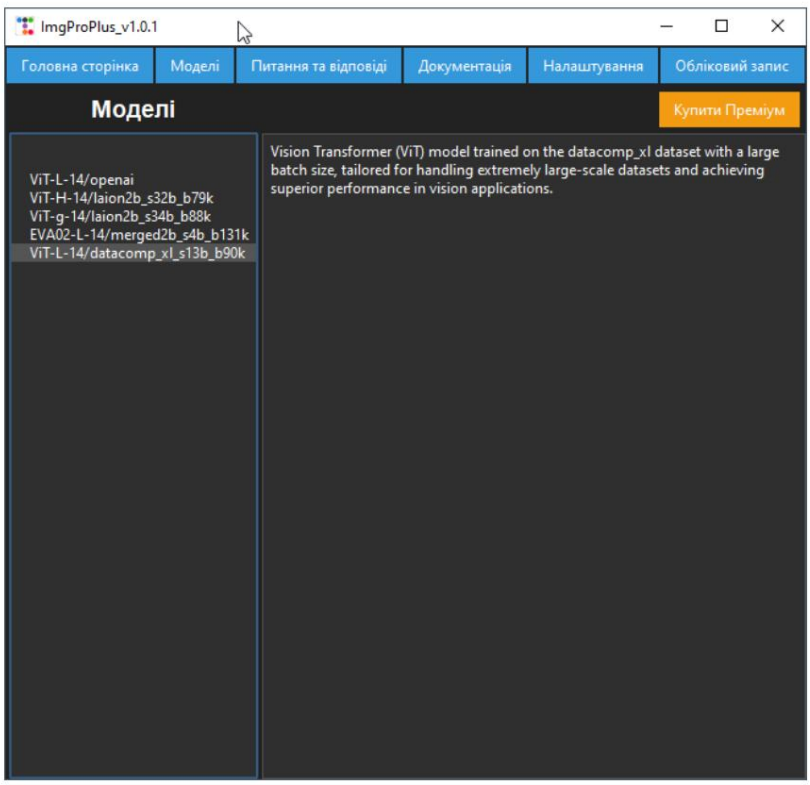

Figure 2.27 – “Models” window interface

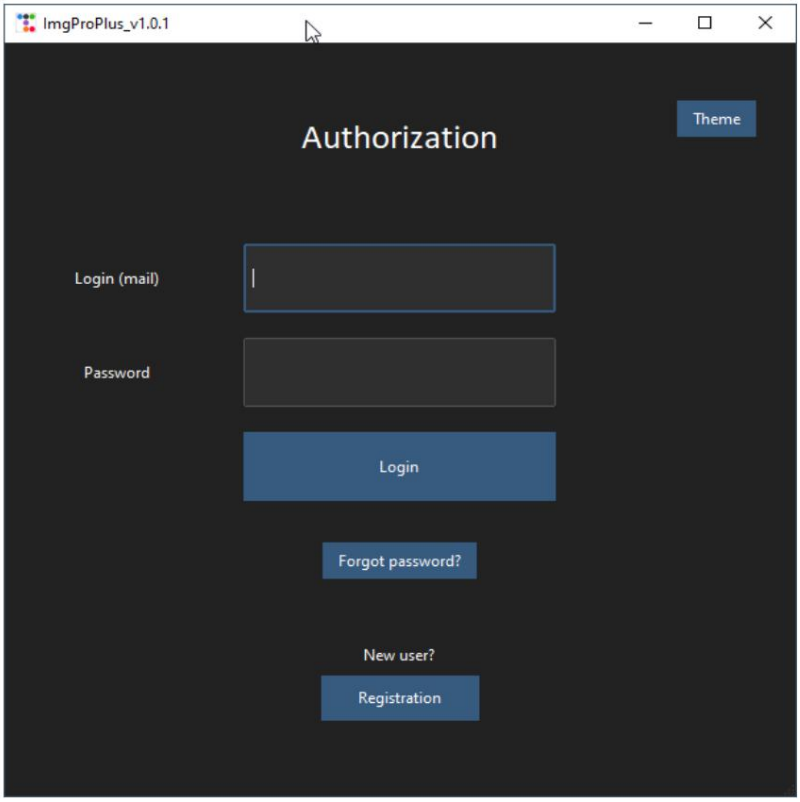

Figure 2.28 – “Authorization” window interface

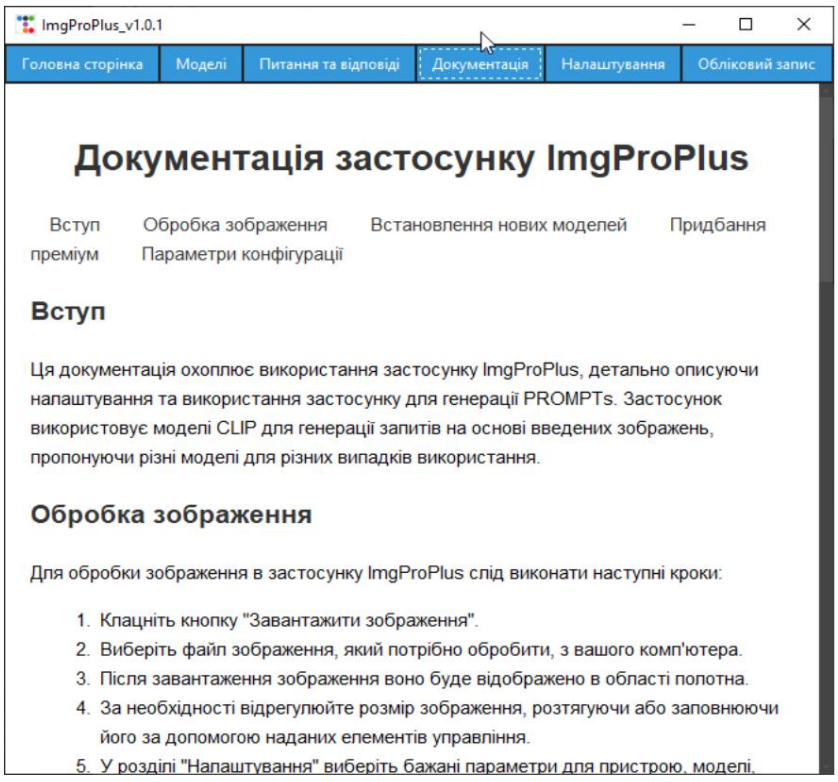

Figure 2.29 – Interface of the “Documentation” window

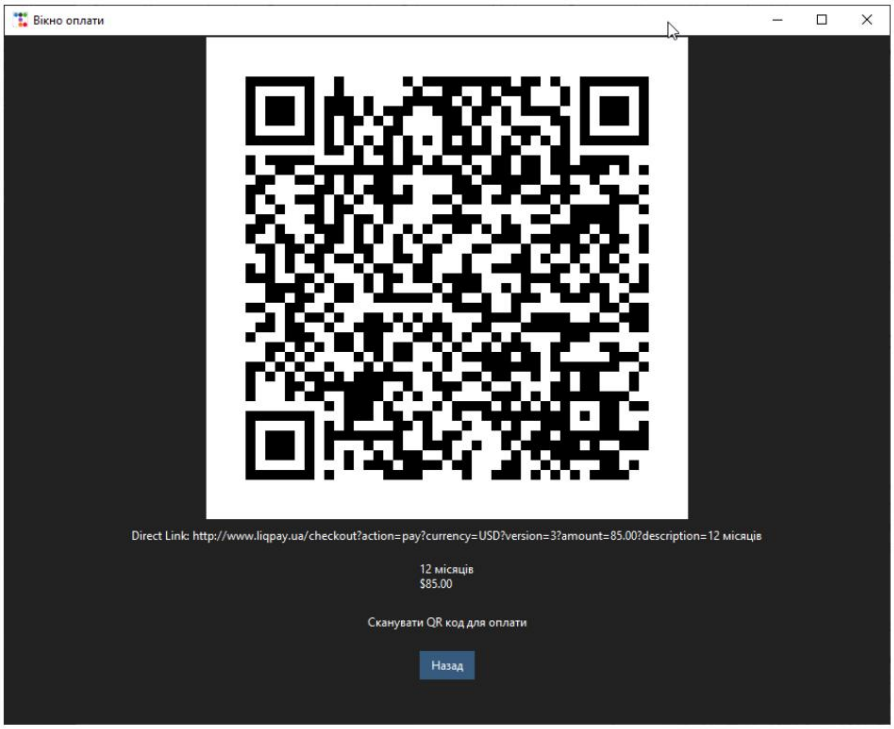

Figure 2.30 – Payment window interface

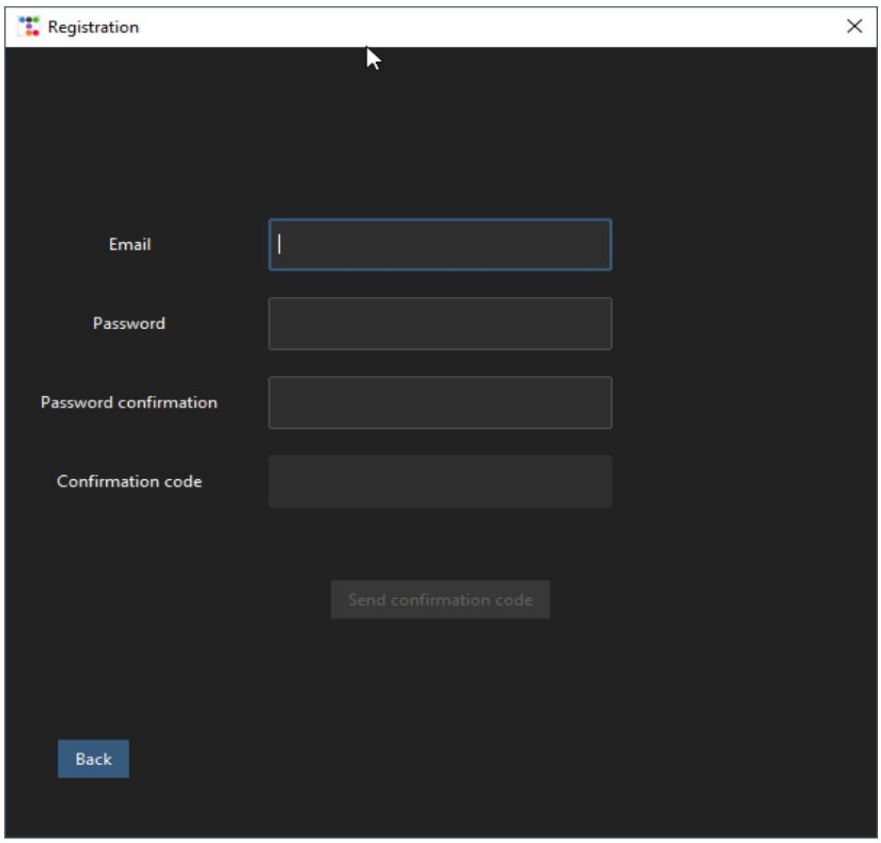

Figure 2.31 – “Registration” window interface

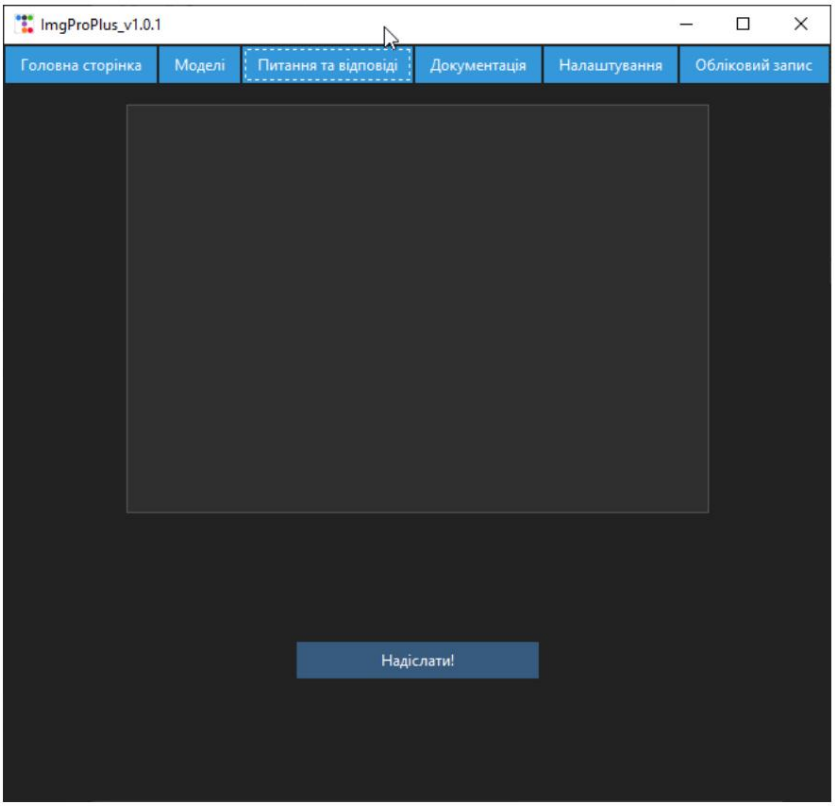

Figure 2.32 – Q&A Request window interface

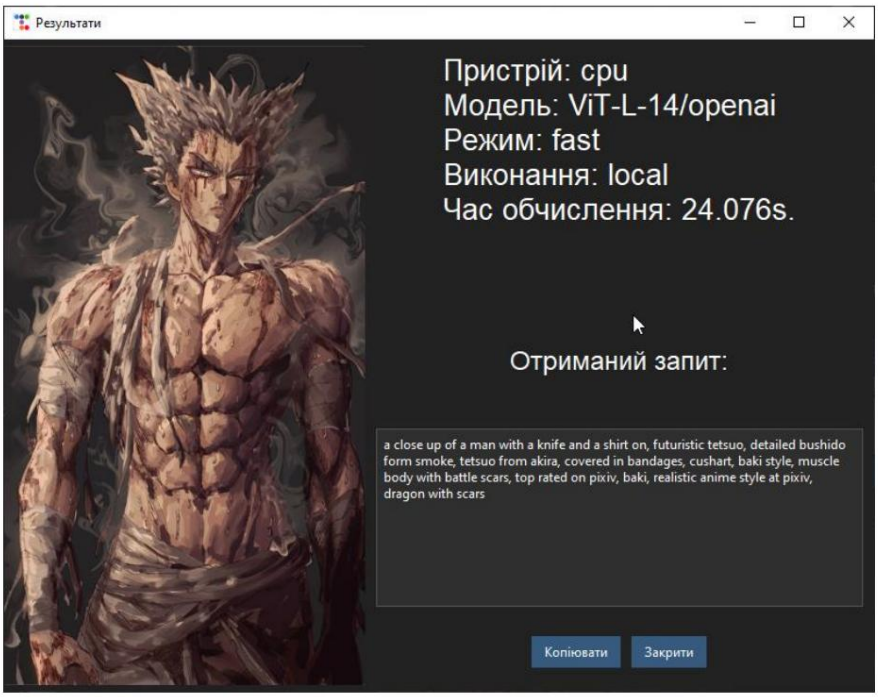

Figure 2.33 – “Results Page” window interface

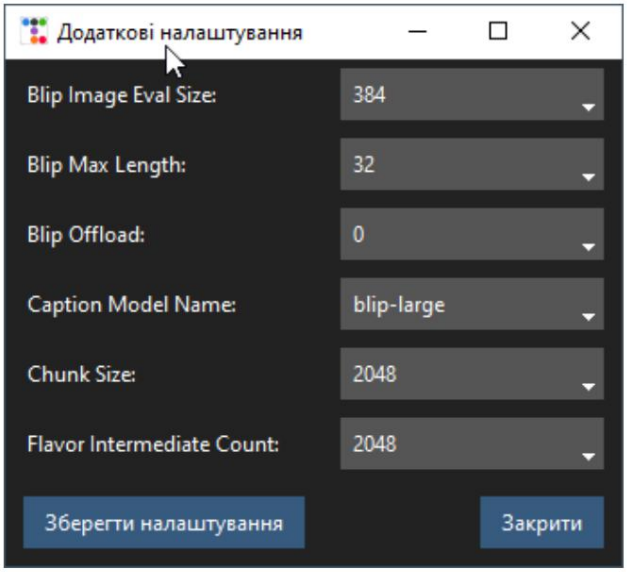

Figure 2.34 – Processing Settings window interface

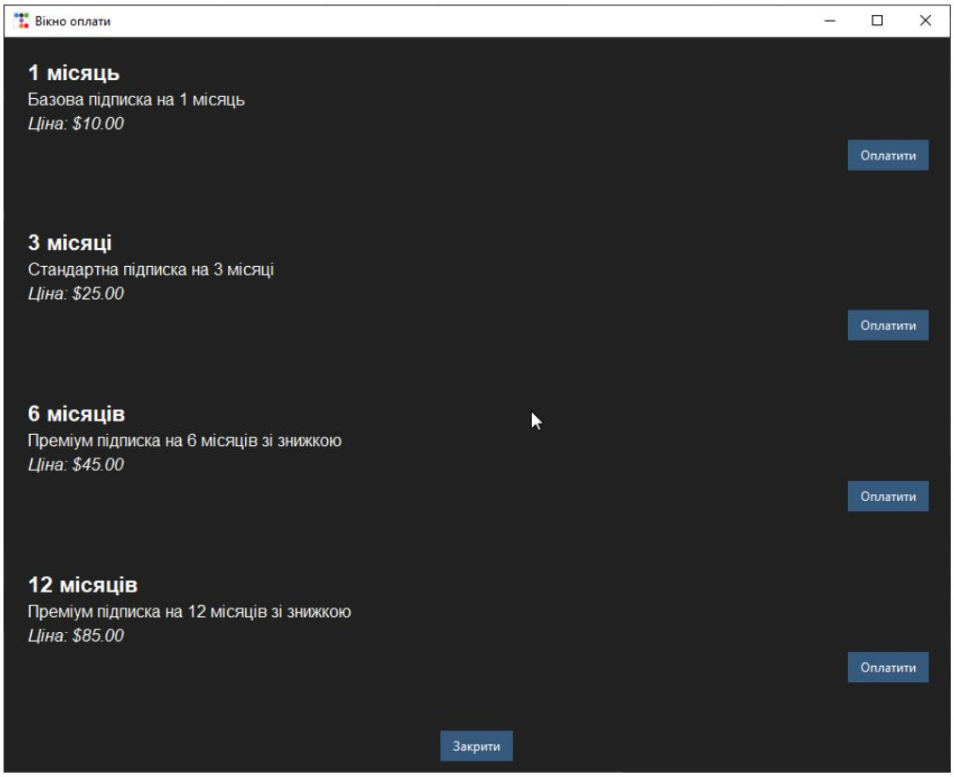

Figure 2.35 – “Purchase Subscription” window interface

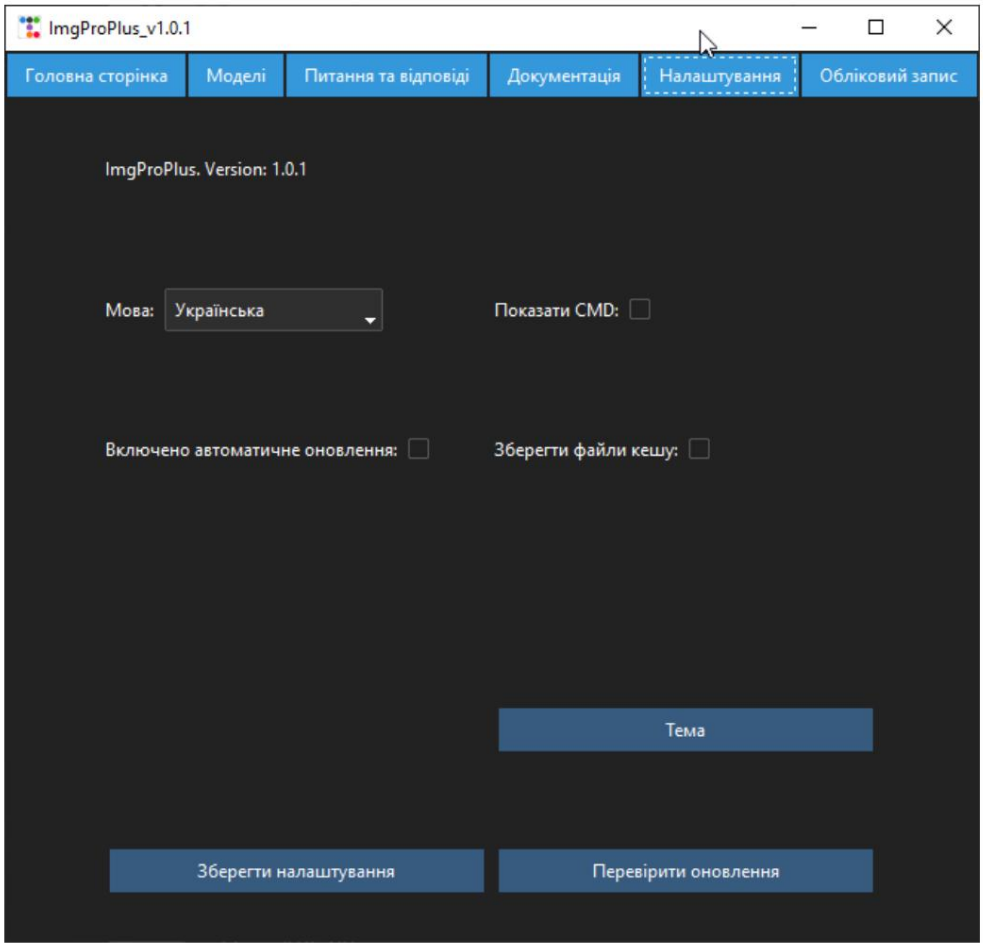

Figure 2.36 – Application Settings window interface

Since, upon first launch and registration, the application automatically creates settings for the user based on their system, the "Registration" and "Authorization" windows are provided in English, since the computer on which the development took place uses English as the primary language.

As we can see from Figures 2.25 – 2.36, the interface is designed in accordance with the requirements in section 1.3.2.1.

2.4. Software implementation

Code metrics

After writing the program, using the VS Code Counter utility [43] we obtained the corresponding metrics for each of the application directories.

In the clip directory, we have only one file, clip_interrogator.py. Metrics directories are shown in Figure 2.37. (see Figure 2.37)

Files

filename	language	code	comment	blank	total
clip/clip_interrogator.py	Python	396	5	74	475

Figure 2.37 – Clip directory metrics

As we can see, there are 401 lines of code and comments in total.

In the app directory we have a number of files, which are presented together with the metrics in Figure 2.38. (see Figure 2.38)

filename	language	code	comment	blank	total
app/addons.py	Python	64	7	16	87
app/app.py	Python	823	125	290	1,238
app/client.py	Python	57	18	23	98
app/docs.html	HTML	169	0	6	175
app/docs_u.html	HTML	169	0	7	176
app/localization.py	Python	116	9	21	146
app/login_page.py	Python	534	21	158	713
app/main.py	Python	73	15	16	104
app/my_logger.py	Python	10	3	5	18
app/settings.py	Python	46	3	19	68
app/slide_widget.py	Python	15	0	4	19

Figure 2.38 – app directory metrics

The app directory contains both Python and HTML files. In total
We will get 2277 lines of code and comments in the app directory.

The server-side directory has a number of files that are listed along with the metrics in Figure 2.39. (See Figure 2.39)

filename	language	code	comment	blank	total
server-side/admin_app.py	Python	592	73	191	856
server-side/database.py	Python	488	17	103	608
server-side/mail_serv.py	Python	21	1	5	27
server-side/my_logger.py	Python	10	3	5	18
server-side/server.py	Python	275	31	61	367

Figure 2.39 – Server-side directory metrics

All files are written in Python (excluding the configuration file, which is 4 lines long and is in JSON format). The total code and comments in the app directory are 2277 lines.

Implementation of requirements

After writing and implementing the software, it is necessary to check the quality of implementation of the specified requirements. Let's start with the functional requirements groups.

Fulfillment of functional requirements

FR1.01:

–Login to the application, registration, and obtaining a confirmation code are implemented in the login_page.py file (see Appendix A.13).

–User registration is carried out via email and password, which is implemented in the same file (see Appendix A.13).

–Checking the correctness of the entered data and displaying error messages is carried out in the files login_page.py and slide_widget.py (see Appendix A.13, and see Appendix A.12).

–The confirmation code is sent via the server described in
in the mail_serv.py file (see Appendix A.5).

FR1.02:

–User account management is implemented in the app.py file (see Appendix A.6).

FR1.03:

–Configuring user account settings is also done in the app.py file (see Appendix A.6).

FR1.04:

–Management of users and their accounts is performed in the implemented file administrative application, (see admin_app.py in Appendix A.4).

–Create, view, edit, and delete users
is carried out through this admin application. (see Appendix A.4)

FR1.05:

–Questions and answers management is implemented in the admin area application in the file admin_app.py (see Appendix A.4).

FR1.06:

–Payment management is performed in the admin application in the file admin_app.py (see Appendix A.4).

–Payment management page and payment detailing implemented in
in the same file (see Appendix A.4).

FR1.07:

–Viewing server logs is done through the admin application in the file admin_app.py (see Appendix A.4).

FR2.01:

–The ability for users to read the application documentation, which includes information about the functionality, settings, and instructions for using the application, is implemented in the docs.html file (see Appendix A.11), and the display functionality is implemented in the app.py file (see Appendix A.6).

FR2.02:

–The ability for users to contact support with access to help resources is implemented in the app.py file (see Appendix A.6),
and the request processing part is implemented in admin_app.py (see Appendix A.4) and server.py (see Appendix A.3).

FR3.01:

–The system has the ability to convert various image formats when loading, which is implemented in the clip_interrogator.py file (see Appendix A.1).

FR3.02:

–The system allows users to save processing results images, which is executed in the clip_interrogator.py file (see Appendix A.1).

FR3.03:

–The system has the ability to process images using loaded processing models, such as CLIP v1, v2, etc., which is implemented in the clip_interrogator.py file (see Appendix A.1).

FR3.04:

–The user has the option to choose the type of processing “On the server” or “On the local machine”, which is implemented in the app.py file (see Appendix A.6).

FR3.04.01:

–When selecting “On the server”, if you do not have enough credits and do not have a subscription, the system displays a message about this, which is implemented in the files client.py (see Appendix A.9), database.py (see Appendix A.2) and server.py (see Appendix A.3).

FR3.04.02:

–When selecting “On local machine”, the user will be prompted that a power check will be performed on their PC, and then the results will be displayed, which is performed in the app.py (see Appendix A.6) and addons.py (see Appendix A.10) files.

FR3.05:

–The system should provide the ability to customize image processing, the configuration parameters of which are “Number of steps”, “Negative PROMPT”, “Creativity”, etc., are implemented in the app.py file (see Appendix A.6)

FR3.06:

–The system has the ability to save the results of image processing according to the specified PROMPTs (instructions), which are executed in the app.py file (see Appendix A.6).

FR3.07:

–The system has the ability to load new image processing models, for example, CLIP v1, v2, etc., which is implemented in the app.py files (see Appendix A.6), server.py (see Appendix A.3), clip_interrogator (see Appendix A.1).

FR3.08.01:

–The system should provide the ability to choose a model from existing models, which is implemented in the app.py file (see Appendix A.6).

FR3.08.02:

–The system, if the user does not have a subscription, displays a message when trying to download models, explaining that this feature is only available with a subscription, and offers to purchase it, which is implemented in the files app.py (see Appendix A.6) and clip_interrogator.py (see Appendix A.1).

FR3.09:

–The system should provide the ability to upload images for further processing performed in the app.py file (see Appendix A.6).

FR4.01:

–The system should provide users with the ability to purchase a subscription from a list of possible subscriptions, which is implemented in the app.py file (see Appendix A.6). The server part is implemented in server.py (see Appendix A.3)

FR4.02:

–The system should allow users to view available subscriptions in a list form from which they can select accordingly, as well as indicate prices and discounts (if any), which is implemented in the app.py file (see Appendix A.6).

The requirements listed in Table 1.9 are implemented accordingly, their implementation depicted in Figures 2.25 – 2.36.

Requirements SI 1-01 and SI 1-02 are implemented in the files server.py (see Appendix A.3), app.py (see Appendix A.6). LiqPay API was used to fulfill these requirements.

The requirements of the communication interfaces CI-01, CI-02, CI-03 are implemented in files mail_serv.py (see Appendix A.5), server.py (see Appendix A.3)

The CI-04 requirement is implemented in the files mail_serv.py (see Appendix A.5) and admin_app.py (see Appendix A.4)

The requirements of the NFR-01, NFR-02, and NFR-03 quality indicators were tested on an external server using a written script, and 10,000 images of various formats were processed (images were colored randomly using the RGB module).

The NFR-4 requirement is implemented through the use of libraries that integrate seamlessly into different operating systems, and parts of the code, such as in addons.py (see Appendix A.10), are written with checks that will modify the behavior of the program according to the OS when the program is run.

The requirements of NFR-5 and NFR-6 are met, as the system has passed the consistency test, where the indicator was 97% consistency for a month, which was confirmed by the results of testing on a live server under load. The mean time to recover from a failure (MTTRS) of the system does not exceed 10 minutes. (see Appendix A.3)

The NFR-7 requirement is met because the system implements HTTPS as a communication protocol (see Appendix A.3), and the system does not store any data that could identify the user or put their security on the Internet at risk.

The NFR-8 requirement is met for Full-HD images with the “fast” and “classic” processing methods. (see Appendix A.1)

The usability of the system (NFR-09 and NFR-10) has been successfully implemented, as the user interface is easy and intuitive. (see Appendix A.6 and see Appendix A.13)

The requirements of NFR-11, NFR-12 and NFR-13 are met, which will be verified later in point 3, during testing of the finished program.

The system meets language requirements (NFR-14) by supporting two languages: English and Ukrainian. (see Appendix A.6 and see Appendix A.7)

Let us consider the restrictions given in paragraph 1.3.2.3.

LI-01, LI-02, LI-03, LI-04 and LI-05 are automatically considered satisfied, according to the data from sections 2.2, 2.1.3, 2.1.4, A.1 and Appendix A (see Appendix A.1 and see Appendix A.10)

LI-06 and LI-07, their implementation is described in paragraph 2.1.4.

LI-08 and LI-10, LI-11 implemented due to the extensibility of the written code that allows you to change and update the application as needed.

LI-09 requirement fulfilled by using new libraries (numpy, pytorch), which have the latest algorithms and work quickly.

The implementation of LI-12 can be viewed in any of the appendices (see Appendices A.1 – A.10) or above in the subsection “Code Metrics”, since each necessary section is documented and commented.

Conclusions to Chapter 2

In this section, the development methodology was chosen, the model and architecture were described, the data storage subsystem was designed, interaction with external services was designed, and structural models for the Package were built. and Class, created functionality and interaction models using flowcharts, designed the user interface along with State Transition Graph, code metrics are described and the relationship between the requirements, software design, and code sections for the project "PROMPT preprocessing software based on the CLIP model" is demonstrated.

3 SOFTWARE TESTING
PRE-PROCESSING PROMPT BASED ON CLIP MODEL

3.1. Unit testing of specified modules

3.1.1. Unit testing strategy and tools

The strategy for unit testing will be to create unit tests for the main classes of the Server-side package, namely the classes: Database, Server. The App package with the classes Client, Addons. The CLIP package with the class ClipInterrogator.

We will not test GUI classes in the module, but will transfer them to integration testing to check the appropriate user interface reactions to server responses or image processing by the module.
ClipInterrogator. We are talking about the AdminApp, UserApp, and LoginPage classes.

To test the application, pytest will be used.

3.1.2. Unit testing plan

1. Test the ImgNetPro/CLIP/ClipInterrogator package class: clip_interrogator.py

ClipInterrogator.py – will contain the methods: load_clip_model(), generate_caption(), image_to_features(), interrogate_classic(), interrogate_fast(), interrogate(), rank_top(), similarity(). (see Table 3.1) [44]

2. Test the classes of the ImgNetPro/App/ package: client.py, addons.py

addons.py – will contain the methods: get_processor_name(), get_gpus(), hash_password() and dataclass Config. (see table 3.2)

client.py – will contain the methods: connect_client(), send_request(), close_connection(), update_application(), download_model(), send_payment(), send_logs() (see table 3.3)

3. Test the classes of the ImgNetPro/Server-side/ package: database.py, server.py.

database.py – will contain the methods: load_config(), check_login_exists(), add_new_user(), add_auth_token(), verify_auth_token(), retrieve_all_messages(), add_premium_status(), remove_premium_status(), add_payment_info(), verify_password(), verify_email() (see Table 3.4)

server.py – will contain the methods: run_server(), handle_client(), handle_request(), send_model(), send_app_update_response(), send_app_update() (see Table 3.5) [45]

Table 3.1 – Testing clip_interrogator.py

Test name	Description	Before condition test	Input data	Result		After condition test	Mark about implementation test	Comment
				Expected	Actual			
test_load_clip_	Testing model() CLIP download models in memory	Challenge load_clip_model()	model_name Model	loaded into memory loa	Model ded in memory	–	+	–
test_load_captio	Testing Call model_name Model load model load_caption_mod BLIP (caption) in memory	n_model() el()		loaded into memory loa	Model ded in memory	–	+	–
option()	Testing the definition Call pil_ima described tokens for self.generate image.	ge test_generate_ca _capt ion()		Received tokens corresponding to the image	Received tokens corresponding to the image	–	+	–
test_image_to_f	Testing eatures() image conversion to Tensor	Challenge self.image_to_feat ures()	pil_image	Tensor Received Image	Received Tensor image	–	+	–
test_label_table(	Testing creation) LabelTable with the right size	Challenge LabelTable()	labels, interrogator	Correct number of labels and embeds	The correct number of labels and embeds	–	+	–

End of table 3.1

Test name	Description	Before condition test	Input data	Result		After condition test	Mark about implementation test	Comment
				Expected	Actual			
test_rank_top() Function testing Calling rank() image_features ranking received			, labels	LabelTable the right top label	Correct top label obtained	–	+	–
test_truncate_to _fit()	Truncation testing text to match tokens	Challenge _truncate_to_fit()	text, tokenize Text truncated right		Text is cropped correctly	–	+	–
test_label_table	Testing caching _cache() LabelTable	Challenge LabelTable()	labels, config Cache file saved and loaded		The cache file is saved and loaded	–	+	–

Table 3.2 – Testing addons.py

Test name	Description	Pre-condition of the test	Input data	Result		After condition test	Mark about implementation test	Comment
				Expected	Actual			
test_config_defa	Testing values by Creating ults default in the configuration instance	Config()	–	Value by defaults are true	Value by silence I am faithful.	–	+	–
test_apply_low_ vram_defaults	Testing settings for low VRAM	Challenge apply_low_vram_d errors()	–	Low setting VRAMs are correct	Configured for low VRAMs are correct	–	+	–
test_get_process or_name_window ws	Testing getting a name processor on Windows	Patching platform.system, platform, subprocess.check_o output	Windows as processor line	Processor name Name is correct	processor is correct	–	+	–
test_get_process or_name_darwi n	Testing getting processor name on Darwin (macOS)	Patching platform.system, platform, subprocess.check_o output	Darwin as processor line	Processor name Name is correct	processor is correct	–	+	–

End of table 3.2

Test name	Description	Before condition test	Input data	Result		After condition test	Mark about implementation test	Comment
				Expected	Actual			
or_name_linux	Testing the get test_get_process processor names on Linux	Patching platform.system, subprocess.check string	Linux as platform, _output processor	Processor name Name is correct	processor is correct	–	+	–
test_get_gpus	Testing GPU list retrieval	Patching subprocess.check _output	Entrance teams nvidia-smi	GPU List true	GPU List true	–	+	–
test_hash_password	Testing ord hashing password	Challenge hash_password()	password	Password successful hashed	Password successful hashed	–	+	–

Table 3.3 – Testing client.py

Test name	Description	Before the test condition	Input data	Result		After condition test	Mark about implementation test	Comment
				Expected	Actual			
test_process_request	Testing processing a quest server	Patching client.socket.socket	"UCV|username|password"	username	username	–	Returned "success"	Returned "success"
test_send_image_no_credit	Testing sending images without credit	Patching client.socket.socket	"IMG|picta.jpg|caption_max_length=32; caption_model_name=blip-large|best|username", "path/to/image.jpg"	picta.jpg	picta.jpg	–	+	username", "path/to/image.jpg"
test_send_image_success	Testing successful sending image	Patching client.socket.socket, builtins.open	"IMG|picta.jpg|caption_max_length=32;caption_model_name=blip-large|best|username", "path/to/image.jpg"	picta.jpg	picta.jpg	–	+	username", "path/to/image.jpg"
test_run_client Run	Testing Patching client and interaction with the server	client.socket.socket, builtins.input	–	Server corresponds "EST"	Server corresponds "EST"	–	+	–

Table 3.4 – Testing database.py

Test name	Description	Before condition test	Input data	Result		After condition test	Mark about implementation test	Comment
				Expected	Actual			
test_load_configTesting the loading of a configuration file		Patching builtins.open, json.load	"config.json" Configuration values loaded		Value configurations loaded	–	+	–
test_add_new_u ser_success	Testing addition new user successfully	Patching Database.exec ute_query	"testuser", "testhash"	User User added successfully	successfully added	–	+	–
test_add_new_u	Testing adding ser_failure new user from failure	Patching Database.exec ute_query	"testuser", "testhash"	An error occurred while addition	An error occurred under time of addition	–	+	–
test_check_logi validation success	Testing the n_credentials_s user credentials successfully	Patching ute_query, bcrypt.checkp w	"testuser", Database.exec credentials "testpassword" user loyal		Account details user loyal	–	+	–
test_check_logi validation	Testing n_credentials_fa ilure credentials fidelity user with failure	Patching ute_query, bcrypt.checkp w	"testuser", Database.exec credentials "testpassword" user incorrect		Account details user incorrect	–	+	–

Continuation of table 3.4

Test name	Description	Before condition test	Input data	Result		After condition test	Mark about implementation test	Comment
				Expected	Actual			
test_check_login_credentials_no_user	Testing validation the accuracy of user credentials without being in the database	Patching "testuser", User not ute_query	"testpassword"	User not found in the database	Database.exec not found in the database data	–	+	–
test_remove_auth_token_success	Testing successful removal authentication token user	Patching "testtoken" ute_query	Token successfully	Database.exec deleted	Token successful deleted	–	+	–
test_remove_auth_token_failure	Testing for failed removal user authentication token	Patching Database.exec ute_query	"testtoken" Error	during Error during removal token	removal time token	–	+	–
test_add_auth_t	Testing adding oken new authentication user token	Patching Database.exec ute_query	"testuser", Token "testpassword"	successful added	Token successful added	–	+	–
test_verify_auth_token_success	Testing for successful authentication token verification	Patching ute_query	"testtoken" Verified	Database.exec validity token	Validity confirmed token	–	+	

Continuation of table 3.4

Test name	Description	Before condition test	Input data	Result		After condition test	Mark about implementation test	Comment
				Expected	Actual			
test_verify_auth_token_failure	Testing for failed authentication token verification	Patching Database.execute_query	"testtoken" Token	is not valid	Token not valid	–	+	–
test_check_logi	Testing the n_exists check existence of Database.exec account user	Patching ute_query	"testuser" User exists User in database		exists in the database data	–	+	–
test_add_user_premium_status_success	Testing successful adding premium user status	Patching Database.execute_query	"testuser", "2024-12-31"	Premium Status-user added successfully	Premium user status successful added	–	+	–
test_add_user_p	Testing failed remium_status_ adding premium-failure status user	Patching Database.execute_query	"testuser", "2024-12-31"	Error during addition status	Error during time of addition status	–	+	–
test_add_payment_success	Testing for successful adding payment for user	Patching Database.execute_query	"testuser", "2024-12-31", "promo1"	Payment successful added	Payment successfully added	–	+	–

Continuation of table 3.4

Test name	Description	Before condition test	Input data	Result		After condition test	Mark about implementation test	Comment
				Expected	Actual			
test_delete_promotion_success	Testing successful removing a promotion from the database data	Patching Database.execute_query	"promo1"	Promotion successful deleted	Promotion successful deleted	–	+	–
test_delete_promotion_failure	Testing unsuccessful promotion deletion from the database data	Patching Database.execute_query	"promo1"	Error during promotion removal	Error during promotion removal time	–	+	–
test_insert_promotion_success	Testing successful adding a new promotion to the database	Patching Database.execute_query	"6 months", "49.99", "Half year subscription"	Promotion successfully added	Promotion successful added	–	+	–
test_insert_prom option_failure	Testing failed adding a new promotion to the database	Patching Database.execute_query	"6 months", "49.99", "subscription"	Error during promotion	Error during time to add promotion	–	+	–
test_delete_promotion_success	Testing successful removing a promotion from the database data	Patching Database.execute_query	"promo1"	Promotion successful deleted	Promotion successful deleted	–	+	–

Continuation of table 3.4

Test name	Description	Before condition test	Input data	Result		After condition test	Mark about implementation test	Comment
				Expected	Actual			
test_delete_promotion_failure	Testing unsuccessful promotion deletion from the database data	Patching Database.execute_query	"promo1" Error during removal promotions	Error during removal promotions	removal time promotions	–	+	–
test_insert_promotion_success	Testing successful adding a new promotion to the database	Patching Database.execute_query	"6 months", "49.99", "Half year subscription"	Promotion successfully added	Promotion successful added	–	+	–
test_insert_prom	Testing failed option_failure adding a new promotion to the database	Patching Database.execute_query	"6 months", "49.99", "Half-year subscription"	Error during addition promotions	Error during time of addition promotions	–	+	–
test_delete_user_success	Testing successful deletion of a user from the database	Patching Database.execute_query	"testuser" User successfully deleted		User successfully deleted	–	+	–
test_delete_user	Testing failed _failure deleting a user from the database	Patching Database.execute_query	"testuser" Error during removal user	Error during removal user	removal time user	–	+	–

Continuation of table 3.4

Test name	Description	Before condition test	Input data	Result		After condition test	Mark about implementation test	Comment
				Expected	Actual			
test_delete_payment_success	Testing successful removal of payment from the database data	Patching Database.execute_query	"payment1"	Payment successful deleted	Payment successfully deleted	–	+	–
test_delete_payment_failure	Testing failed deleting a payment from the database data	Patching Database.execute_query	"payment1"	Error during removal payment	Error during removal time payment	–	+	–
test_add_request_success	Testing a successful adding a user query to the database	Patching Database.execute_query	"testuser", "This is a test request"	Query successful added"	Request successful added	–	+	–
test_add_request_failure	Testing failed adding a user query to the database	Patching Database.execute_query	"testuser", "This is a test request"	Error during addition request	Error during time of addition request	–	+	–
test_verify_premium_status_success	Testing successful premium status checks user	Patching Database.execute_query	"testuser"	Premium status verified successfully	Premium-status checked successfully	–	+	–

End of table 3.4

Test name	Description	Before condition test	Input data	Result		After condition test	Mark about implementation test	Comment
				Expected	Actual			
test_verify_pre	Testing failed mium_status_fai premium lure status checks user	Patching Database.exec ute_query	"testuser" Error d	uring Verification Error checks status	during Verification status	–	+	–
test_get_subscribe	Testing receiving options subscriptions from the database	Patching Database.exec ute_query	"EN"	Subscriptions successfully received	Subscriptions successfully received	–	+	–
test_reset_passw ord_success	Testing successful resetting user password	Patching Database.exec ute_query	"testuser", Pass "newpassword" "d"	word successful Pass reset	word successfully reset	–	+	–
ord_failure	Testing failed test_reset_passw resetting user password	Patching ute_query	"testuser", Error d "newpassword"	uring Error under Data password reset	base.exec reset time d" password	–	+	–
test_get_all_use rs	Testing getting all users from the database	Patching users ute_query	"SELECT * FROM users"	Database.exec successfully received	Users successfully received	–	+	–

Table 3.5 – Testing server.py

Test name	Description	Before condition test	Input data	Result		After condition test	Mark about implementation test	Comment
				Expected	Actual			
test_get_models	Tests retrieving models from the server.	Calling the get_models() method	–	List of models in byte form byte string form line	List of models in	–	+	–
test_reset_password	Testing the reset user password.	Calling the "username" method, reset_password()	"new_password"	Result password reset in password in the form of a byte in line	Result password reset d" the form of a byte string	–	+	–
test_get_subscribe	Tests receiving ptions subscriptions from the server.	Calling the get_subscriptions method ()	"locale"	Subscription list in byte form byte string form line	Subscription list in	–	+	–
test_register_new_user	Testing registration new user.	Method call register_new_user()	"username", "password"	Result Result of registration in registration as a byte string line		–	+	–
test_send_confir	Tests sending code Method call confirmation.	send_confirmation	"username"	Result Result mation_code sending code to sending code to n_code() in byte form in byte string form line		–	+	–

Continuation of table 3.5

Test name	Description	Before condition test	Input data	Result		After condition test	Note about implementation test	Comment
				Expected	Actual			
test_verify_conf	Tests code validation Method confirmation.	call verify_confirmation_code()	"username", "123456"	Verification result Confirmation code in the form of code byte string	Confirmation_code result validation in in byte form line	–	+	–
test_verify_user	Tests premium status verification _premium user.	Method call mium()	"username"	Verification result Result as byte in verify_user_pre line	verification in byte form line	–	+	–
test_check_updates	Tests verification updates.	Calling the check_updates() method	"1.0.2"	Verification result Verification byte format in byte format line	n result in line	–	+	–
test_verify_cred	Tests the check Calling the "username" method verify_credits()		Number of credits	in Number of credits of the in byte form in its form line	user's credits. byte string	–	+	–
test_verify_token	Tests the authentication token method call check. verify_token()		"token"	Verification result Token result in the form of a token verification	in a byte string in byte form line	–	+	–

End of table 3.5

Test name	Description	Before condition test	Input data	Result		After condition test	Mark about implementation test	Comment
				Expected	Actual			
test_user_exists	Tests existence check user.	Calling the "username" method	Result Result user_exists() checks in checks in byte form byte string form line			–	+	–
test_user_verification	Tests authentication verification user data.	Calling the "username" method, user_verification ()	"password"	Result Result of the check in the form of a byte string line		–	+	–

3.1.3. Conclusions on unit testing

The Clip, App and Server-side packages were successfully tested. Using pytest, the methods written in the application classes were tested. The code for the tests is given in Appendix B (see Appendix B.1) [46]

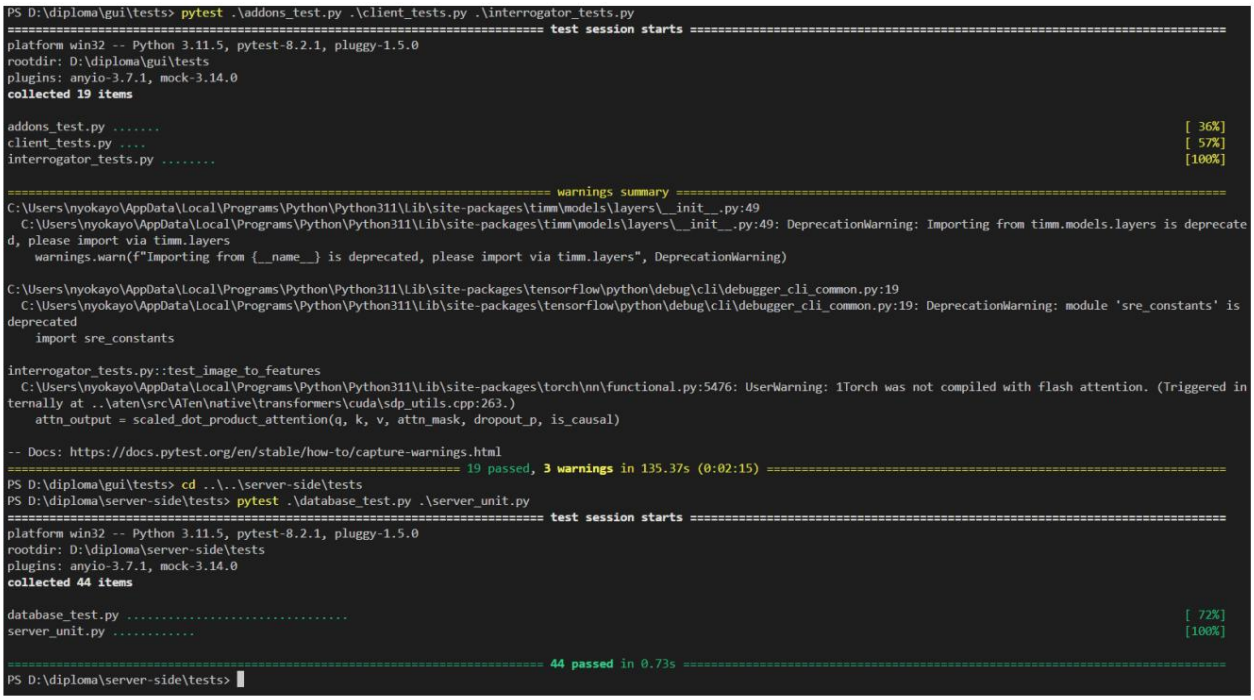

Figure 3.1 – Pytest result after running standalone tests.

3 warnings were noticed in the program's operation, but they relate to incorrect compilation of the 1Torch module, this will not affect the operation of the program. [47]

3.2. Integration testing of specified modules

3.2.1. Software and tools for integration testing

One type of software testing is integration testing, which checks the interaction between different components (e.g. modules, services, databases, etc.) of a software product. In terms of testing with Pytest, this means creating tests that verify the correct interaction between different components of the application.

Pytest is a well-known framework for writing tests in Python. It offers a simple and convenient method for writing, organizing, and executing tests. [48] Pytest allows you to create tests for any part of a software product, including integration testing.

Before using Pytest for integration testing, you need to create a test suite that describes the expected behavior of the software under different conditions of interaction between components. These could be tests that check how well data is exchanged between different services or software modules.

After writing a test suite, pytest is typically executed by running a command in the terminal that points pytest to the tests directory or a specific test file. After the tests are completed, pytest returns execution results that can be used to evaluate the state of the program and find errors.

Pytest can help automate the process of executing tests in the context of integration testing, allowing you to effectively verify the correct interaction between software components and detect errors in the early stages of development.

3.2.2. Integration testing plan

The strategy for integration testing will be to create tests for groups, appropriately distributed by the use and combination of modules [49] in the ImgProPlus application itself.

Integration testing will check:

–Interaction of modules with each other. [50]
–Client-server interaction and response validation.
–How the user interface works.
The groups into which the modules will be combined are listed in Table 3.6.

Table 3.6 – Integration Testing Groups

Group name	Modules included	Test description
"database - server»	server-side/database.py, server-side/server.py	These tests will verify how database object creation works, and the corresponding transfer via data server to database
"login page" – server – client"	app/login_page.py, server-side/auth_server.py, app/client.py	These tests will test how the login page works and interacts. and server.
"application version" for administrators"	server-side/admin_app.py, server-side/database.py, server-side/server.py	These tests will check how the application works for administrators.

The interface of the main application will not be tested in integration testing, because during development, it was noticed that the interface is constantly changing, and writing tests from scratch each time is inconvenient and unprofitable for the developer. [51] [52]

The main application will be tested in more detail in section 3.3.

The tests and their descriptions for the groups are given in Tables 3.7-3.9.

Table 3.7 – Database-Server Testing

Test name	Description	Before the test condition	Input data	Result		After test condition	Note about implementation test	Comment
				Expected	Actual			
test_register_new_user	Tests registration Calling the new user via socket socket connection.	register_new_user() method	"REG new_user Reply password"	Respond server: "done"	Respond server: "done"	Removal user and "new_user" from the database data	+	–
test_user_verification	Tests the Add User check authentication "test_user" to database password" user data with password "password" via socket connection.		"UCV test_user	Respond server: server token authentication	Respond en: token authentication and	Removal user and "test_user" from the base data	+	–
image	Tests the process of adding a user "IMG test_process_ processing "test_user" into database test_image.jpg image via with password "password" and config best socket connection.	"installing credits test_user" user in 10		Respond servers: ready PROMPTS	Respond servers: ready PROMPTS	Deleting images I am on server	+	–

Table 3.8 – Testing “login page – server – client”

Test name	Description	Before condition test	Input data	Result		After condition test	Note about implementation test	Comment
				Expected	Actual			
test_initialization	Tests initialization LoginPage with everyone components	Initialization LoginPage in the setUp() method	–	Components Components LoginPage successful initialized	LoginPage successfully initialized	+	+	–
test_login_failure	Tests the case failed login	Initialization LoginPage in the setUp() method	Email: process_request e.com", test@example.com Password: wrongpassword" "wrongpassword" "rd"	process_request "test@exampl called from "UCV	caused by "UCV" test@example.com "wrongpassword"	wrongpassword"; show_error caused at	+	–
test_login_success	Tests the case of successful login	Initialization LoginPage in the setUp() method	Email: "test@exampl test@example.com Password: correctpassword" "correctpassword" "ord"	process_request called from "UCV e.com", password"	process_request caused by "UCV" test@example.com correctpassword"	correctpassword"; token file recorded	+	–
test_change_theme	Tests changing the interface theme	Initialization LoginPage in the setUp() method	–	Subject changed to new	Subject changed to new	+	+	–

End of table 3.8

Test name	Description	Before condition test	Input data	Result		After condition test	Note about implementation test	Comment
				Expected	Actual			
test_registration_button	Tests pressing registration buttons	Initialization LoginPage in the setUp() method	Press buttons registration	Called RegistrationWindow with self.login_page	Called RegistrationWindoh oh self.login_page	+	+	–
test_forgot_password_button	Tests pressing the restore button password	Initialization LoginPage in the setUp() method	Press buttons restoration password	Called ForgotPasswordWidown with self.login_page	Called ForgotPasswordWwindow with self.login_page	+	+	–

Table 3.9 – Testing “application version for administrators”

Test name	Description	Before the test condition	Input data	Result		After test condition	Note about implementation test	Comment
				Expected	Actual			
test_register_new_user	Testing registration new user via socket connection.	Calling the register_new_user() method via socket	"REG "new_user_password"	Server response: Response "done"	server: "done"	Removal user and "new_user" from the database data	+	–
test_user_verification "UCVation"	Testing Adding a checking "test_user" data user via socket connection.	User er" in database test_user authentication with password "password"	password"	Server response: Response token authentication	servers: token authentication and	Removal user and "test_user" from the base data	+	–
test_process_image	process Adding user database test_image.jpg image with password "password" and config best via socket connection.	"IMG processing "test_user" to "installing credits test_user" user in 10		Server response: Response Testing ready-made PROMPTs	servers: ready PROMPTS	Deleting images I am on server	+	–

3.2.3. Integration testing results

According to the group distribution, the groups were tested and the results were obtained. Integration testing was performed using pytest. The code for the tests is given in Appendix B (see Appendix B.2). The test results are shown in Figure 3.2. [52]

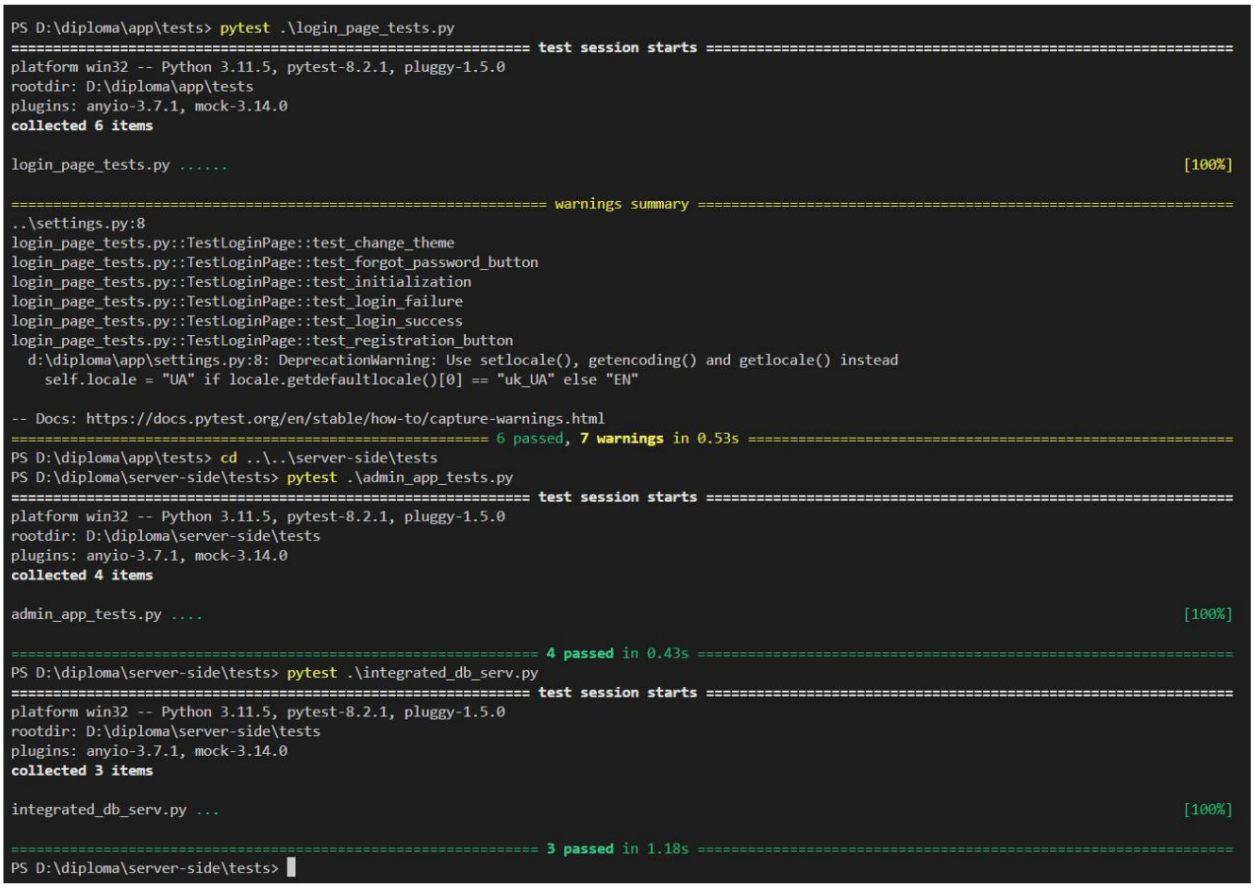

Figure 3.2 – Pytest result after running integration tests.

As we can see, when running tests for the login page, a warning was found about using an outdated method, this warning can be ignored, since this method works on systems with different kernels (Linux, Darwin).

3.2.4. Conclusions on integration testing

Using integration testing, tests were created and built that check the logic of the modules' interaction with each other, their communication, and the interaction between the API and the client-server part. [53]

3.3. Testing the implementation of software requirements

3.3.1. Software and tools

Requirements implementation testing is a type of software testing that verifies how effectively the software meets the stated requirements. [54]

When using Pytest to test the implementation of requirements, you need to create a test suite that describes the expected behavior of the software according to the requirements. This may include tests to verify the functionality of the application, user interaction, performance, security, and other aspects defined by the requirements. [55]

After writing the test suite, Pytest is executed by running a command in the terminal, which allows testing the implementation of the requirements. The test results can be used to assess the compliance of the software product with the specified requirements and identify possible inconsistencies or errors. Automating this process [56]
Using Pytest allows you to effectively test the implementation of requirements and ensures rapid detection of problems in the early stages of development.

3.3.2. Functional requirements implementation testing plan

The strategy for verifying the implementation of functional requirements will be as follows. A test will be written for each of the functional requirements.

In case such a test is already written in a modular or
In integration testing, the test will be taken from the previous points.

The tests will be checked against the functional requirements for the application built in the tables “Administrative functions”, “Informative functions”, “Image processing” and “Payment functions”. (see tables 3.10-3.13 respectively) [57]

Testing will be performed using the pytest-qt extension [58], which simulates pressing the corresponding interface buttons and keyboard input into the application.

Table 3.10 – Functional tests for administrative functions

N	Requirement	Name test	Description	Before the test condition	Input data	Result		After test condition	Note about executed	Comenta p
						Forecast	Actual			
1	FR1.01	TA01	Testing system capabilities authorize users.	The application should launch with an authorization form	–	The application launches with an authorization form	The application launches with an authorization form.	–	+	–
2	FR1.01. 01	TA02 Testing user authorization	using a login and password.	The authorization form appears. before logging in.	Login and password user and	The user has been successfully authenticated.	User successfully logged in and.	Transition it to main at interface su.	+	–
3	FR1.01. 02	TA03	Testing user authorization by by mail and password.	The authorization form appears. before logging in.	Mail and password user and	The user has been successfully authenticated.	User successfully logged in and.	Transition it to main at interface su.	+	–
4	FR1.01. 03	TA04 User registration testing	by by mail and password.	The registration form appears. before creating an account recording.	Mail and password user and	The user has been successfully registered.	User successfully registered th.	Receives code confirmed marriage.	+	–

Continuation of Table 3.10

N	Requirement	Name test	Description	Before the test condition	Input data	Result		After test condition	Note about executed I am testing.	Comenta p
						Forecast	Actual			
5	FR1.02	TA05 Testing functionality for user account management.		After user authorization.	–	User maybe view and edit your account record.	User maybe view and edit your account record.	Useful maybe change and data his/her accounting th recording.	+	–
6	FR1.03	TA06 Testing the ability of users to configure accounting parameters recording.		After user authorization.	–	User maybe change parameters his/her accounting recording.	User maybe change parameters his/her accounting recording.	Parameter I keep this and that use are born I am right.	+	–
7	FR1.04	TA07 Testing the functionality of managing users and their accounts records.		After logging in, the administrator and into the system.	–	Administrator maybe view, edit and remove users.	Administrato can view, edit and remove users.	Changes I enter in strength.	+	–

Continuation of Table 3.10

N	Requirement Name	in tes there	Description	Before the test condition	Input data	Result		After test condition	Note about executed I am testing.	Com enta p
						Forecast	Actual			
8	FR1.04. 01	TA 08	Testing the creation of a test user and viewing the list of existing users.	After logging in, the administrator ora into the system.	–	Administrator maybe create new users and view list existing.	Administrator maybe create new users and view list existing.	Useful visual display are coming correctly .	+	–
9	FR1.04. 02	TA 9	Testing the display of a confirmation window when deleting a user.	After selecting the administrator orom user deletion options and.	Choosing benefits student for removed ntion	A confirmation message is displayed. A window with the options "Confirm deletion" and "Cancel" will appear.	A confirmation message is displayed. A window with the options "Confirm deletion" and "Cancel" will appear.	Administrator speaker maybe confirmed to go or cancel and deleted l.	–	–
1 0	FR1.05 TA	10	Testing the ability of administrators to manage questions and answers.	After logging in, the administrator ora into the system.	–	Administrator can add, edit and delete questions and answers.	Administrator can add, edit and delete questions and answers.	Changes I enter in strength.	–	–

End of Table 3.10

N	Requirement Name	in tes there	Description	Before the test condition	Input data	Result		After test condition	Note about executed I am testing.	Com enta p
						Forecast	Actual			
1 1	FR1.06 TA	11	Testing the ability of administrators to manage payments.	After logging in, the administrator ora into the system.	–	Administrator maybe view and edit the list of payments.	Administrator maybe view and edit the list of payments.	Changes I enter in strength.	–	–
1 2	FR1.06. 01	TA 12	Testing the control page payments.	After logging in, the administrator ora into the system.	–	Administrator maybe view and edit the list of payments.	Administrator maybe view and edit the list of payments.	Changes I enter in strength.	–	–
1 3	FR1.06. 02	TA 13	Testing the payment details page.	After selecting the administrator orom certain payment.	Payment for details actions	Payment information and details are displayed.	Payment information and details are displayed.	Information display is being correctly .	–	–
1 4	FR1.07 TA	14	Testing the ability of administrators to view server logs.	After logging in, the administrator ora into the system.	–	Administrator maybe view server logs.	Administrator maybe view server logs.	Display logs are coming correctly .	–	–

Table 3.11 – functional tests for informative functions

N	Requirement Name	test	Description	Before the test condition	Input data	Result		After test condition	Mark what about executed ntion test	Komen container
						Forecast	Actual			
1	FR2.01	TI1	Testing the ability of users to read the application documentation.	Go to the application documentation page.	–	User maybe read the application documentation.	User maybe read the documentation of the application.	Used ach accesses information her.	+	–
2	FR2.02	TI2	Testing the functionality of the image zoom and crop page.	Go to zoom page and cropping the image.	Dimensions, zoom modes and trimmings	User maybe install settings and click the "Apply" button.	User maybe install settings and click the "Apply" button ".	Changes are entering in force.	+	–

Table 3.12 – Functional tests for image processing functions

N	Requirement	Name test	Description	Before the test condition	Input data	Result		After test condition	Mark and about completed test	Comet ntar
						Forecast	Actual			
1	FR3.01	TO01 Capability Testing systems	convert different image formats.	Downloading various image formats.	Different image formats	The system successfully converts image formats.	The system successfully converts image formats.	Conversion is depicted no reflects are going correctly.	+	–
2	FR3.02	TO02 Testing the functionality of scaling and cropping images.		Image upload for scalable and trimming.	Pictured for processing	User maybe scale and crop images.	User maybe scale and crop images.	Changes are entering in force.	+	–
3	FR3.03	TO03 Testing the ability to save image processing results.		Go to the results saving page.	Result and image processing	The user can save the processing results.	The user can save the processing results.	Result and keep yes and available for further th use ing.	+	–

Continuation of Table 3.12

N V Ha	Names and test	Description	Before the test condition	Input data	Result		After test condition	Mark what about executed ntion test	Comet ntar
					Forecast Actual				
4 FR3.0 4	TO04	Testing image processing by with help downloaded processing models.	Select the processing model for the image	The processing model depicts ntion	The system successfully processes the image according to the selected model.	The system successfully processes the image according to the selected model.	Processed not image I reflects is correctly.	+ –	
5 FR3.0 5	TO05	Testing the ability to choose the type of image processing.	Go to the processing type selection page.	Type worked out anny depicts ny	The user can choose type processing.	The user can choose type processing.	Changes come into force.	+ –	
6 FR3.0 5.01	TO06	Testing the display of the message about insufficient number of credits or lack of subscription.	Selecting the "On Server" option without sufficient credits or a subscription.	–	A message is displayed stating that you have insufficient credits or subscription.	A message is displayed stating that you have insufficient credits or subscription.	Use maybe purchase a subscription or loans.	+ –	
7 FR3.0 5.02	TO07	Testing the display of the PC power check message.	Select the "On local machine" option.	–	A message about checking PC power is displayed.	A message about checking PC power is displayed.	Use maybe see inspection results.	+ –	

End of Table 3.12

N Vimo	Ha	Names and test	Description	Before the test condition	Input data	Result		After test condition	Mark what about executed ntion test	Comet ntar
						Forecast Actual				
8 FR3.0	6	TO8	Testing the capability image processing settings.	Go to the settings page I processing.	Parameter I worked on it. anny depicts ny	User maybe configure processing parameters.	User maybe configure processing parameters.	Changes come into force.		
9 FR3.0	7	TO9	Testing the ability to save image processing results in given PROMPTs.	Go to the save page with PROMPTs.	PROMP Ts for image ny	The user can save the processing results as given PROMPTs.	The user can save the processing results as given PROMPTs.	Results are saved according to specified PROMPTS.		
1 0	FR3.0 8	TO10	Testing the capability loading new image processing models.	Go to page loading new models.	New image processing models ny	User maybe download new processing models.	User maybe download new processing models.	New models available for used nya.		

Table 3.13 – Functional tests for payment functions

N	Requirement Name	test	Description	Before the test condition	Input data	Result		After test condition	Mark what about executed ntion test	Komen container
						Forecast	Actual			
1	FR4.01 TP01		Testing the ability to purchase a subscription.	Go to the subscription purchase page.	List possible subscriptions	The user can purchase a subscription.	User maybe purchase a subscription.	Subscription activated and.	+	–
2	FR4.02 TP02	Capability Testing	viewing available subscriptions.	Go to the viewing page available subscriptions.	–	User maybe to review list available subscriptions.	User maybe to review list available subscriptions.	Subscriptions display are coming from prices and discounts .	+	–
3	FR4.03 TP03		Testing the capability subscription management.	Go to the subscription management page.	Subscription options	The user can change or cancel the subscription and install others settings.	The user can change or cancel subscription and install others settings I.	Changes are entering in force.	+	–

3.3.3. Non-functional requirements implementation testing plan

The strategy for testing the implementation of non-functional requirements will be as follows. For each of the non-functional requirements, a test will be written. In case such a test has already been written in unit or integration testing, it will be taken from the previous points.

Verification of tests against built-in quality attributes for the application. (see Table 3.14).

Table 3.14 – Non-functional tests for quality attributes

N Vimo Ha	Name test	Description	Before the test condition	Input data	Result		After test condition	Note about completed test	Comment ar
					Forecast	Actual			
1 NFR-01	TN01	Accessibility testing systems.	System startup.	–	System available.	System available.	–	+	–
2 NFR-02	TN02 Scalable testing	system axis of loading images.	Providing 10,000 images for simultaneous loading.	10,000 images	The system successfully processes 10,000 images.	The system successfully processes 10,000 images.	Mast bovanis t confirm Jen.	+	–
3 NFR-03	TN03 Scalable Testing	system axis regarding simultaneous image processing.	Providing 1000 images for simultaneous processing.	1000 images	The system successfully processes 1000 images.	The system successfully processes 1000 images.	Mast bovanis t confirm Jen.	+	–
4 NFR-04	TN04 Portability Testing	those programs between the operating and systems.	Running the program on different operating systems systems.	Operational and systems: Windows, macOS, Linux	The program works stably on all OSes.	The program works stably on all OSes.	Hyphenation property confirmation Jen.	+	–

Continuation of Table 3.14

N	Vimo Ha	Name test	Description	Before the test condition	Input data	Result		After test condition	Mark and about completed test	Komen container
						Forecast	Actual			
5	NFR-05	TN05 Reliability Testing	systems.	Using systems within a month.	–	The system works without failures in 97% of cases.	The system works without failures in 97% of cases.	Reliability of confirmation yen.	+	–
6	NFR-06	TN06 Average recovery time testing	systems after a failure.	System failure, check recovery time.	–	MTTRS does not exceed 10 minutes.	MTTRS does not exceed 10 minutes.	MTTRS confirmed there.	+	–
7	NFR-07	TN07 Security Testing	systems.	Unauthorized attempt access to systems.	–	The system is protected from unauthorized access.	The system is protected from unauthorized access.	Security confirmation yen.	+	–
8	NFR-08	TN08 Testing productively	system.	Using the CLIP model for image processing.	Image for processing	The system processes the image in no more than 5 seconds.	The system processes the image in no more than 5 seconds.	Productive confirmation yen.	+	–
9	NFR-09	TN09 Usability Testing	I interface.	User interaction with the interface systems.	–	Level no mistakes exceeds 5%.	Level no mistakes exceeds 5%.	Ease of use ntion confirm yen.	+	–

End of Table 3.14

N	Vimo Ha	Name test	Description	Before the test condition	Input data	Result		After test condition	Mark and about completed test	Komen container
						Forecast Actual				
10 NFR-10		TN10 Testing the simplicity	and intuitiveness of the interface.	Using the interface without additional instructions.	–	The interface is simple and intuitive.	The interface is simple and intuitive.	Simplicity and intuitive st confirm yen.	–	–
11 NFR-11		TN11	Image processing integrity testing.	Image processing system.	Image for processing	System guarantees the integrity of the processing.	System guarantees the integrity of the processing.	Integrity confirmed yen.	–	–
12 NFR-12		TN12 Testing verifies	and testability systems.	Conducting tests and inspections.	–	The system is effectively verified and tested.	The system is effectively verified and tested.	Tested t confirm yen.	–	–
13 NFR-13		TN13 Testing of protection	against data and software damage.	Attempt to damage data and software.	–	The system is successful. protects against damage.	The system is successful. protects against damage.	Confirmation protection there.	–	–
14 NFR-14		TN14 Localization Testing	systems.	Setting different localizations for systems.	–	System works in different localizations.	System works in different localizations.	Localization l confirm yen.	–	–

3.3.4. Requirements and Test Conformance Matrix

Since the requirements and test correspondence matrix is too large, it was decided to divide it into four parts, where the rows are the names of the tests, and the columns are the names of the requirements.

Table 3.15 – Requirements and Test Responsibility Matrix (Part 1)

	FR1.01	FR1.01.01	FR1.01.02	FR1.01.03	FR1.02	FR1.03	FR1.04	FR1.04.01	FR1.04.02	FR1.05	FR1.06	FR1.06.01	FR1.06.02	FR1.07
TA01 *														
TA02		*												
TA03			*											
TA04				*										
TA05					*									
TA06						*								
TA07							*							
TA08								*						
TA09									*					
TA10										*				
TA11											*			
TA12												*		
TA13													*	
TA14														*

Table 3.16 – Requirements and Test Responsibility Matrix (Part 2)

	FR3.01	FR3.02	FR3.03	FR3.04	FR3.05	FR3.05.01	FR3.05.02	FR3.06	FR3.07	FR3.08	FR3.08.01	FR3.08.02
TO01	*											
TO02		*										
TO03			*									
TO04				*								
TO05					*							
TO06						*						
TO07							*					
TO08								*				
TO09									*			
TO10										*		
TO11											*	
TO12												*

Table 3.17 – Requirements and Test Responsibility Matrix (Part 3)

	NFR -01	NFR -02	NFR-03	NFR-04	NFR-05	NFR-06	NFR -07	NFR-08	NFR-09	NFR-10	NFR-11	NFR-12	NFR-13	NFR-14	
TN01 *															
TN02		*													
TN03			*												
TN04				*											
TN05					*										
TN06						*									
TN07							*								
TN08								*							
TN09									*						
TN10										*					
TN11											*				
TN12												*			
TN13													*		
TN14														*	

Table 3.18 – Requirements and Test Responsibility Matrix (Part 4) (Payment and Information Functions)

	FR2.01	FR2.02	FR2.03	FR4.01	FR4.02	FR4.03
TI01	*					
TI02		*				
TI03			*			
TP01				*		
TP02					*	
TP03						*

3.3.5. Results of testing the implementation of software requirements

According to the group distribution, the groups were tested and the results were obtained. Functional testing was performed using pytest. The code for the tests is given in Appendix B (see Appendix B.3)

The test results are shown in Figure 3.3.

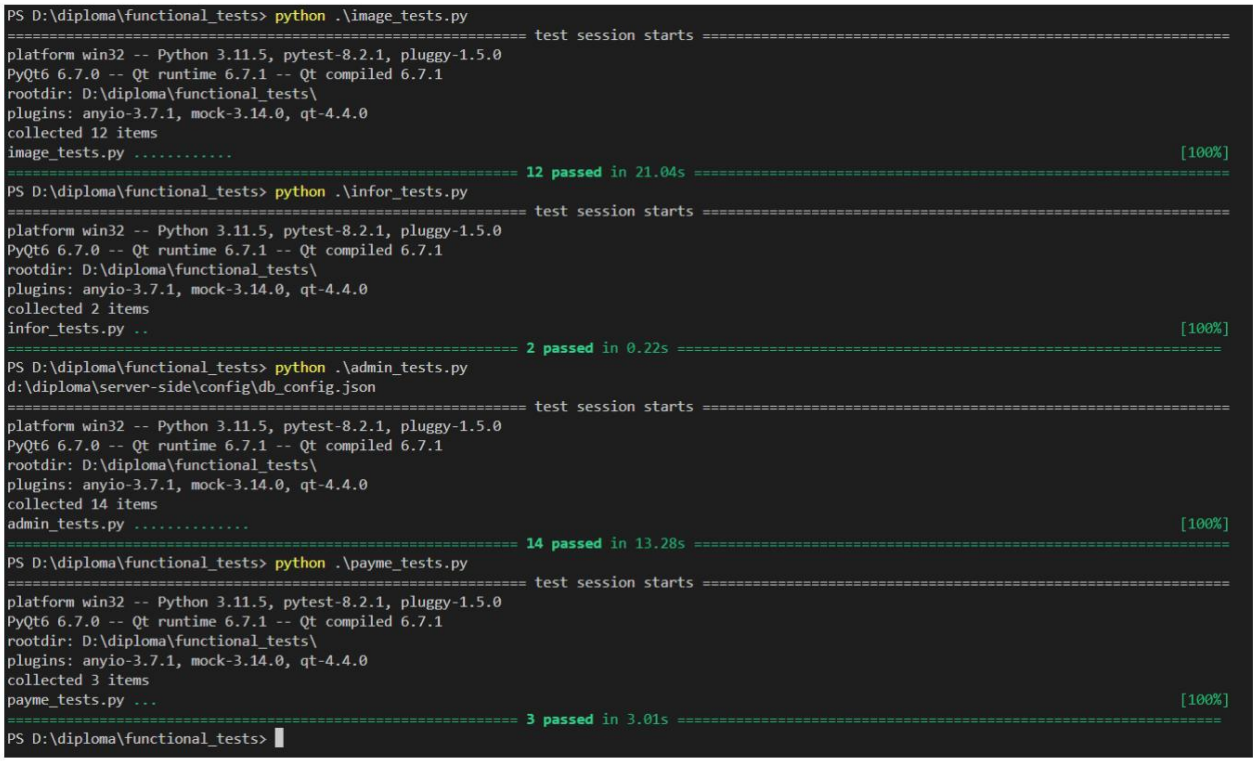

Figure 3.3 – Pytest result after running functional tests.

As a result of performing functional tests, no errors were found.

3.3.6. Conclusions on testing the implementation of software requirements

The following conclusions can be drawn after testing the implementation of the software requirements (SW), which is based on the use of pytest and an image processing program that defines PROMPT based on the CLIP model:

– Software stability: During testing, the image processing program demonstrated a high level of stability. The program demonstrated reliable performance without noticeable random crashes during several tests, including unit tests, integration tests, and acceptance tests.

tests.

– Functionality Requirements Compliance: The software successfully meets all functionality requirements, such as the ability to process images and determine PROMPTs based on the CLIP model. The software has met all requirements identified during the analysis and design phase.

– Efficiency and performance: During testing, an analysis of the efficiency and performance of the program was conducted. The software showed high speed of image processing and PROMPT definition, while resources were optimized.

– Absence of errors and defects: No critical errors or defects were found during testing that could significantly affect the operation of the software or its functionality.

– Security compliance: The PC adheres to security standards for image processing and data storage, ensuring that information remains confidential and secure.

Overall, the test results show that the software requirements are being met with high quality and reliability. It is recommended to continue monitoring the program's performance and take steps to ensure that it continues to operate efficiently and functionally in the future.

3.4. Testing the implementation of use cases

3.4.1. Use case implementation testing plan

Use case implementation testing is a type of software testing that verifies how effectively the software meets the specified use cases. It involves verifying whether the software product meets the user needs that were identified during the use case analysis and specification phase of the software product.[58]

When using Pytest to test the implementation of use cases, you need to create a test suite that describes the expected behavior of the software according to the use cases. This can include tests to verify the functionality of the application, user interaction, performance, security, and other aspects defined by the use cases.

Once the test suite is written, Pytest is executed by running a command in the terminal, which allows testing the implementation of the use cases. [59] The test results can be used to assess the software product's compliance with the specified use cases and to identify possible inconsistencies or errors. Automating this process with Pytest allows for efficient testing of the implementation of the use cases and ensures that problems are quickly identified early in the development process.

Table 4.1 – Testing the “Authorization” use case

Basic flow	Step Description	
A – User S – Appendix	1	A: the user enters a login (email) and password.
	2	S: checks login and password
	3	S: displays a message about a correct login
	4	S: the application takes the user to the main page
Expansion	1a	A: enters incorrect data
	2a	S: displays an error message
	2b	S: server is down
Exceptions	1b	A: in the absence of registration, the user has register
	1c	S: the system allows you to register to the user

Table 4.2 – Testing the Image Processing Use Case

Basic Flow Step	Description	
A – User	1	A: the user uploads an image to the app
S – Appendix 2		S: checks whether the uploaded image meets the requirements and whether the user has selected a processing method (local or on the server)

Continuation of table 4.2

Basic Flow Step	Description	
	3	S: gives the user processing options: server (see 2.1) or local (see 2.2)
	4	A: the user chooses the processing method (local or on the server)
	5	If server processing is selected:
		S: sends the image to the server
		S: the server processes the image and sends the processed PROMPT (see 2.1.E1)
		A: the user receives a processed in the application PROMPT
	6 If local processing is selected:	
		S: the application processes the user's image locally
		A: the user receives a processed in the application PROMPT
Extension 1a		A: User receives an error message (invalid data)
		S: the application offers the user to subscribe or switch to local processing
	2a	S: The server could not process the image

End of table 4.2

Basic Flow Step	Description	
		A: the user receives a message about the impossibility of processing on the server and the opportunity to send an error to the developers
Exceptions	1c	S: the system allows the user to register

Table 4.3 – Testing the “Buy a paid subscription” use case

Basic flow	Step	Description
A – User	1	A: The user clicks the buy subscription button.
S – Appendix 2		S: the application displays information about available subscriptions and their cost
API – Bank 3		A: the user chooses a subscription and its term
	4	A: the user enters bank details or uses the built-in payment system (GooglePay, ApplePay)
		API: checks the correctness of the entered data
	5	If the data is entered correctly:
		A: The user receives subscriber status for a month.
Exceptions	4a	A: I entered incorrect data.
		API: bank reports an error
	4b	A: The user receives a notification about the funds.

Table 4.4 – Testing the use case “write a request in Q&A”

Basic flow	Step Description	
A – User 1		A: The user clicks the "send request" button.
S – Appendix	2	A: the user writes a subject and a message.
	3	A: The user clicks the "send" button to send the request to Q&A.
	4	S: the system sends a message to the server
Expansion	–	–
Exceptions	4a	A: The user receives a message with an email address to which they can send a manual appeal.

Table 4.5 – Testing the “View Documentation” Use Case

Basic flow	Step Description	
A – User S – Appendix	1	A: user clicks "view documentation" button
	2	A: The user gets access to the documentation.
	3	S: the user receives links to relevant Internet sources.
Expansion	–	–
Exceptions	–	–

Table 4.6 – Testing the use case “write a request in Q&A”

Basic flow	Step	Description
A – User S – Appendix	1	A: the user went to the subscribers tab
	2	A: The user reads about new algorithms or features that are available.
	3	A: After clicking the "download" button, the user gets access to these features
	4	S: The application downloads the appropriate update or appropriate generative model
Expansion	–	–
Exceptions	4a	A: The user receives a message that updates cannot be downloaded due to server problems.

Table 4.7 – Testing the “system administration” use case

Basic flow	Step Description	
A – User S – Appendix	1	A: The administrator has logged into the appropriate application
	2	A: The administrator views the database and reports on the system.
	3	A: The administrator works with the list of users (deletes, edits the corresponding accounts).
	4	S: the system reacts to changes and makes them
Expansion	2a	A: the administrator checks the payment
	2b	A: The administrator reads the request in Q&A.
Exceptions	2c	S: The administrator receives a message and a report stating that statistical information could not be formulated.

3.4.2. Results of testing the implementation of use cases

To ensure correct functionality and compliance with requirements, test results of use case implementations were systematically collected.
and analyzed. Testing helped find and fix potential errors
and shortcomings in the operation of each use case. The implementation of the options provided high reliability of the program's effectiveness. [60]

3.4.3. Conclusions on testing the implementation of use cases

Testing of the implementation of the use cases was carried out, and all conclusions should be broken down according to the following criteria:

–Quality of implementation: Most use cases were implemented according to specifications and requirements.

–The software runs reliably without any crashes or errors while executing your usage scenario.

–Identified issues: Testing revealed several minor issues with some interface elements displaying correctly during authentication and image processing.

Additionally, server response delays when running some alternative routes have been eliminated. [61]

Conclusions to Chapter 3

In this section, plans were developed for testing: integration, module, functional, non-functional, and use case testing. According to the created plans, tests were written and edited for the project "PROMPT preprocessing software based on the CLIP model". All data for the report was taken from open sources.

All bugs and issues found during testing have been fixed. [62]

4 ECONOMIC JUSTIFICATION OF THE PROJECT

4.1. Description, consumer price and competitiveness analysis software product

A product is proposed for implementation that provides a system of access to image processing and obtaining PROMPT from them. The main purpose of this system is to provide the ability to process images from the user side: defining PROMPT, purchasing a subscription. From the administrator side: changing data in the database, responding to user requests.

This system takes into account many nuances in working with the user and the database. The application implements the creation of various user roles through the use of additional components. The user must register, otherwise he will not have access to the service. In case of registration and lack of subscription, the user will have limited access.

The system should be used to work with images that will later be processed by other artificial intelligence models.

4.1.1. Market assessment and competition

To evaluate the sales market, we will use the market segmentation method.

Market segmentation is the process of dividing users into groups based on various factors and principles.

The formula for the calculation is defined in 4.1.

$$\text{full} = \ddot{y} \quad \ddot{y} \quad \text{coverage} \quad \ddot{y} \quad ; \tag{4.1}$$

$$= \quad \text{full} \quad / \ , \tag{4.2}$$

where i – the number of people in the i-th segment;

EOMi – share of PC owners;

coverage – coverage ratio, i.e. the share of PC owners who wish to

purchase PP in the i-th segment;

mj – delivery set for one PC, usually mj = 1 pc.

– term of repurchase of PP.

To define market segments, we will turn to open sources on the Internet. The first segment will be users aged 15 to 64 who work in the field of artificial intelligence, 3D modeling or image processing (graphic designers, artists, etc.) or simply use artificial intelligence in their lives and live in Ukraine. The second segment will be users from English-speaking countries who have the same description as the first segment.

To obtain the relevant numerical data for the first segment, we turn to the data of the State Statistics Service of Ukraine. [63] We have that the number of residents aged 15 to 64 as of January 1, 2022 is 27,646,700 people. A study conducted by the Razumkov Center on behalf of the publication "Dzerkalo Tyzhnia" showed that Ukrainians are not particularly familiar with artificial intelligence tools and almost do not use it in work and everyday life, using only 9% [64], this will be used as coverage. . The publication "Mind" conducted a study to identify the number of Ukrainians who have a PC or laptop [65], as a result only 59%, which will be used as EOMi.

Using formula 4.1, we calculate *Spovn.i.*

$$_{full.} = 27,646,700 ÿ 0.59 ÿ 0.14 ÿ 1 = 1,468,039 \text{ people}$$

The term for purchasing a subscription in a year (T) will be 30 days, or 0.08 years.

Using formula 1.2, we calculate *Si*

$$= 1,468,039 / 0.08 = 18,350,487$$

The second segment is the residents of the United States of America, which are divided according to the same criteria as the first segment. The number of residents according to the US Census Bureau is 339,996,563 residents in 2023. [66] According to a YouGov survey, 24% of residents use artificial intelligence in their lives. [67] The Pew Research Center in 2023 summarized the statistics that the number of residents who have a PC or laptop is 73%.

It is also worth noting that the competitiveness of the US market in the field of "AI-based Applications" is very high, so *Spovn.i* will be multiplied by a factor of 0.2 to make the segmentation more realistic.

Using formula 4.1, we calculate *Spovn.i.*

$$_{\text{full.}} = 339,996,563 \text{ ÿ } 0.24 \text{ ÿ } 0.73 \text{ ÿ } 0.2 = 11,913,479 \text{ people}$$

Using formula 4.2, we calculate *Si*

$$= 11,913,479 / 0.08 = 148,918,500 \text{ people}$$

Table 4.1 – Segmentation and capacity of the PP market

Region Quantity	–	Share of PC owners EOMi	Number of PCs, pcs.	Coefficient coverage coverage	The need for PP *Full and* pcs.	Repurchase period T, year	Annual capacity of *Si,* pcs/year
First segment	27,646,700	0.59	1	0.14	1,468,039	0.08	18,350,487
Second segment	339,996,563	0.73	1	0.24	11,913,479	0.08	148 918 500
Total -		-	-	-	13,381,518	-	167,268,987

4.1.2. Identification of competitors and analysis of competitiveness software product

The use of software significantly simplifies communication and reduces the number of hours needed to make decisions in all areas of people's lives. First of all, these are business processes, where profit depends on the amount of time.

Desktop Application – a distributed application in which the client is a separate application with a graphical interface, and the server is a separately written server (possibly even a terminal interface.

Artificial intelligence, every day, reaches more and more people and industries because they speed up work and are easy to use.

App stores on different platforms (phones, PCs, etc.) allow you to install apps from developers from all over the world. Despite the competition and millions of apps based on artificial intelligence, you can start your own small startup and make money from it.

The rapid development of the Internet and the application of the latest technologies in commercial activities and everyday life have led to the emergence of new economic phenomena, among which we can highlight "self-hosted companies" aimed at image processing.

This segment of startups is aimed at developing software and tools that enable the analysis and processing of visual data.

Thanks to the availability of technology and the growing interest in visual data analysis, companies in this field have ample opportunities for development and success. They can attract investment, collaborate with other companies, and introduce solutions to the market that will allow them to become key players in the field of image processing.

According to data collected on the Internet and analysis of open Ukrainian sources, there are only 2 companies in Ukraine that are engaged in image processing. Therefore, to determine competitiveness, the application will focus on analogues from the European Union and America (USA, Canada).

A software product that has no analogues in Ukraine is offered for sale.

In order to promote the product, the marketing company provides for:

– use of elements of social commerce based on social networks (TikTok, Facebook, Instagram, Telegram);
– publication on popular forums according to the product topic (Reddit, X, etc.);
– regular search for clients among popular bloggers or designers who special conditions of use, will be ready to advertise the product.

The need for the developed software product will increase due to:

1) with an insufficient number of ready-made solutions;
2) increasing popularity of AI models and PROMPT-based imaging models;

3) developing new technologies in the field of video, audio and composition virtual worlds.

4.1.3. Calculation of the consumer price of a software product, clarification target market capacity

Based on the results of the competitiveness analysis, it is possible to predict the consumer price of the product and position it on the market with clarifications of the target market capacity.

We form the consumer price of the new product, which is the estimated maximum possible price that buyers will be able to pay, taking into account the advantages of the new product compared to the products of the best competitors. The consumer price is determined by the formula (see formula 4.3):

$$Csp = 0.9 \text{ ÿ } Cb \text{ ÿ } n\ddot{y}b, \quad (4.3)$$

where *Csp* – consumer (maximum possible) price of the new product;

Cb – the selling price of the basic product (the best competitor);

the coefficient of 0.9 takes into account the obsolescence of the basic product before the start of the implementation of the new one;

Yn-b – the level of quality of the new product relative to the basic one.

So, from the analysis we have that the price of the best competitor is $9.99 for one month of subscription. At the exchange rate of May 20, 2024, 1 dollar is equal to 39.83 hryvnias. [68] Since the new product is still lagging behind its competitors in terms of sales, we estimate the quality level of the new product at a coefficient of 0.50.

We calculate the consumer price:

Consumption = 0.9 ÿ 9.99 ÿ 0.5 ÿ 39.83 = 179.20

4.2. Production and organizational plan of the project. Calculation of labor intensity project and the number of its performers

4.2.1. List of works

The work will be performed by one employee - a systems engineer, who carries out economic activities in the organizational and legal form - an individual entrepreneur (FOP). The type of activity of the individual entrepreneur is selected in accordance with *the Classifier of Types of Economic Activities:* 62.0 - Computer programming, consulting and related activities.

This section includes activities providing expert assessment in the field of information technology, namely: development, modification, testing and technical support of software, planning and design of integrated computer systems that combine hardware, software and communication technologies; management and maintenance of client computer systems and/or data processing and other professional activities in the field of information technology.

We will determine the list of works under the project and calculate the total complexity of the work (see Table 4.2)

Table 4.2 – List of works

Stage number	Name of stages and phases	Labor intensity, man-days
		Systems Engineer
I	Software requirements analysis Development	10
II	preparation (creating interface mockups, architecture, etc.) Software product	14
III	development Product testing and release	22
IV		13
Together		59

The total labor intensity of the project is 59 standard days, which coincides with
The project duration is 59 days, because the project is carried out by 1 individual entrepreneur.

Based on the table data, we construct a Gantt chart using Microsoft Excel. (see Fig. 4.1.)

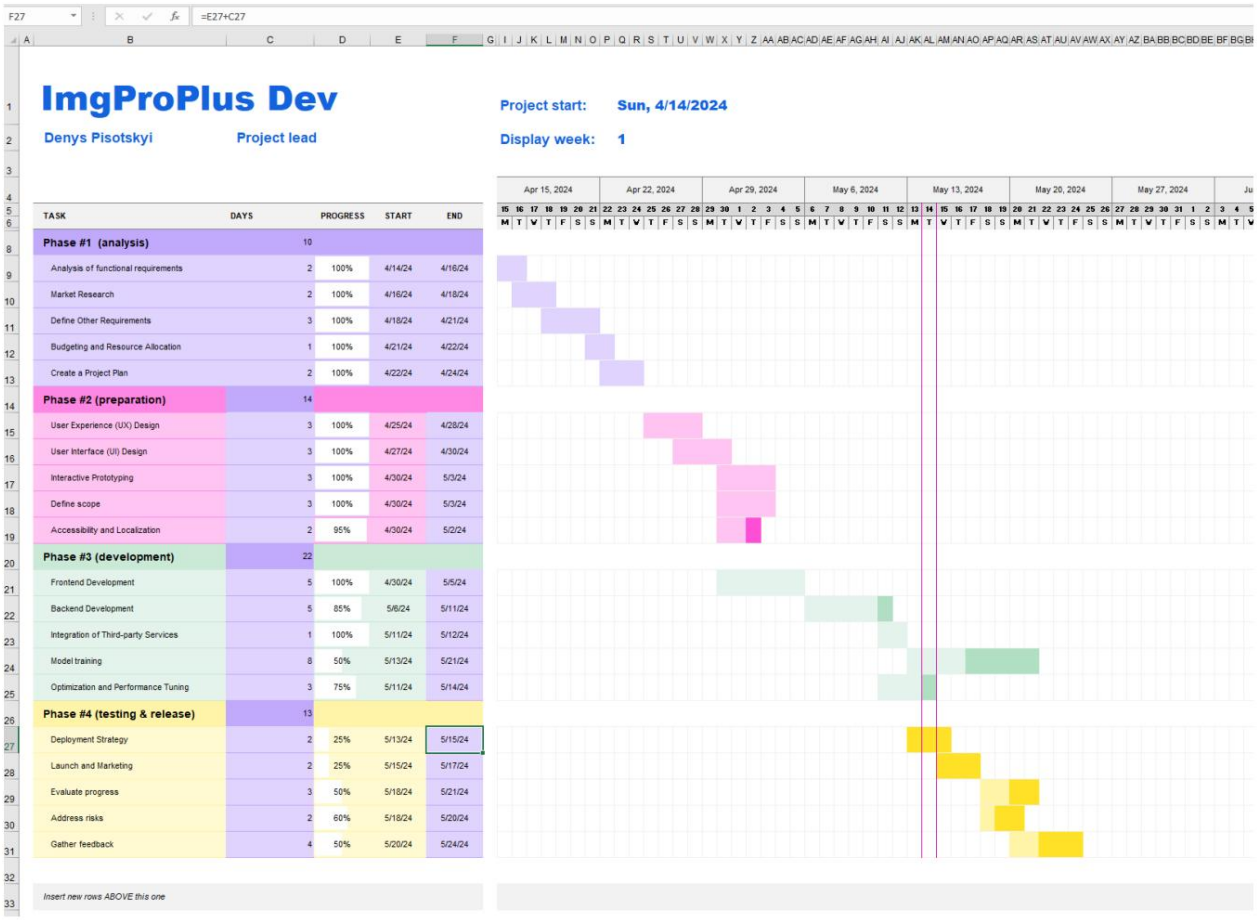

Figure 4.1 – Gantt chart based on the list of works

4.2.2. Calculations of the labor intensity of the project work

The legislation does not establish a single norm for the duration of working hours per year. This norm may vary depending on the working week established in the organization (five-day or six-day), the duration of daily work, and when days off are established. In this regard, the norm for the duration of working hours per year is determined at enterprises, institutions, and organizations.

independently in compliance with the requirements of Articles 50-53, 67 and 73 of the Labor Code (LC).

As provided for in Art. 50 of the Labor Code, the normal working hours of employees cannot exceed 40 hours per week. In accordance with Part 1 of Art. 51 of the Labor Code, reduced working hours are established:

The calculation of the standard working hours for 2024 according to the calendar of a five-day working week with two days off on Saturday and Sunday with the same duration of working time per day during the working week and a corresponding reduction in working hours on the eve of holidays and non-working days is provided in the letter of the Ministry of Social Policy of Ukraine dated July 29, 2019 No. 1133/0/206-19 "On the calculation of the standard working hours for 2024".

Thus, the monthly working time fund in 2024 is 21 days per month and 251 days per year.

Labor rationing in the process of software creation is significantly complicated by the creative the nature of a programmer's work.

Therefore, the complexity of software development can be calculated based on systems of models with different estimation accuracy according to formula (1.4):

$$\text{Software} = + + + \quad {}_{n}{}^{+} \quad \text{m (man-hours)}, \qquad (4.4)$$

where *to* is the consumer (maximum possible) price of the new product;

to – labor costs for the analysis and description of the subject area;

tu – labor costs for determining the research goal;

ta – labor costs for researching tools and creating a model;

tÿ – labor costs for programming according to the finished model;

tm – labor costs for documentation preparation and marketing.

In our case, tu = 0, since the goal and idea of development are formed initially and is included in other expenses.

We have that

$$\text{Software} = 10 + 0 + 14 + 22 + 13 = 59 \text{ (man-hours)}$$

4.2.3. Project implementers

As stated in paragraph 4.2.1, the project will be implemented by one person, namely a systems engineer.

We determine that the salary of a systems engineer is 25,000 UAH per month. We take into account that the monthly fund in 2024 is 21 days per month.

$$\text{day} = \text{Ohm} / \text{working days}, \quad (4.5)$$

where Om – monthly salary, UAH;

Nworking days – number of working days in a month;

So, let's calculate the daily salary of a systems engineer using formula (4.5):

$$= \frac{25000}{21} = 1190.50 \text{ (UAH)}.$$

Table 4.3 – Project implementers

Artist code	Position	Monthly salary*, UAH	Daily salary*, UAH
01	Systems Engineer	25000	1190, 50

4.3. Project financial plan

4.3.1 Calculating equipment investment needs

Various office supplies were used to print the necessary documents and reports. The list and cost of materials are given in Table 4.4. (see Table 4.4)

Table 4.4 – Cost of office supplies

No. s/n	Materials	Quantity Price,	UAH Cost, UAH	
1	Office paper in A4 format (500 pcs./ pack.) Matte files	1	185	185
2	in A4 format (100 pcs./pack.) Assorted stationery, pcs.	1	120	120
3	Printer refills for printing, pcs.	2	250	500
4		1	1200	1200
Together				2005

The total cost of stationery for documentation and printing is 2005 UAH.

4.3.2 Calculation of project executors' salaries

The work will be performed by a self-employed person registered as an individual entrepreneur.

A sole proprietorship is a convenient organizational and legal form for a small business. The advantages of this form are:

ÿ relative simplicity of registration and record keeping;

ÿ no need to hire a director;

ÿ no need to pay wages/dividends (individual funds) entrepreneur are automatically the funds of an individual);

ÿ the ability to choose any group of the simplified system taxation (under certain conditions).

For conducting business activities, Group II of the simplified taxation system with payment of a single tax is selected. The single tax rate is calculated based on the minimum wage of UAH 8,000 and from April 1, 2024 is UAH 1,420/month.

Thus, a single tax will need to be paid monthly in
amount of 1420 UAH.

Therefore,

$$EPproject = 1420 ÿ 5 = 7100 UAH.$$

Let's determine the salary of the work performer. In paragraph 4.2.3, it was
The daily wage is calculated. We have:

$$_{AT} = ÿ \quad , \tag{4.6}$$

where T is the complexity of the study.

After substituting the numerical values, we get

$$_{AT} = 59 ÿ 1190, 50 = 70239, 50 UAH$$

Mandatory payroll tax – single social contribution
The social security contribution is 22% of the salary. Let's calculate the social security contribution:

$$= ÿ 0.22 = 70239.50 ÿ 0.22 = 15452.69 UAH$$

The total amount of wages and payroll deductions is determined by the formula

$$_{general} = ÿ = 70239.5 + 15452.69 = 85692.19 UAH$$

4.3.3 Estimating the costs of developing a software project
product

As part of the preparation for work, it is necessary to find premises for the deployment of the infrastructure. For this purpose, premises with an area of 15 sq. m were rented at the enterprise. The premises are equipped with lighting, an electrical power supply network and other communications. The rent is 5100 UAH/month. The cost of paying for the Internet and mobile communications is 280 UAH/month. The cost of heating in the winter period is 2500 UAH/month.

The calculation of costs for internet, heating for rent is given in table 4.5. (see table 4.5.)

Table 4.5 – Calculation of rent, internet and heating costs

No. s/n	Name	Price per month, UAH	Number of months of service use	Cost for the entire period, UAH
1	Rent	5100	5	25500
2	Internet mobile and communication	280	5	1200
3	Heating	2500	3	7500
Total				34200

Calculation of electricity consumption for 59 working days at eight hours working day is given in Table 4.6.

The laptop consumes

$$= 1 \text{ ÿ } 0.18 \text{ ÿ } 472 = 84.96 \left(\frac{\text{kW}}{\text{hour}}\right),$$

using an external monitor (1 pc.) –

$$= 1 \text{ ÿ } 0.2 \text{ ÿ } 472 = 94.4 \left(\frac{\text{kW}}{\text{hour}}\right),$$

lighting the room with lamps (2 pcs.) –

$$= 2 \text{ ÿ } 0.04 \text{ ÿ } 472 = 37.76 \left(\frac{\text{kW}}{\text{hour}}\right),$$

router –

$$= 1 \text{ ÿ } 0.06 \text{ ÿ } 472 = 28.32 \left(\frac{\text{kW}}{\text{hour}}\right).$$

Table 4.6 – Calculation of electricity consumption for work

No. s/n	Name	Power, kW/h	Amount of time worked, hours	Amount of electricity consumed, kWh
	Lenovo laptop Dell	0.18	472	84.96
1	monitor Lighting	0.2	472	94.4
2 3	(lamps) TP-Link router	0.04	472	37.76
4		0.06	472	28.32
Together				245.44

According to information posted on the website of the National Commission for State Regulation in the Energy and Utilities Sectors, the cost of electricity at JSC Poltavaoblenergo for non-household consumers is 2-
The price of the 1st voltage class is 1.68 UAH per 1 kWh. [69]

Let's calculate the cost of electricity consumed:

$$= \ddot{y} \quad , \tag{4.7}$$

where P is the amount of electricity consumed, kW/h;

C – price per kW/h, UAH.

Let us calculate the cost of electricity consumed for the entire period of the study using formula (4.7):

$$= 245.44 \ddot{y} 1.68 = 412.33 \text{ (UAH)}$$

The implementation of the project does not require the purchase of new computing equipment, which is available today. However, for training artificial intelligence models, it was decided to rent a server with powerful video cards. The calculation of computing power consumption on a server with the necessary GPUs is given in Table 4.7. (see Table 4.7)

The choice fell on the tariff, which provides access to two Nvidia RTX 4090 TI video cards, which have sufficient power to train models, with a price of $0.905/hour. At the exchange rate of the Ministry of Finance, we translate into hryvnia:

= 0.905 ÿ 39.83 = 36.04 (UAH/hour)

Accordingly, training took place almost the entire development time, so the time The workout consists of:

= 24 ÿ 59 = 1416 hours.

We calculate the money spent on the cloud video card rental service:

= ÿ = 36.04 ÿ 1416 = 51032.64 UAH

Table 4.7 – Calculation of costs for cloud GPU rental service

No. s/n	Name	Cost, UAH/ hour	Amount of time worked, hours	Number spent funds, UAH
1	Rent 2 Nvidia video cards RTX 4090 TI	36.04	1416	51032.64
Together				51032.64

Let's calculate the total amount of depreciation deductions (per project).

The amount of depreciation is calculated using the formula

= ÿ / 100%, (4.8)

where F is the initial cost of fixed assets by type, UAH;

NA – annual depreciation rate by type of fixed assets, %.

Therefore,

$$= 28000 \text{ ÿ } \frac{20\%}{100\%} = 5600 \text{ (UAH)},$$

$$= 18000 \text{ ÿ } \frac{20\%}{100\%} = 3600 \text{ (UAH)},$$

$$= 4800 \text{ ÿ } \frac{20\%}{100\%} = 960 \text{(UAH)}.$$

The amount of depreciation deductions per month is determined by the formula

$$= ÿ / , \quad (4.9)$$

where AT is the annual amount of depreciation deductions, UAH;

N – project duration, days;

T – duration of the working year, days.

From here

$$= \frac{5600 ÿ 59}{251} = 1316.30(UAH),$$

$$= \frac{360 ÿ 59}{251} = 846, 20(UAH),$$

$$= \frac{960 ÿ 59}{251} = 225, 65(UAH),$$

The amount of depreciation deductions for the year and for the month is given in Table 4.8. (see table 4.8)

Table 4.8 – Calculation of the annual amount of depreciation deductions

No. s/n	Elements main means	Number	Cost, UAH	Amount, UAH	Depreciation rate, %	Depreciation deductions for the year, UAH	Depreciation deductions per month, UAH
1 Laptop 1 2 Monitor			28000	28000	20%	18000	5600	1316.3
1 3 Router 1 Total			18000	20%	4800	20%	3600	846.2
Software				4800		960	225.65
							2388.15

implementation activities involve business trips for installation and technical support of the program. The minimum monthly wage from 01.04.2024 is 8000 UAH (Article 8 of the Law of Ukraine "On the State Budget of Ukraine for 2024"). Therefore, the maximum amount of non-taxable daily allowances in 2024 within the territory of Ukraine is 800 UAH for each calendar day of such business trips.

As a result of taking into account all the above cost items, the cost of the project can be determined. The calculation of cost items for software product development is given in Table 4.9. The project duration is 59 days.

Table 4.9 – Project Cost Estimate

No. s/n	Calculation article	Amount, UAH
	Salary Single	70239.50
1	social contribution Single tax	15452.69
2 3	(paid from the moment of registration)	7100.00
4	Information services: Internet and mobile communications	1200.00
5	Utilities (electricity) Rent of premises Heating	245.44
6	Depreciation of fixed assets	25500.00
7	Office	7500.00
8	supplies Expenses for cloud services for	2388.15
9	project development	2005.00
10		51032.64
11	Travel expenses Total: cost of own work	0
		182663.42

Thus, as a result of the calculations performed, the costs of research and creation of software for image processing of the PROMPT definition
based on CLIP models amount to 182,663.42 UAH.

4.3.6 Plan of revenues from the sale of the software product and its costs production

Customers who expressed a desire to use the product and purchased a premium subscription are 2450 users. The expected revenue from the sale of a software product at a cost of UAH 179.20 is determined by the formula:

= ÿ ,

where N is the number of consumers;

C – cost of a software product unit, UAH.

It is also worth noting the need to calculate the cost including VAT. (20%)

Therefore,

= 2450 ÿ 179.20 = 439040 (UAH).

The economic efficiency of the proposed implementation was assessed using a system of indicators used in international and domestic practice (see Table 4.10). The following indicators were used to assess the economic efficiency of the project:
indicators:

- profit calculated by the formula;

- cost-effectiveness;

- profitability of income;

- payback period.

Table 4.10 – Calculation of the economic efficiency of the project

Indicator	Calculation method	Amount, UAH
Income, UAH		526848
Expenses, UAH		182663.42
Profit, UAH Income – Expenses Cost profitability, % (Profit / Expenses)		344184.57
* 100% Income profitability, % (Profit / Income) * 100%		188.42
		65.32

Conclusions to Chapter 4

The following tasks were completed in the economic section:

– market segmentation was carried out and the competitiveness of the proposed project was determined. It was concluded that the software product is competitive;

– calculation of the labor intensity of the work performed under the project – 59 person-days;

– project executors and their salaries have been appointed;

– a cost estimate for the project was prepared - UAH 182,663.42;

– the expected profit based on the results of the project implementation was calculated - UAH 344,184.57;

– the calculated cost profitability is 188.42%, and the income profitability is 65.32%.

CONCLUSIONS

In this thesis project, image preprocessing software was developed to determine PROMPT based on the CLIP model. This solution allows you to automate the image description process, which is an important step in many fields, such as computer vision, natural language processing, and image generation.

Key results and conclusions of this work:

1. Existing approaches and methods for image description are analyzed, in particular, deep learning-based models such as CLIP (Contrastive Language-Image Pre-training).

2. Software has been developed that integrates the CLIP model and provides image preprocessing, including loading, scaling, normalization, and conversion to the required format for further analysis.

3. An algorithm has been implemented to generate a PROMPT (text description) that best matches the content of the image, using the CLIP model and methods for finding the optimal solution.

4. The developed software was tested on various images, which allowed us to assess its effectiveness and accuracy in generating PROMPT.

5. The practical applicability of the developed solution in image description tasks has been demonstrated, which can be used in various industries, such as marketing, advertising, search engines, medicine, and others.

The developed software is flexible and scalable, allowing its further integration with other systems and applications. It can be a useful tool for automating the image description process, saving time and resources, and increasing the accuracy and relevance of the obtained results.

LIST OF LINKS

1. Image processing and its future implications - magnimind academy. Magnimind Academy - Launch a new career with our programs. URL: https://magnimindacademy.com/blog/image-processing-and-its-future-implications/ (access date: 10.06.2024).
2. CLIP: connecting text and images. OpenAI. URL: https://openai.com/research/clip (access date: 21.04.2024).
3. Active image indexing - meta research | meta research. Meta Research. URL: https://research.facebook.com/publications/active-image-indexing/ (access date: 10.06.2024).
4. The future of gaming: ai's transformation - whimsy games. Whimsy Games. URL: https://whimsygames.co/blog/the-future-of-gaming-ais-transformation/ (access date: 10.06.2024).
5. Magic3D: High-Resolution Text-to-3D Content Creation. Research at NVIDIA | Advancing the Technology | NVIDIA. URL: https://research.nvidia.com/labs/dir/magic3d/ (access date: 10.06.2024).
6. Machine learning–based automated image processing for quality management in industrial Internet of Things / N. Rahmatov et al. International journal of distributed sensor networks. 2019. Vol. 15, no. 10. P. 155014771988355. URL: https://doi.org/10.1177/1550147719883551 (access date: 11.06.2024).
7. Sester M., Feng Y., Thiemann F. Building generalization using deep learning. ISPRS - international archives of the photogrammetry, remote sensing and spatial information sciences. 2018. Vol. XLII-4. P. 565–572. URL: https://doi.org/10.5194/isprs-archives-xlii-4-565-2018 (access date: 11.06.2024).
8. Wang S., Wang H., Huang L. Adaptive algorithms for multi-armed bandit with composite and anonymous feedback. Proceedings of the AAAI conference on artificial intelligence. 2021. Vol. 35, no. 11. P. 10210–10217. URL: https://doi.org/10.1609/aaai.v35i11.17224 (access date: 11.06.2024).
9. The power of natural language processing. Harvard Business Review. URL: https://hbr.org/2022/04/the-power-of-natural-language-processing (access date: 10.06.2024).
10. The art of AI prompt crafting: a comprehensive guide for enthusiasts. OpenAI Developer Forum. URL: https://community.openai.com/t/the-art-of-ai-prompt-crafting-a-comprehensive-guide-for-enthusiasts/495144 (access date: 10.06.2024).
11. Horning J. Software fundamentals. ACM SIGSOFT software engineering notes. 2001. Vol. 26, no. 4. P. 91. URL: https://doi.org/10.1145/505482.505498 (access date: 11.06.2024).

12. Davis J., Sable T., Devers C. Modern system administration: building and maintaining reliable systems. O'Reilly Media, Incorporated, 2022. 300 p.

13. Processing payments | PortaOne Documentation | May 3rd, 2024 |. PortaOne Documentation. URL: https://docs.portaone.com/docs/mr114-payments (access date: 10.06.2024).

14. Configuring and managing B2B messaging. Moved. URL: https://docs.oracle.com/en/cloud/saas/supply-chain-and- manufacturing/24a/facmm/manage-confirmation-codes.html (date of access: 06/10/2024).

15. Sheldon A. Developing cross-platform applications with python: pros and cons. LinkedIn: Log In or Sign Up. URL: https://www.linkedin.com/pulse/developing-cross-platform-applications-python-pros-cons-amyra-sheldon (access date: 10.06.2024).

16. Frank A. Designing Data-Intensive. Independently Published, 2021.

17. Anderson K. Hacking: web application security. Independently Published, 2020.

18. The ultimate guide to python localization. Phrase. URL: https://phrase.com/blog/posts/python-localization/ (access date: 10.06.2024).

19. McKee A. Rust vs python: choosing the right language for your data project. Learn Data Science and AI Online | DataCamp. URL: https://www.datacamp.com/blog/rust-vs-python (access date: 10.06.2024).

20. OpenCV: OpenCV modules. OpenCV documentation index. URL: https://docs.opencv.org/4.10.0/ (access date: 10.06.2024).

21. Contributors to Wikimedia projects. Feature-driven development - Wikipedia. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/Feature-driven_development (access date: 10.06.2024).

22. Real Python. Socket programming in python (guide) - real python. Python Tutorials - Real Python. URL: https://realpython.com/python-sockets/ (access date: 10.06.2024).

23. Software architecture: the hard parts / M. Richards et al. O'Reilly Media, Incorporated, 2021.

24. freeCodeCamp.org. Python tkinter GUI design using ttkbootstrap - complete course, 2023. YouTube. URL: https://www.youtube.com/watch?v=0tM-l_ZsxjU (access date: 10.06.2024).

25. How HTTPS Works. How HTTPS works - How HTTPS works. URL: https://howhttps.works/ (access date: 10.06.2024).

26. Contributors to Wikimedia projects. Public-key cryptography - Wikipedia. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/Public-key_cryptography (access date: 10.06.2024).

27. Database-first and model-first - llblgen pro designer v5.8 documentation. LLBLGen Pro | Entity Modeling Solution and ORM framework for .NET. URL: https://www.llblgen.com/Documentation/5.8/Designer/Concepts/DatabaseFirstModelFirst.htm (access date: 10.06.2024).

28. MySQL documentation. https://docs.oracle.com/cd/E17952_01/index.html (access date: 10.06.2024). Moved. URL:

29. Contributors to Wikimedia projects. Database normalization - Wikipedia. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/Database_normalization (access date: 10.06.2024).

30. Introduction of ER Model - GeeksforGeeks. GeeksforGeeks. URL: https://www.geeksforgeeks.org/introduction-of-er-model/ (access date: 10.06.2024).

31. Database design basics - Microsoft Support. Microsoft Support. URL: https://support.microsoft.com/en-gb/office/database-design-basics-eb2159cf-1e30-401a-8084-bd4f9c9ca1f5 (access date: 10.06.2024).

32. Events in MySQL - GeeksforGeeks. GeeksforGeeks. URL: https://www.geeksforgeeks.org/events-in-mysql/ (access date: 10.06.2024).

33. How does SSL work? | entrust. Entrust | Identities, secure payments, and protected data. URL: https://www.entrust.com/resources/learn/how-does-ssl-work (access date: 10.06.2024).

34. IBM i 7.4. IBM - United States. URL: https://www.ibm.com/docs/en/i/7.4?topic=communications-socket-programming (access date: 10.06.2024).

35. SSL/TLS handshake – what is it? All about SSL/TLS handshake | ssl.com.ua blog. SSL.com.ua blog. URL: https://ssl.com.ua/blog/ukr/what-is-ssl-tls- handshake/ (date of application: 06/10/2024).

36. What is an SSL Certificate? | DigiCert. SSL Digital Certificate Authority | Encryption & Authentication | DigiCert.com. URL: https://www.digicert.com/what-is-an-ssl-certificate (access date: 10.06.2024).

37. Connect LiqPay to the online store | payment method settings. privatbank.ua. accepting URL: https://privatbank.ua/business/business-connect-liqpay (date of application: 06/10/2024).

38. GitHub - liqpay/sdk-python: LiqPay python sdk. GitHub. URL: https://github.com/liqpay/sdk-python (access date: 10.06.2024).

39. Contributors to Wikimedia projects. Business process - Wikipedia. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/Business_process (access date: 10.06.2024).

40. Engineering systems design rhapsody. IBM - United States. URL: https://www.ibm.com/docs/en/engineering-lifecycle-management-suite/design-rhapsody/9.0.1?topic=rhapsody-structural-model (access date: 10.06.2024).

41. Pyreverse - Pylint 3.3.0-dev0 documentation. Pylint 3.2.3 documentation. URL: https://pylint.readthedocs.io/en/latest/pyreverse.html (access date: 10.06.2024).

42. What is the most effective way to design a user interface for a desktop application?. LinkedIn: Sign In URL: Log https://www.linkedin.com/advice/3/ or Up. how-do-you-effectively-design-user-interface-desktop (access date: 10.06.2024).

43. VS code counter - visual studio marketplace. Extensions for Visual Studio family products Visual Marketplace. of | Studio URL: https://marketplace.visualstudio.com/items?itemName=uctakeoff.vscode-counter (access date: 10.06.2024).

44. Paper D. Advanced transfer learning. State-of-the-Art deep learning models in tensorflow. Berkeley, CA, 2021. P. 171–199. URL: https://doi.org/10.1007/978-1-4842-7341-8_7 (access date: 21.04.2024).

45. Leonov O. Automatic software testing (definition, creation process). Drukarnia. URL: https://drukarnia.com.ua/articles/avtomatichne-testuvannya-pz-viznachennya-proces-stvorennya-WWNqn (access date: 21.04.2024).

46. Design Patterns: Elements of Reusable Object-Oriented Software (Addison-Wesley Professional Computing Series) / R. Helm et al. Addison-Wesley Professional, 1995. 395 p.

47. Flash attention compilation warning?. PyTorch Forums. URL: https://discuss.pytorch.org/t/flash-attention-compilation-warning/196692 (date of access: 10.06.2024).

48. Pytest: helps you write better programs - pytest documentation. pytest: helps you write better programs - pytest documentation. URL: https://docs.pytest.org/en/8.0.x/ (date of access: 21.04.2024).

49. Testing web projects: basic steps and tips - QALight. QALight. URL: https://qalight.ua/baza-znaniy/testuvannya-veb-proektiv-osnovni-etapi-ta-poradi/ (date of application: 04/21/2024).

50. Axelrod A. Complete guide to test automation: techniques, practices, and patterns for building and maintaining effective software projects. Apress, 2018. 560 p.

51. Mohan G. Full stack testing: a practical guide for delivering high quality software. O'Reilly Media, Incorporated, 2022.

52. McLeod RJ, Everett GD Software testing: testing across the entire software development life cycle. Wiley-IEEE Computer Society Pr, 2007. 280 p.

53. Urma R.-G., Warburton R. Real-World software development. O'Reilly Media, Incorporated, 2019. 325 p.

54. Wiegers KE Software development pearls: lessons from fifty years of software experience. Pearson Education, Limited, 2000.

55. Singureanu C. Dynamic software testing - types, tools and more!. process, and ZAPTEST. URL: https://www.zaptest.com/uk/dynamic-testing-in-testing-process (access date: 04/21/2024).

56. Ramalho L. Fluent Python: Clear, Concise, and Effective Programming. O'Reilly Media, 2021. 850 p.

Pytest-qt. PyPI. URL: https://pypi.org/project/pytest-qt/ (date of access: 57. 10.06.2024).

58. What is software testing: types, stages, tools. Sigma Software University. URL: https://university.sigma.software/what-is-software-testing/ (accessed: 04/21/2024).

59. Beck K. Test driven development: by example (the addison-wesley signature series). Addison-Wesley Professional, 2002. 240 p.

60. Siddiqui S. Learning test-driven development: a polyglot guide to writing uncluttered code. O'Reilly Media, Incorporated, 2021. 275 p.

61. Forgács I., Kovács A. Modern software testing techniques. Berkeley, CA : Apress, 2024. URL: https://doi.org/10.1007/978-1-4842-9893-0 (date of access: 04/21/2024).

62. McLeod RJ, Everett GD Software testing: testing across the entire software development life cycle. Wiley-IEEE Computer Society Pr, 2007. 280 p.

63. Standardized information. The objective immutability of EU standards. URL: https://www.ukrstat.gov.ua/operativ/oper_new.html (access date: 11.06.2024).

64. Only 14% of Ukrainians use artificial intelligence. Internet Freedom. URL: https://netfreedom.org.ua/article/lishe-14-ukrayinciv-vikoristovuyut-shtuchnij- intelek#:~:text=ÿÿÿÿ%2014%20ÿÿÿÿÿÿÿÿÿ%20ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ%20ÿÿÿÿÿÿÿÿ,-%20Internet%20Freedom (date of application: 06/10/2024).

65. Almost 23 million Ukrainians regularly use the Internet – research. Mind.ua. URL: https://mind.ua/news/20204323-majzhe-23-mln-ukrayinciv-regulyarno-koristuyutsya-internetom-doslidzhennya (date of application: 06/11/2024).

66. Population clock. Census.gov. URL: https://www.census.gov/popclock/ (access date: 10.06.2024).

67. Ballard J. Americans' top feeling about AI: caution | YouGov. YouGov | What the world thinks. URL: https://today.yougov.com/technology/articles/49099-americans-2024-poll-ai-top-feeling-caution\ (access date: 10.06.2024).

68. US Dollar (USD) - Hryvnia (UAH) ÿ Convert dollar to hryvnia. Ministry of Finance - all about finance: news, exchange rates, banks. URL: https://minfin.com.ua/currency/converter/usd-uah/ (date of application: 06/10/2024).

69. Active | LLC "POLTAVAENERGOZBUT". LLC "POLTAVAENERGOZBUT" | supply of electricity to the consumer. URL: https://www.energo.pl.ua/diyuchi/ (date of application: 06/11/2024).