Міністерство освіти і науки України

Луцький національний технічний університет

Факультет комп'ютерних та інформаційних технологій

Кафедра інженерії програмного забезпечення

КВАЛІФІКАЦІЙНА РОБОТА ЗА СТУПЕНЕМ ВИЩОЇ ОСВІТИ «БАКАЛАВР»

СИСТЕМА АВТОМАТИЧНОГО РОЗТАШУВАННЯ КАМЕР ВІДЕОСПОСТЕРЕЖЕННЯ У ПРИМІЩЕННІ НА БАЗІ DJANGO

SYSTEM FOR AUTOMATIC PLACEMENT OF VIDEO SURVEILLANCE CAMERAS IN THE ROOM BASED ON DJANGO

спеціальність 121 Інженерія програмного забезпечення
(шифр і назва спеціальності)

освітня програма «Інженерія програмного забезпечення»
(назва освітньої програми)

Виконав: здобувач вищої освіти
Групи ІПЗс-21
Смірнов Максим Вікторович
_____
(підпис)
Керівник:
к.т.н., доцент
Ящук Андрій Анатолійович
_____
(підпис)

Кваліфікаційну роботу
допущено до захисту
«1» 06 2024 р.
к.т.н., доцент
Гарант освітньої програми.
Ліщина Наталія Миколаївна
_____
(підпис)

Луцьк – 2024 року

Луцький національний технічний університет

Факультет *комп'ютерних та інформаційних технологій*
Кафедра *інженерії програмного забезпечення*
Ступінь вищої освіти. *бакалавр*
Галузь знань. *12 Інформаційні технології*
Спеціальність. *121 Інженерія програмного забезпечення*
Освітня програма. *«Інженерія програмного забезпечення»*

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Н.Ліщина
«2» 02 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧУ ВИЩОЇ ОСВІТИ

Смірнову Максиму Вікторовичу
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи. Система автоматичного розташування камер відеоспостереження у приміщенні на базі Django
Керівник роботи. *к.т.н., доцент, Ящук Андрій Анатолійович*
затверджені наказом закладу вищої освіти від «*30» грудня 2023 р.* № *477/01-02*
2. Строк подання здобувачем вищої освіти кваліфікаційної роботи *«5» червня 2024 р.*
3. Вихідні дані до роботи. *технічне та програмне забезпечення ЕОМ, вимоги до розробки програмного забезпечення, ергономічні вимоги до функціонування програмного засобу.*

4. Зміст розрахунково-пояснювальної записки (перелік питань, що потрібно розробити).
Аналіз сучасного стану проблеми, існуючих методів і засобів її розв'язання, аналіз, визначення вимог до розроблюваного програмного забезпечення та проектування програмного забезпечення, опис функціонального наповнення об'єкта проектування, практична реалізація об'єкта проектування.

5. Перелік графічного матеріалу. *20 рисунки; 3 схеми.*

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис	
		завдання видав	завдання прийняв
Аналіз предметної області	*Ящук А. А.*		
Специфікації вимог до розробленої системи	*Ящук А. А.*		
Розробка об'єкта проєктування	*Ящук А. А.*		
Нормоконтроль	*Повстяна Ю. С.*		
Гарант ОП	*Ліщина Н. М.*		
Показник запозичень тексту	12,3 %		
Академічна доброчесність	*Ящук А. А.*		

7. Дата видачі завдання *«2» лютого 2024 р.*

КАЛЕНДАРНИЙ ПЛАН

№ З/п	Назва етапів кваліфікаційної роботи бакалавра	Строк виконання етапів роботи	Примітка
1	*Огляд літературних джерел по темі кваліфікаційної роботи бакалавра*	*07.02.2024 р.*	вик.
2	*Аналіз проблеми розробки та впровадження об'єкту проектування*	*27.02.2024 р.*	вик.
3	*Обгрунтування вибору шляхів, технологій (алгоритмів) і засобів вирішення поставленого завдання*	*12.03.2024 р.*	вик.
4	*Розробка функціонально-структурної схеми роботи об'єкта проектування та проектування бази даних*	*17.04.2024 р.*	вик.
5	*Практична реалізація об'єкта проектування та розробка бази даних*	*18.05.2024 р.*	вик.
6	*Тестування та налагодження об'єкта проектування*	*01.06.2024 р.*	вик.
7	*Здача чистового варіанту кваліфікаційної роботи бакалавра на кафедру*	*05.06.2024 р.*	вик

Здобувач вищої освіти _____ (підпис) *Смірнов М. В.* (прізвище, ініціали)

Керівник кваліфікаційної роботи _____ (підпис) *Ящук А. А.* (прізвище, ініціали)

Міністерство освіти і науки України
Луцький національний технічний університет

Рецензія
на кваліфікаційну роботу

Здобувач вищої освіти: *Смірнов Максим Вікторович*

Тема: *«Система автоматичного розташування камер відеоспостереження у приміщенні на базі Django».*

Коротка характеристика кваліфікаційної роботи та прийнятих рішень:
Кваліфікаційна робота бакалавра складається з трьох розділів, в яких проаналізовані проблематика та поставлені завдання за темою роботи, здійснено теоретичне дослідження та практичну реалізацію системи автоматичного розташування камер відеоспостереження у приміщенні, здійснено тестування розробленої системи

Самостійні розробки і пропозицій автора: *Автор самостійно розробив систему автоматичного розташування камер відеоспостереження у приміщенні.*

Практичне значення роботи *полягає в тому, що розроблений засіб допомагає оптимально розташувати камери відеоспостереження в приміщенні.*

Недоліки: *Істотних недоліків в роботі не виявлено.*

Загальний висновок: *У кваліфікаційній роботі бакалавра здійснено аналіз предметної області, вибір методів і засобів для розробки, розроблено програмний продукт, здійснено його тестування. Усі поставлені завдання виконано в повному обсязі.*

Кваліфікаційна робота здобувача освіти Смірнова Максима Вікторовича рекомендується до захисту екзаменаційній комісії та заслуговує позитивної оцінки.

Рецензент кваліфікаційної роботи: Саварин Павло Вікторович, к.п.н., доцент кафедри ЦОТ

(підпис)

«07» червня 2024 р.

Міністерство освіти і науки України
Луцький національний технічний університет
Факультет комп'ютерних та інформаційних технологій
Кафедра інженерії програмного забезпечення

ВІДГУК
КЕРІВНИКА НА КВАЛІФІКАЦІЙНУ РОБОТУ

Здобувач вищої освіти *Смірнов Максим Вікторович*
(прізвище, ім'я, по батькові)
група *ІПЗс-21*

Тема кваліфікаційної роботи:
Система автоматичного розташування камер відеоспостереження у приміщенні на базі Django
Керівник *Ящук Андрій Анатолійович*

Актуальність теми
У сучасному світі безпека є однією з найважливіших потреб як для бізнесу, так і для приватних осіб. Камери відеоспостереження використовуються для запобігання злочинам, моніторингу персоналу, контролю доступу та забезпечення загальної безпеки приміщень. Система, що автоматично визначає оптимальне розташування камер, може значно підвищити ефективність цих заходів

Об'єкт дослідження
Розробка системи автоматичного розташування камер у приміщеннні

Характеристика теоретичного рівня, наявності самостійних розробок і практичної значимості роботи:
Автор аналізує різні методи та засоби для розробки функціональної системи розташування камер відеоспостереження у приміщенні. Визначає найкращі з них і обґрунтовує своє рішення. Зміст роботи відповідає обраній темі.

Зауваження та недоліки: *Суттєвих недоліків в роботі не виявлено.*

Загальний висновок:
Кваліфікаційна робота бакалавра виконана на високому рівні. Робота виконана відповідно до вимог, логічно й послідовно описує хід виконання поставленого завдання. Пояснювальна записка відповідає стандартам до її оформлення. Робота може бути оцінена на високу оцінку

Керівник _____ **(*Ящук А.А.*)**
(підпис) (прізвище, ім'я, по батькові)

«7» червня 2024 р.

АНОТАЦІЯ

Смірнов М. В. Система автоматичного розташування камер відеоспостереження у приміщенні на базі Django. Рукопис.

Кваліфікаційна робота бакалавра ОП «Інженерія програмного забезпечення». Луцький національний технічний університет. Луцьк, 2024.

Кваліфікаційна робота бакалавра складається зі вступу, трьох розділів, висновків, списку використаних джерел та додатків.

У першому розділі роботи аналізується предметна область. У другому розділі кваліфікаційної роботи представлена специфікація вимог до розроблюваної системи. У третьому розділі кваліфікаційної роботи бакалавра описано систему автоматичного розташування камер відеоспостереження у приміщенні. У висновках узагальнено інформацію, відображену у попередніх частинах. Результати розробок можуть бути застосовані в практичній діяльності.

Ключові слова: веб-додаток, клієнт-серверна архітектура, зручний інтерфейс.

ABSTRACT

Smirnov M. V. System for Automatic Placement of Video Surveillance Cameras in the Room Based on Django. Manuscript.

Bachelor's qualifying thesis of the OP "Software Engineering". Lutsk National Technical University. Lutsk, 2024.

The bachelor's qualification work consists of an introduction, three chapters, conclusions, a list of used sources and appendices.

The subject area is analyzed in the first section of the work. The specification of requirements for the developed system is presented in the second section of the qualification work. The third chapter of the thesis describes the system of automatic location of video surveillance cameras in the room. The conclusions summarize the information presented in the previous parts. Development results can be applied in practical activities.

Keywords: web application, client-server architecture, user-friendly interface.

ЗМІСТ

ВСТУП

Актуальність теми. У сучасному світі безпека є однією з найважливіших потреб як для бізнесу, так і для приватних осіб. Камери відеоспостереження використовуються для запобігання злочинам, моніторингу персоналу, контролю доступу та забезпечення загальної безпеки приміщень. Система, що автоматично визначає оптимальне розташування камер, може значно підвищити ефективність цих заходів. Сучасні технології, такі як машинне навчання, штучний інтелект та алгоритми обробки зображень, дозволяють створювати системи, які можуть автоматично аналізувати простір та визначати найкращі місця для встановлення камер. Це забезпечує більш ефективне використання обладнання та ресурсів.

Ручне планування розташування камер відеоспостереження може бути трудомістким і дорогим процесом. Автоматизація цього процесу знижує витрати на проектування та встановлення системи відеоспостереження, а також мінімізує ризик людських помилок. Автоматизовані системи можуть враховувати безліч факторів, таких як освітлення, кути огляду, можливі перешкоди та маршрути руху людей. Це дозволяє забезпечити повніше охоплення простору та мінімізувати сліпі зони, що підвищує ефективність відеоспостереження.

Автоматизовані системи легко масштабуються та адаптуються до змін у плануванні приміщень або функціональних потреб. Це особливо важливо для великих об'єктів, таких як торгові центри, офісні будівлі або промислові підприємства, де ручне розміщення камер може бути надзвичайно складним. У разі змін у конфігурації приміщень або в умовах експлуатації, автоматична система може швидко переналаштувати розташування камер, забезпечуючи безперервний високий рівень безпеки та контролю.

Автоматичне розташування камер відеоспостереження у приміщенні є надзвичайно актуальною темою в контексті сучасних вимог до безпеки та технологічних інновацій. Це напрямок, який об'єднує інтереси як практичних застосувань, так і наукових досліджень, спрямованих на створення більш ефективних та економічних рішень у сфері безпеки.

Метою роботи є створення інноваційної системи автоматичного розташування камер відеоспостереження, яка забезпечить високу ефективність, надійність та гнучкість у плануванні та експлуатації систем безпеки в різних типах приміщень.

Основні завдання:

– здійснити аналіз існуючих аналогів розміщення камер в приміщенні;

– створити математичну модель приміщення для аналізу можливих розташувань камер;

– розробити алгоритми для автоматичного визначення оптимальних позицій камер, з урахуванням заданих критеріїв;

– створити програмне забезпечення для автоматичного розташування камер;

– забезпечити інтуїтивно зрозумілий інтерфейс для користувачів;

– провести тестування системи.

Об’єктом дослідження є сукупність методів, технологій та інструментів, які дозволяють автоматизувати процес розташування камер відеоспостереження у приміщеннях.

Предметом дослідження є конкретні технологічні та алгоритмічні рішення, спрямовані на автоматизацію процесу розташування камер відеоспостереження в приміщеннях.

Практичне значення системи автоматичного розташування камер відеоспостереження у приміщенні полягає в підвищенні ефективності та надійності систем безпеки, зниженні витрат, забезпеченні гнучкості та масштабованості, а також у впровадженні передових технологій для постійного вдосконалення систем відеоспостереження.

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ СИСТЕМИ АВТОМАТИЧНОГО РОЗТАШУВАННЯ КАМЕР ВІДЕОСПОСТЕРЕЖЕННЯ У ПРИМІЩЕННІ

1.1 Аналіз сучасного стану проблеми систем автоматичного розташування камер відеоспостереження у приміщенні

Програмний комплекс для системи автоматичного розташування камер відеоспостереження може включати декілька підсистем, кожна з яких відповідає за виконання певних функцій.

Ось можливий опис таких підсистем. Ця підсистема відповідає за аналіз геометричних та фізичних параметрів приміщення, таких як розміри, форма, положення дверей, вікон, колон і т. д. Вона використовує ці дані для визначення оптимального розташування камер відеоспостереження та мінімізації сліпих зон.

Підсистема обробки зображень відповідає за обробку відеопотоків з камер в реальному часі. Використовується для виявлення руху, аналізу об’єктів, виявлення сліпих зон тощо. Підсистема аналізу даних освітлення аналізує дані про освітлення в приміщенні, а також враховує вплив освітлення на якість зображення та можливість виявлення об’єктів.

Підсистема стратегії розташування камер розробляє стратегії для оптимального розташування камер відеоспостереження, враховуючи результати аналізу просторових даних та обробки зображень, використовує алгоритми машинного навчання та оптимізації для підвищення ефективності стратегій розташування і забезпечує інтерфейс для введення вхідних даних (наприклад, план приміщення, параметри камер тощо) та відображення результатів. Вона може включати в себе графічний інтерфейс користувача для зручного взаємодії з системою, відповідає за інтеграцію з існуючими системами безпеки, такими як системи виявлення руху, виклику тривоги тощо, забезпечує обмін даними та взаємодію між системою автоматичного розташування камер і іншими системами безпеки [1].

Ці підсистеми утворюють комплексне програмне забезпечення для автоматичного розташування камер відеоспостереження та забезпечують оптимальний та ефективний процес відеоспостереження в приміщеннях. Підсистеми програмного комплексу для системи автоматичного розташування камер відеоспостереження можуть включати:

– модуль аналізу геометрії приміщення, що забезпечує аналіз геометрії приміщення на основі вхідних даних. Визначає розміри, форму та особливості приміщення для подальшого використання в системі;

– модуль аналізу сліпих зон та перекриття. Використовує інформацію про геометрію приміщення для визначення зон сліпих зон та перекриття між камерами, розробляє алгоритми для мінімізації цих зон шляхом оптимізації розташування камер, аналізує освітлення у приміщенні для визначення оптимального розташування камер, враховує фактори, такі як природне та штучне освітлення, для забезпечення належної якості зображення;

– модуль адаптації до зміни умов. Відповідає за автоматичне виявлення змін у приміщенні (наприклад, переміщення меблів, зміни в освітленні) і відновлення оптимального розташування камер;

– модуль врахування областей інтересу. Аналізує області інтересу в приміщенні (наприклад, входи, виходи, каси) та враховує їх при визначенні розташування камер;

– модуль інтеграції з існуючими системами безпеки. Забезпечує інтеграцію системи з існуючими системами безпеки, такими як системи виявлення руху, системи тривоги тощо;

– модуль візуалізації та налаштування. Забезпечує інтерфейс для відображення графічної інформації про оптимальне розташування камер на плані приміщення, дозволяє користувачу налаштовувати параметри системи та візуалізувати результати.

Ці підсистеми разом створюють програмний комплекс, який забезпечує автоматичне розташування камер відеоспостереження у приміщеннях з урахуванням різних факторів та умов.

Система управління базами даних (СУБД) в контексті системи автоматичного розташування камер відеоспостереження може включати наступні компоненти:

– визначення структури бази даних, включаючи таблиці, поля та зв’язки між ними. Структура бази даних може включати дані про геометрію приміщення, розташування камер, параметри камер та іншу важливу інформацію [1];

– модуль зберігання та доступу до даних. Забезпечує функції для зберігання, оновлення та видалення даних у базі даних, надає можливість доступу до даних для інших компонентів системи, таких як аналітичні модулі та візуалізаційні інструменти;

– модуль керування транзакціями забезпечує управління транзакціями, які змінюють стан бази даних, забезпечуючи їх атомарність, консистентність, ізоляцію та стійкість (ACID-властивості);

– модуль оптимізації запитів. Аналізує та оптимізує запити до бази даних для підвищення продуктивності та ефективності виконання, враховує індекси, статистику та інші параметри для вибору оптимального плану виконання запиту;

– модуль безпеки даних. Забезпечує захист даних у базі даних шляхом визначення прав доступу до даних для різних користувачів та ролей, включає функції автентифікації, авторизації та аудиту доступу до даних;

– модуль резервного копіювання та відновлення. Забезпечує можливість створення резервних копій бази даних та відновлення їх у випадку аварій або втрати даних;

– модуль моніторингу та управління базою даних. Надає інструменти для моніторингу та управління роботою бази даних, включаючи перегляд використання ресурсів, виявлення проблем та вирішення їх.

Ці компоненти спільно створюють систему управління базами даних, яка забезпечує надійне та ефективне зберігання, організацію та доступ до даних, необхідних для функціонування системи автоматичного розташування камер відеоспостереження [2].

Система користувачів для системи автоматичного розташування камер відеоспостереження включає в себе наступні компоненти:

– аутентифікація та авторизація: забезпечення процесу аутентифікації користувачів для входу до системи, наприклад, через логін і пароль або інші методи автентифікації;

– визначення прав доступу до функціональності системи на основі ролей користувачів (наприклад, адміністратор, оператор, аналітик);

– можливість створення, редагування та видалення облікових записів користувачів;

– налаштування прав доступу для кожного користувача відповідно до їхніх обов’язків та вимог системи;

– запис подій, пов’язаних з діяльністю користувачів у системі;

– забезпечення можливості перегляду та аналізу журналу подій для виявлення випадків порушення безпеки або інших проблем;

– розробка зручного та інтуїтивно зрозумілого інтерфейсу для користувачів системи;

– надання можливості керування різними аспектами системи, включаючи налаштування параметрів камер, перегляд відеопотоків, доступ до аналітичних звітів тощо;

– повідомлення та сповіщення: механізми надсилання повідомлень та сповіщень користувачам про важливі події або тривоги, виявлені системою відеоспостереження;

– забезпечення доступу до системи з різних пристроїв та платформ, таких як комп’ютери, смартфони, планшети тощо;

– розробка веб-інтерфейсу або мобільних додатків для зручного користування системою на різних пристроях.

Ці компоненти дозволяють ефективно керувати користувачами системи автоматичного розташування камер відеоспостереження, забезпечуючи безпеку, зручність та ефективність роботи з системою [3].

Постановка задачі: розробка системи автоматичного розміщення камер відеоспостереження в приміщенні.

Розглянуто всі вимоги до системи, яка є веб-додатком. Описано завдання, які має вирішувати розроблене програмне забезпечення, розглянуто вхідні та вихідні дані системи. Сформульовано перелік підсистем та чіткі вимоги до їх реалізації.

2D та 3D моделювання – CAD5D є першим онлайн CAD у режимі 2D і 3D для проектувальників. Ось кілька можливих характеристик та переваг цієї платформи:

– CAD5D надає можливість працювати у режимі онлайн, що дозволяє користувачам створювати, редагувати та ділитися своїми проектами безпосередньо через веб-браузер [4];

– 2D та 3D моделювання: платформа підтримує як двовимірне, так і тривимірне моделювання, що дозволяє користувачам створювати як плани та схеми, так і тривимірні моделі будівельних об’єктів;

– інтуїтивний інтерфейс: CAD5D може містити інтуїтивний і простий у використанні інтерфейс, який дозволяє користувачам швидко навчатися та ефективно використовувати всі функції платформи. Платформа може підтримувати можливість спільної роботи над проектами, дозволяючи кільком користувачам працювати над одним проектом одночасно та здійснювати обмін даними;

– інтеграція з іншими програмами: CAD5D може мати можливість інтеграції з іншими програмами і сервісами, такими як програми для обробки зображень, архітектурні програми та інші інструменти, що полегшує обмін даними та спільну роботу;

– обробка великих обсягів даних: платформа може бути оптимізована для обробки великих обсягів графічних даних, що дозволяє користувачам працювати над складними проектами з великою кількістю деталей.

Загалом, CAD5D може представляти собою потужний і зручний інструмент для проектування та моделювання як у двовимірному, так і у тривимірному просторі для проектувальників і будівельників [5].

IP Video System Design Tool – це програмне забезпечення, яке допомагає проектувальникам і інженерам швидко та ефективно розраховувати системи відеоспостереження. Ось деякі ключові можливості цього інструменту:

– пошук оптимальної кількості та розташування камер: засоби програмного забезпечення дозволяють аналізувати місцевість або приміщення та розраховувати оптимальну кількість камер відеоспостереження та їхнє розташування для найкращого охоплення зон ідентифікації;

– розрахунок системи відеоспостереження: програмне забезпечення дозволяє виконувати розрахунки щодо параметрів системи відеоспостереження, таких як роздільна здатність камер, кут огляду, дальність детекції, освітлення тощо;

– оцінка довжини кабелів: інструмент може розраховувати необхідну довжину кабелів для підключення камер відеоспостереження до системи, враховуючи їхнє розташування та фізичні параметри кабелів;

– відображення на плані місцевості або приміщення зон ідентифікації: інструмент дозволяє створювати візуалізації, які показують розташування камер відеоспостереження та їхнє покриття на плані місцевості або приміщення, щоб забезпечити оптимальне покриття зон ідентифікації.

В цілому, IP Video System Design Tool є потужним інструментом для швидкого та ефективного проектування систем відеоспостереження, який дозволяє інженерам і проектувальникам проводити детальний аналіз і розрахунки для досягнення оптимальних результатів [5].

Системи IP Video: інструмент дизайну системи IP Video дозволяє користувачам обчислити точну фокусну відстань об’єктива та кути огляду кожної камери, а також перевіряти поле зору кожної камери. Ось як це можливо здійснити:

– обчислення точної фокусної відстані об’єктива: користувач може вказати параметри камери (такі як тип об’єктива, розмір матриці, відстань до об’єкта) в інструменті, який враховує ці дані та розраховує точну фокусну відстань об’єктива для найчіткішого зображення;

– визначення кутів огляду кожної камери: інструмент також дозволяє користувачам визначати кути огляду кожної камери, що важливо для визначення покриття кожною камерою.

Користувач може ввести параметри камери та характеристики об’єктів, які потрібно спостерігати, а інструмент розрахує кути огляду камери. Перевірка поля зору кожної камери – користувач може перевірити поле зору кожної камери, щоб переконатися, що потрібні об’єкти або зони зображені на зображенні. Інструмент може надати візуальне відображення поля зору камери на плані місцевості або приміщення. Ці функції дозволяють користувачам ефективно планувати та налаштовувати системи відеоспостереження, забезпечуючи оптимальне покриття та якість зображення для кожної камери [5].

Незважаючи на те, що у розглянутих систем більше переваг, ніж недоліків, можна зробити висновок, що ідеальної системи поки не існує. Тому кожен інженер, проектувальник системи внутрішнього відеоспостереження або монтажник вибирає найбільш зручне програмне забезпечення, яке може вирішити проблеми. Однак користувачеві, який має намір заощадити і використовувати систему вдома, вибрати немає з чого. Описані вище системи не можуть працювати без фахівця з профільною освітою. Тобто, звичайний користувач повинен найняти співробітника, який вміє працювати з такими системами або може надати для них вхідні дані. Інженер повинен добре знати інтерфейс програми, проводити спеціальні вимірювання, керувати проектом планування та стежити за точністю. Тож можна точно сказати, що йде розробка системи автоматичного визначення місця розташування. Камери відеоспостереження доречні і актуальні в наш час.

1.2 Постановка завдання на кваліфікаційну роботу бакалавра

В результаті анлізу предметної області було сформульовано наступні завдання на кваліфікаційну роботу бакалавра:

– здійснити аналіз існуючих аналогів розміщення камер в приміщенні, для подальшого вдосконалення;

– створити математичну модель приміщення для аналізу можливих розташувань камер. Впровадження нових алгоритмів дозволить постійно вдосконалювати систему та адаптуватися до нових умов експлуатації;

– розробити алгоритми автоматизованого розташування камер, які забезпечать оптимальне покриття простору, мінімізуючи сліпі зони і покращуючи загальну ефективність системи відеоспостереження. Використання технологій машинного навчання та комп'ютерного зору дозволить точно аналізувати просторові характеристики приміщень і адаптуватися до змін у конфігурації простору. Автоматизація процесу планування розташування камер значно знизить витрати на проектування та встановлення системи відеоспостереження. Оптимізація використання обладнання забезпечить ефективне розташування меншої кількості камер, що зменшить витрати на придбання та обслуговування обладнання;

– створити програмне забезпечення для автоматичного розташування камер. Розробити систему, яка демонструватиме високу гнучкість і може буде легко адаптуватися до змін у конфігурації приміщень та вимог безпеки. Система має бути здатна масштабуватися для використання в різних типах і розмірах приміщень, від невеликих офісів до великих промислових об'єктів;

– забезпечити інтуїтивно зрозумілий інтерфейс для користувачів системи автоматичного розташування камер відеоспостереження у приміщенні представляє собою значний крок вперед у сфері безпеки та моніторингу. Автоматизувати процес розташування камер, інтегрувати з існуючими системами та використати передові технології, щоб ця система стала важливим інструментом для забезпечення безпеки в сучасному світі;

– провести тестування системи для підтвердження правильності роботи, забезпечення високої якості та надійності програмного продукту.

РОЗДІЛ 2
МЕТОДИ ТА ПРОГРАМНІ ЗАСОБИ СИСТЕМИ АВТОМАТИЧНОГО РОЗТАШУВАННЯ КАМЕР ВІДЕОСПОСТЕРЕЖЕННЯ У ПРИМІЩЕННІ

2.1 Аналіз, визначення вимог до розроблюваного програмного забезпечення та проектування програмного забезпечення

Matplotlib – це потужна бібліотека для створення двовимірних графіків у мові програмування Python. Вона дійсно бере свій початок в емуляції графічних команд MATLAB, але вона не залежить від MATLAB і може використовуватися в об’єктно-орієнтованому способі. Ось деякі ключові особливості Matplotlib:

– чистий Python: Matplotlib написаний в основному на мові програмування Python, що робить його легким у використанні та розширенні;

– використання NumPy: бібліотека використовує NumPy для роботи з числовими даними та масивами, що забезпечує хорошу продуктивність навіть для великих обсягів даних;

– гнучкість та розширюваність: Matplotlib надає широкі можливості налаштування графіків, включаючи налаштування ліній, кольорів, шрифтів, маркерів тощо. Вона також підтримує використання різних стилів графіків;

– різноманітні типи графіків: Matplotlib дозволяє створювати різноманітні типи графіків, включаючи лінійні, стовпчасті, колові, контурні графіки, гістограми, точкові графіки, теплові карти тощо [5];

– підтримка LaTeX: Matplotlib підтримує використання LaTeX для відображення математичних символів та формул на графіках;

– платформонезалежність: бібліотека підтримується на різних операційних системах, таких як Windows, macOS, Linux, що робить її універсальною для використання на різних платформах.

Загалом, Matplotlib є потужним інструментом для візуалізації даних у Python і широко використовується як в наукових, так і візуалізаційних застосунках [5].

Matplotlib – це потужна бібліотека для створення двовимірних графіків у мові програмування Python. Вона дійсно почала свій шлях як імітація графічних команд MATLAB, але з часом стала власною, незалежною бібліотекою. Основні особливості Matplotlib включають:

– об’єктно-орієнтований підхід: надає можливість створювати графіки через об’єкти, що дозволяє більшу гнучкість та контроль над відображенням;

– Matplotlib інтегрується з іншими популярними бібліотеками Python, такими як NumPy, що дозволяє ефективно працювати з великими обсягами даних;

– різноманітність графіків: Matplotlib надає широкий спектр можливостей для створення різних типів графіків, включаючи лінійні, стовпчасті, кругові, гістограми, контурні, поверхневі та інші;

– бібліотека працює на різних операційних системах і може використовуватися як у відкритих, так і у комерційних проектах.

Matplotlib є однією з найпопулярніших інструментів для візуалізації даних у Python завдяки своїй потужності, гнучкості та зручності використання.

2.2 Вибір засобів, методів і алгоритмів вирішення поставленого завдання

PyCharm є інтегрованим середовищем розробки (IDE) для мови програмування Python, розробленим компанією JetBrains і призначеним спеціально для програмістів, які працюють з Python. Основні особливості PyCharm включають:

– аналіз коду: PyCharm забезпечує потужні інструменти для аналізу коду, автоматичне виявлення помилок, наголошення синтаксичних помилок та підказки з використанням рекомендацій PEP8;

– графічний налагоджувач, який дозволяє зупиняти, відлагоджувати та виконувати код крок за кроком, а також виводити змінні та структури даних під час виконання;

– інтегрований тестер одиниць: підтримує інтегрований тестувальник одиниць для автоматизованого тестування коду та виявлення помилок. Інтеграція з системами управління версіями (VCS);

– підтримує інтеграцію з такими системами управління версіями, як Git, SVN, Mercurial та інші;

– підтримка веб-розробки з Django забезпечує спеціалізовані інструменти для розробки веб-додатків на основі фреймворка Django;

– підтримка Data Science з Anaconda: має можливість роботи з пакетом Anaconda для розробки програм в галузі Data Science та наукових обчислень;

– кросплатформенність: PyCharm доступний для операційних систем Windows, macOS та Linux, що дозволяє розробникам працювати на різних платформах.

PyCharm є дуже популярним інструментом для розробки на Python завдяки своїм розширеним можливостям, зручному інтерфейсу та підтримці великої кількості різних фреймворків і бібліотек [2].

PyCharm – це інтегроване середовище розробки (IDE), призначене спеціально для роботи з мовою програмування Воно розроблене компанією JetBrains з Чехії і є одним з найпопулярніших інструментів для розробки на Python. Основні функції та можливості PyCharm включають:

– аналіз коду: PyCharm надає розширені інструменти для аналізу коду, включаючи перевірку синтаксису, автоматичне доповнення коду, відстеження помилок та інші корисні функції;

– графічний налагоджувач: IDE має вбудований графічний налагоджувач, що дозволяє легко відлагоджувати Python-код шляхом пошуку та виправлення помилок у виконанні програм;

– інтегрований тестер одиниць: PyCharm надає можливість створювати, запускати та відлагоджувати тести одиниць коду безпосередньо в IDE;

– інтеграція з системами управління версіями (VCS). Підтримка інтеграції з такими системами, як Git, SVN, Mercurial, дозволяє командам програмістів працювати разом над проектами та вести контроль версій;

– підтримка веб-розробки з Django: PyCharm має спеціальні інструменти для розробки веб-програм на основі фреймворку Django, що полегшує роботу з цим популярним фреймворком;

– IDE інтегрується з платформою Anaconda, що дозволяє зручно працювати з інструментами Data Science, такими як Jupyter Notebook, NumPy, Pandas та іншими;

– PyCharm доступний для операційних систем Windows, macOS та Linux, що робить його універсальним інструментом для розробки на Python на будь-якій платформі [2].

PyCharm – це потужний і зручний інструмент для розробки програм на Python, який дозволяє розробникам працювати ефективно над різноманітними проектами.

Python – це високорівнева, інтерпретована мова програмування, яка відома своєю простотою, зручністю використання та широким спектром застосувань.

Основні особливості Python включають:

– простий синтаксис: Python має простий і зрозумілий синтаксис, який робить його легким для вивчення та розуміння, навіть для початківців;

– широкий спектр застосувань: Python використовується у багатьох областях, включаючи веб-розробку, наукові обчислення, штучний інтелект, аналіз даних, автоматизацію, розробку ігор та багато інших;

– багатофункціональність: Python має велику кількість стандартних бібліотек та сторонніх модулів, що розширюють його можливості для різних завдань;

– платформенна незалежність: програми на Python можуть працювати на різних операційних системах, таких як Windows, macOS, Linux та інші;

– велике співтовариство: Python має велике та активне співтовариство розробників, яке активно співпрацює над розвитком мови та її екосистеми [2].

Python став дуже популярним в останні роки завдяки своїм перевагам та зручності використання. Ця мова програмування знаходить своє застосування в

різних сферах та допомагає вирішувати різноманітні завдання від найпростіших до найскладніших.

JavaScript – це високорівнева, інтерпретована мова програмування, яка використовується переважно для розробки веб-додатків та веб-сайтів. Ось деякі ключові особливості та характеристики JavaScript:

– веб-розробка: JavaScript є однією з основних мов програмування для розробки веб-додатків та динамічних веб-сайтів. Він використовується для додавання інтерактивності до веб-сторінок, валідації даних, обробки подій користувача та багато іншого [6];

– клієнт-серверна взаємодія: JavaScript може використовуватися для створення клієнтської логіки веб-додатків, що запускаються в браузері користувача, а також для взаємодії з сервером за допомогою технологій, таких як AJAX (асинхронний JavaScript і XML);

– широке використання: JavaScript використовується не тільки для веб-розробки, але й для розробки мобільних додатків (за допомогою фреймворків, таких як React Native або Ionic), настільних додатків, розширень браузера та навіть серверних застосунків (за допомогою платформи Node.js);

– динамічна мова програмування: JavaScript є динамічною мовою програмування, що означає, що типи даних визначаються автоматично під час виконання програми, а не в момент компіляції;

– легкість вивчення: JavaScript має простий синтаксис, що робить його дружнім для початківців та швидким для вивчення;

– багатий екосистема: JavaScript має велику кількість бібліотек, фреймворків та інструментів, які розширюють його можливості та полегшують розробку веб-додатків.

Загалом, JavaScript є однією з найпопулярніших та найбільш використовуваних мов програмування, яка грає ключову роль у сучасному веб-розробці та взаємодії з користувачем в інтернеті.

HTML (HyperText Markup Language) – це стандартна мова розмітки для створення веб-сторінок та веб-додатків. HTML використовується для визначення

структури та контенту веб-сторінки. Вона визначає елементи, такі як заголовки, параграфи, списки, таблиці, зображення та багато іншого:

– HTML використовує теги для визначення різних елементів та їх взаємного розташування на сторінці. Наприклад, <p> тег використовується для визначення параграфів, <h1>-<h6> теги для заголовків різного рівня, а <img> для вставки зображень [6];

– гіперпосилання: HTML дозволяє створювати гіперпосилання на інші веб-сторінки, документи або ресурси в Інтернеті за допомогою тегу <a>. Це дозволяє користувачам навігувати між сторінками;

– форми: HTML надає можливість створювати веб-форми за допомогою тегів, таких як <form>, <input>, <textarea>, <select> тощо. Форми дозволяють користувачам взаємодіяти зі сторінкою, надсилати дані та здійснювати запити на сервер;

– мета-інформація: HTML дозволяє визначати мета-інформацію про сторінку, таку як заголовок, метатеги ключових слів, опис сторінки та інше. Ця інформація використовується пошуковими системами та іншими інструментами для індексації та опису сторінок;

– сумісність із CSS та JavaScript: HTML співпрацює з CSS (Cascading Style Sheets) та JavaScript для стилізації та надання інтерактивності веб-сторінок. CSS використовується для оформлення та стилізації елементів сторінки, а JavaScript для додавання динамічного функціоналу.

HTML є одним із найважливіших і основних елементів веб-розробки і відіграє ключову роль у створенні сторінок та додатків в Інтернеті [6].

CSS (Cascading Style Sheets) – це мова стилів, яка використовується для оформлення та візуального оформлення веб-сторінок, написаних мовою HTML. Ось деякі ключові аспекти та характеристики CSS:

– стилізація елементів: CSS дозволяє задавати різні стилі та оформлення для елементів сторінки HTML, таких як текст, фон, кольори, розміри, відступи, рамки та інше;

– селектори: CSS використовує селектори для вибору конкретних елементів на сторінці, до яких будуть застосовані стилі. Селектори можуть вибирати елементи за їхнім тегом, класом, ідентифікатором або іншими атрибутами;

– каскадність і спадкування: CSS використовує принцип каскаду, що дозволяє задавати рівні пріоритету для стилів. Стилі можуть бути успадковані від батьківських елементів, а також перекривати один одного в залежності від їхнього порядку та специфічності селекторів;

– адаптивність та респонсивний дизайн: CSS дозволяє створювати адаптивні стилі, що адаптуються до різних розмірів екранів та пристроїв. Це дозволяє створювати веб-сторінки, які оптимізовані для перегляду на різних пристроях, від комп'ютерів до мобільних телефонів. CSS дозволяє створювати анімації та переходи для різних елементів на сторінці. Це може включати зміну кольорів, розмірів, положення елементів та інше;

– підтримка веб-стандартів: CSS підтримується всіма сучасними веб-браузерами і використовується для створення стандартних та сучасних веб-сайтів.

CSS є невід'ємною частиною веб-розробки і використовується разом з HTML та JavaScript для створення візуально привабливих та інтерактивних веб-сторінок.

Фреймворк в програмуванні – це структурований набір коду, який надає загальну архітектуру для розробки програмного забезпечення. Використання фреймворку дозволяє розробникам прискорити процес розробки шляхом надання готових компонентів, бібліотек та стандартів, які допомагають розв'язувати типові задачі та уникати дублювання коду.

Ось деякі популярні фреймворки у різних областях програмування:

– Django: фреймворк для розробки веб-додатків на мові програмування Python. Він надає готові рішення для роботи з базами даних, URL-адресами, шаблонами та іншими складовими веб-додатків;

– Ruby on Rails: фреймворк для швидкої розробки веб-додатків на мові програмування Ruby. Rails пропонує конвенцію перед конфігурацією та вбудовані рішення для багатьох аспектів веб-розробки;

– React.js / Vue.js / Angular: фронтенд-фреймворки для створення інтерактивних веб-інтерфейсів та односторінкових додатків [7];

– мобільна розробка React Native: фреймворк для розробки мобільних додатків на базі React та JavaScript, який дозволяє використовувати один код для розробки як для iOS, так і для Android;

– Flutter: фреймворк для розробки мобільних, веб- та настільних додатків, який використовує один код для різних платформ;

– дані та аналіз TensorFlow / PyTorch: фреймворки для розробки та навчання моделей глибокого навчання та штучного інтелекту;

– Pandas / NumPy / Matplotlib – бібліотеки для обробки, аналізу та візуалізації даних у мові програмування;

– Python, інші області, Spring (Java): Фреймворк для розробки платформозалежних додатків на мові програмування Java;

– Express.js (Node.js) – фреймворк для створення веб-додатків та API на платформі Node.js.

Це лише декілька прикладів фреймворків, які використовуються у різних областях програмування. Кожен з них має свої особливості та призначення, що дозволяє розробникам вибирати той, який найкраще відповідає їхнім потребам та завданням. Фреймворки значно спрощують розробку шляхом надання загальної архітектури, готових компонентів, бібліотек та інструментів [7].

Ось кілька типових фреймворків у сфері програмування:

– веб-фреймворки, використовуються для розробки веб-додатків та веб-сайтів. Деякі популярні веб-фреймворки включають Django та Flask для Python, Ruby on Rails для Ruby, Laravel для PHP, Express.js для Node.js і Angular та React для JavaScript;

– мобільні фреймворки, призначені для розробки мобільних додатків для платформ iOS та Android. Наприклад, для розробки додатків на iOS

використовуються фреймворки, такі як SwiftUI та UIKit для мови програмування Swift, а для Android – фреймворки, такі як Flutter для Dart або Android SDK для Java або Kotlin;

– фреймворки для веб-серверів, використовуються для розробки серверних додатків та API. Наприклад, для Python це фреймворки, такі як Django, Flask або FastAPI, а для JavaScript – Express.js;

– фреймворки для графічного інтерфейсу користувача (GUI), використовуються для розробки десктопних програм з графічним інтерфейсом. Наприклад, для мов програмування Java існують фреймворки, такі як JavaFX та Swing, а для мови програмування Python – PyQt та Tkinter;

– фреймворки для машинного навчання та штучного інтелекту, використовуються для розробки моделей машинного навчання та аналізу даних. Наприклад, для Python це фреймворки, такі як TensorFlow, PyTorch та Scikit-learn.

Ці фреймворки забезпечують розробникам готові рішення для широкого спектру завдань, що дозволяє ефективно розвивати програмне забезпечення та прискорює розробку нових проектів.

MySQL – це система управління реляційними базами даних (RDBMS), яка використовує структуровану мову запитів SQL (Structured Query Language). Вона є однією з найпопулярніших та широко використовуваних систем управління базами даних у світі. Ось деякі ключові аспекти та характеристики MySQL:

– відкритий код: MySQL є програмним забезпеченням з відкритим кодом, що означає, що ви можете змінювати, розповсюджувати та використовувати його безкоштовно згідно з ліцензією GPL (General Public License);

– реляційна модель даних: MySQL використовує реляційну модель даних, яка організовує дані у вигляді таблиць зі зв’язками між ними. Це дозволяє зберігати та керувати даними ефективно та логічно;

– масштабованість: MySQL підтримує різні рівні масштабованості, включаючи одиночні сервери, кластери баз даних та розподілені системи. Це

дозволяє використовувати MySQL для різних потреб, від малих проектів до великих корпоративних рішень;

– підтримка операційних систем: MySQL доступний для всіх основних операційних систем, включаючи Windows, Linux, macOS та інші, що робить його універсальним рішенням для розробки [7].

– широкий функціонал: MySQL має широкий спектр функціоналу, включаючи підтримку транзакцій, індексацію, безпеку даних, реплікацію, резервне копіювання та багато іншого.

Загалом, MySQL є потужною та надійною системою управління базами даних, яка використовується для різних завдань, від невеликих веб-сайтів до великих підприємственних застосунків.

Docker – це платформа для створення, розгортання та управління контейнерами. Контейнери – це невеликі та легкі середовища виконання, які містять усі необхідні для роботи програми компоненти, включаючи код, залежності, бібліотеки та налаштування. Однак вони працюють у відриві один від одного та можуть бути перенесені між різними середовищами без змін. Докер надає інструменти для створення, запуску та керування контейнерами, що дозволяє розробникам швидко та ефективно розгортати свої додатки незалежно від середовища. Він допомагає у створенні ізольованих середовищ для різних компонентів додатків, що спрощує розробку, тестування та розгортання програмного забезпечення. Завдяки Docker, розробники можуть створювати мікросервісні архітектури, де різні компоненти додатку розгортаються як окремі контейнери, що дозволяє їм бути більш гнучкими, масштабованими та надійними. Docker також допомагає у впровадженні концепції хмарного розвитку, де додатки розгортаються та працюють в різних хмарних середовищах. Gunicorn (Green Unicorn) – це сервер The Web Server Gateway Interface (WSGI) для Python. WSGI є стандартом, який визначає, як веб-додатки Python взаємодіють з веб-серверами, дозволяючи різним фреймворкам і серверам взаємодіяти між собою [1].

Gunicorn побудований таким чином, щоб забезпечити ефективну обробку запитів та відповідей від веб-додатків Python, що відповідають стандарту WSGI. Він може використовуватися з різними веб-серверами, такими як Nginx, Apache та іншими, які підтримують протокол WSGI. Один з важливих аспектів Gunicorn – це його здатність до масштабування. Він може працювати в режимі з декількома робочими процесами (workers), що дозволяє обробляти більше запитів одночасно та підвищує швидкодію додатків. Крім того, Gunicorn добре справляється з навантаженими веб-додатками та забезпечує стабільну та надійну роботу.

Загалом, Gunicorn є популярним інструментом у світі Python для розгортання веб-додатків, який допомагає забезпечити ефективну роботу та масштабованість додатків Python через стандартний інтерфейс WSGI. У секції розглядалися засоби розробки системи автоматичного розміщення камер відеоспостереження в приміщенні.

Описано програмні засоби, мови програмування, бібліотеки, мови розмітки та стилів, фреймворки та середовища розробки, вимоги до комп’ютерного програмного забезпечення та систему керування базами даних. У розділі підтверджується доцільність використання цих засобів для розробки сучасної системи веб-додатків.

Для опису програмної реалізації та функціональності системи, давайте розглянемо можливий приклад системи автоматичного розташування камер відеоспостереження у приміщеннях. Ось загальний опис програмної реалізації та її функціональності:

– опис програмної реалізації, мова програмування: система може бути реалізована за допомогою Python з використанням фреймворку Django для розробки веб-інтерфейсу та обробки бізнес-логіки;

– база даних: икористання реляційної бази даних, такої як MySQL або PostgreSQL, для зберігання інформації про приміщення, параметри камер, розташування та інше;

– веб-інтерфейс: розробка веб-інтерфейсу для користувачів, де вони можуть ввести параметри приміщення (розміри, висота стелі, тип стін і т. д.) та отримати оптимальне розташування камер відеоспостереження;

– алгоритми розташування камер: реалізація алгоритмів для автоматичного визначення оптимального розташування камер в приміщенні з урахуванням покриття, зон охоплення, обмежень приміщення та інших факторів;

– відображення результатів: виведення результатів розташування камер на веб-інтерфейсі у вигляді плану приміщення з позначками місць для встановлення камер;

– інтеграція з веб-сервером: забезпечення взаємодії з веб-сервером для забезпечення доступу користувачів до системи через веб-інтерфейс [2];

– ведення параметрів приміщення: користувач може вводити параметри приміщення, такі як розміри, висота стелі, типи стін, наявність перешкод тощо. Система аналізує введені дані та використовує їх для розрахунку оптимального розташування камер відеоспостереження;

– автоматичне розташування камер: за допомогою алгоритмів автоматичного розташування, система визначає оптимальне місце для розташування камер відеоспостереження. Результати розташування камер відображаються на веб-інтерфейсі у вигляді плану приміщення з позначками місць для встановлення камер. Користувач може вносити корективи в результати розташування камер та переглядати їх на веб-інтерфейсі;

– можливість збереження результатів аналізу та налаштувань користувача для подальшого використання. Це лише загальний опис програмної реалізації та функціональності системи автоматичного розташування камер відеоспостереження.

Конкретні деталі та реалізація можуть відрізнятися в залежності від вимог та потреб користувачів рисунок 2.1.

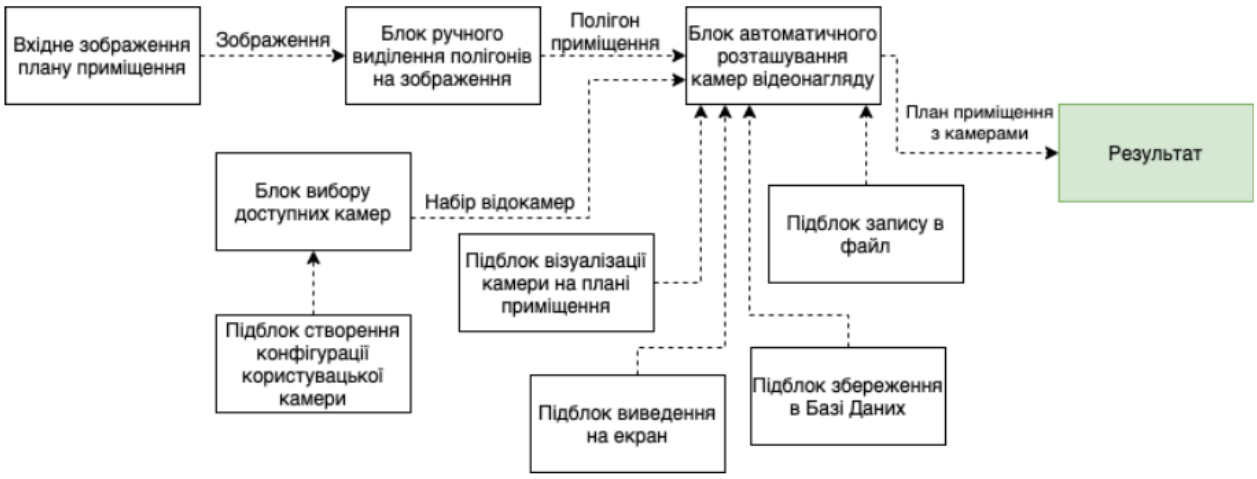

Рисунок 2.1 – Структура програмної реалізації

Опис програмної реалізації і функціональності системи зазвичай включає наступні аспекти. Пояснення загальної структури системи, включаючи компоненти, їхні взаємозв'язки та взаємодію. Опис окремих модулів або компонентів системи та їхніх функцій. Це може включати опис основних класів, методів та їх призначення. Пояснення основної функціональності системи, включаючи можливості, які вона надає користувачам або іншим системам. Взаємодія з користувачем – опис інтерфейсу користувача (якщо є) та способів взаємодії з системою, наприклад, через веб-інтерфейс, графічний інтерфейс користувача (GUI) або командний рядок.

Пояснення того, як система обробляє дані або виконує різні операції, включаючи послідовність кроків, що виконуються:

– робробка помилок і винятків: опис методів обробки помилок та винятків, які можуть виникнути під час виконання програми;

– інтеграція з іншими системами: опис взаємодії системи з іншими системами, сервісами або компонентами, якщо це необхідно;

– безпека та захист даних: пояснення заходів, прийнятих для забезпечення безпеки та захисту даних в системі [9].

Опис методів тестування системи та підтримки якості коду, наприклад, за допомогою автоматизованих тестів, контролю якості коду тощо (рис. 2.2).

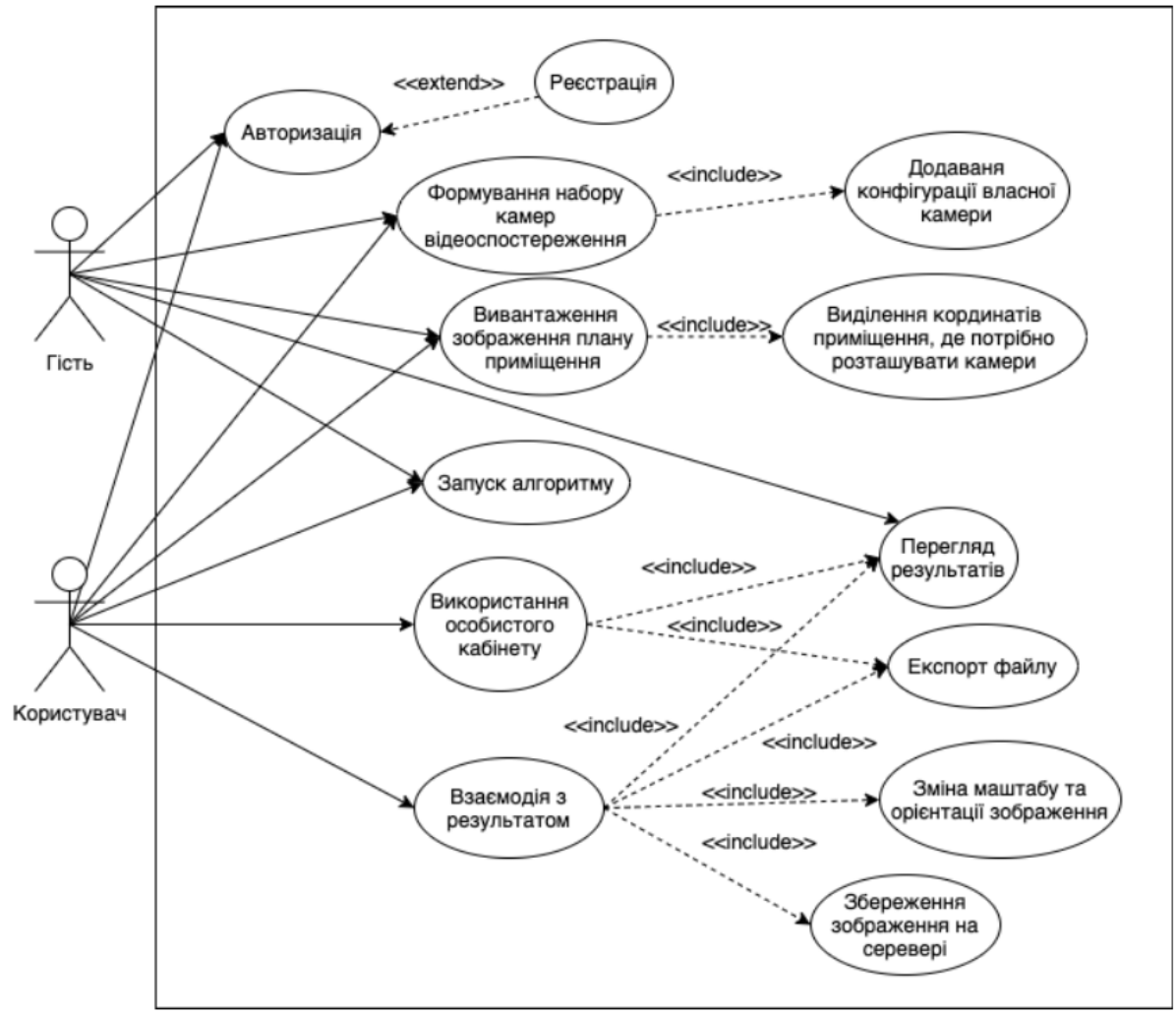

Рисунок 2.2 – Схема прецедентів

Пояснення того, як користувачі можуть отримати документацію про систему та отримати підтримку у разі необхідності. Опис програмної реалізації та функціональності системи допомагає розробникам, користувачам та іншим учасникам проекту краще зрозуміти, як працює система та як користуватися нею. UML є стандартом для візуального моделювання програмних систем. Він надає набір графічних символів та правил для створення різних типів діаграм, які дозволяють розробникам та аналітикам уявити, аналізувати та спілкуватися про складність систем.

Корпуси з міткою овальної форми. Вони представляють об’єкти або класи системи та їх властивості або характеристики (рис. 2.3).

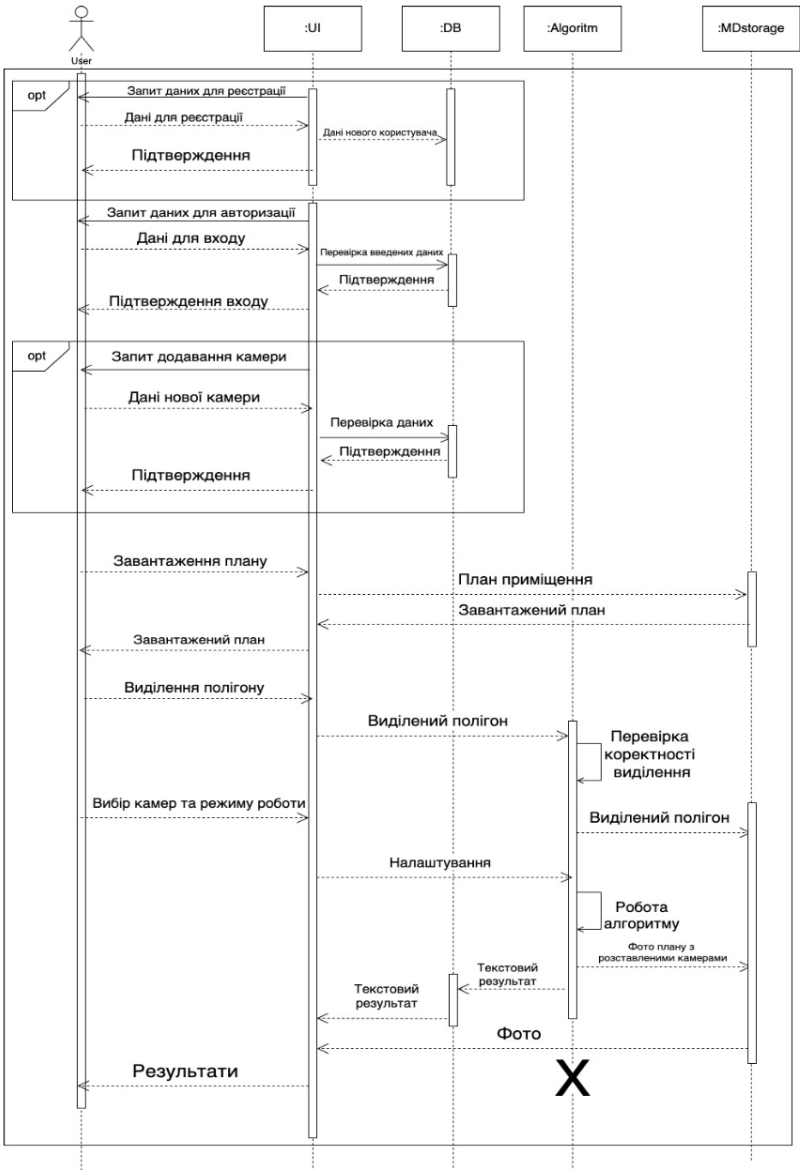

Рисунок 2.3 – Схема поетапності

Для створення такого функціоналу ви можете скористатися бібліотеками для обробки зображень та роботи з мапами, такими як OpenCV.js для обробки зображень та Leaflet.js для відображення мапи. Ось приблизний опис алгоритму. Завантажте фото плану приміщення у веб-додаток. Створіть інтерфейс для виділення полігонів на зображенні. Ви можете використовувати бібліотеку, таку як Fabric.js або Konva.js для цього. Коли користувач створить полігон на зображенні, збережіть його координати, використовуючи OpenCV.js, витягніть

область, обмежену цим полігоном, з оригінального зображення. Застосуйте алгоритми обробки зображень (наприклад, фільтрація, покращення контрастності тощо) до витягнутої області, щоб зменшити помилки та покращити якість. Повторіть цей процес для кожного полігону, який користувач створив на оригінальному зображенні. Завершивши ці кроки, ви зможете отримати поліпшені зображення з тією частиною приміщення, де користувач наміренно розміщуватиме камери рисунок 2.4.

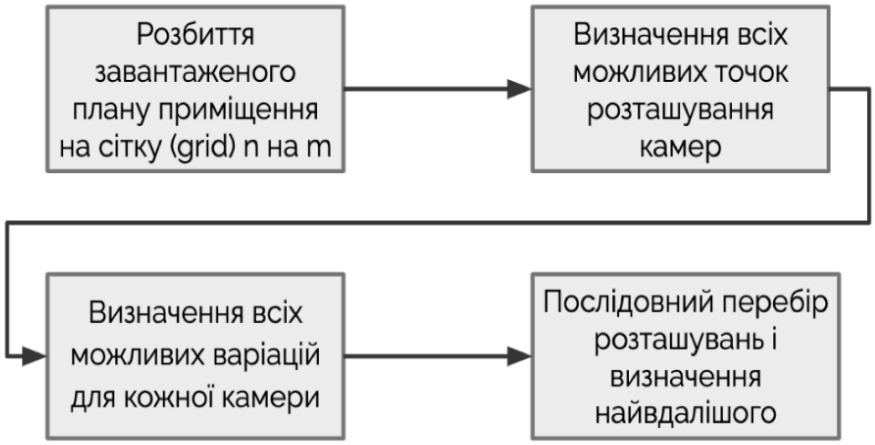

Рисунок 2.4 – Клієнт-сервер

Якщо спілкування між клієнтом і сервером відбувається за моделлю клієнт-сервер, де сервер не може ініціювати комунікацію з клієнтом, то процес обробки фото та отримання результатів відбудеться за наступною схемою. Клієнт завантажує фото плану приміщення на сервер. Сервер зберігає це фото тимчасово або надає ідентифікатор файлу клієнту. Клієнт відображає зображення та дозволяє користувачеві виділити полігони на ньому. Клієнт надсилає ці координати на сервер. Сервер отримує координати полігонів від клієнта. Сервер використовує OpenCV або інші інструменти для обробки зображення та витягає частини зображення, що відповідають цим полігонам. Після обробки сервер може зберегти ці зображення або надіслати їх назад клієнту для відображення. Це вимагає реалізації такого інтерфейсу між клієнтом та

сервером, який би дозволяв передавати дані про оброблене зображення в обидві сторони. Для цього можна використати AJAX-запити або WebSocket для взаємодії між клієнтом та сервером без прямого запиту клієнта (рис. 2.5).

Рисунок 2.5 – Архітектура обробки даних між клієнтом та сервером

MVT (Model-View-Template) – це шаблон для розробки програмного забезпечення, який використовується у веб-фреймворках, таких як Django.

Основні компоненти MVT – Модель (Model), це компонент, який відповідає за доступ до даних програми та управління ними.

Модель представляє собою структуровані дані, такі як користувачі, товари, замовлення тощо, і включає в себе методи для взаємодії з цими даними, такі як отримання, оновлення, видалення тощо.

Наприклад, в базі даних може бути таблиця користувачів, а клас моделі Django буде представляти цю таблицю і включати методи для роботи з користувачами. Перегляд (View).

Це компонент, який відповідає за логіку програми та взаємодію з користувачем. Перегляд отримує запит від користувача, обробляє його та визначає, які дані або сторінки повернути. Він може взаємодіяти з моделлю для

отримання необхідних даних та передавати їх у шаблон для відображення. Наприклад, в перегляді може бути логіка, щоб відобразити сторінку зі списком користувачів.

Шаблон (Template) – це компонент, який відповідає за відображення даних користувачу. Він містить HTML-код з вбудованими тегами шаблону, які динамічно заповнюються даними, що передаються з перегляду.

Наприклад, шаблон може містити HTML-структуру для відображення сторінки зі списком користувачів, де дані про користувачі будуть динамічно підставлятися з перегляду. У монолітній архітектурі це буде єдиний сервіс або додаток, який обробляє всі запити від клієнтів. Він може бути реалізований за допомогою будь-якої мови програмування та фреймворка, такого як Django для Python, Express.js для JavaScript тощо. У цій архітектурі модель відповідає за дані, перегляд – за логіку, а шаблон – за відображення. Це дозволяє розділити логіку програми та представлення даних, що сприяє зрозумілості коду, його підтримці та розширенню (рис. 2.6).

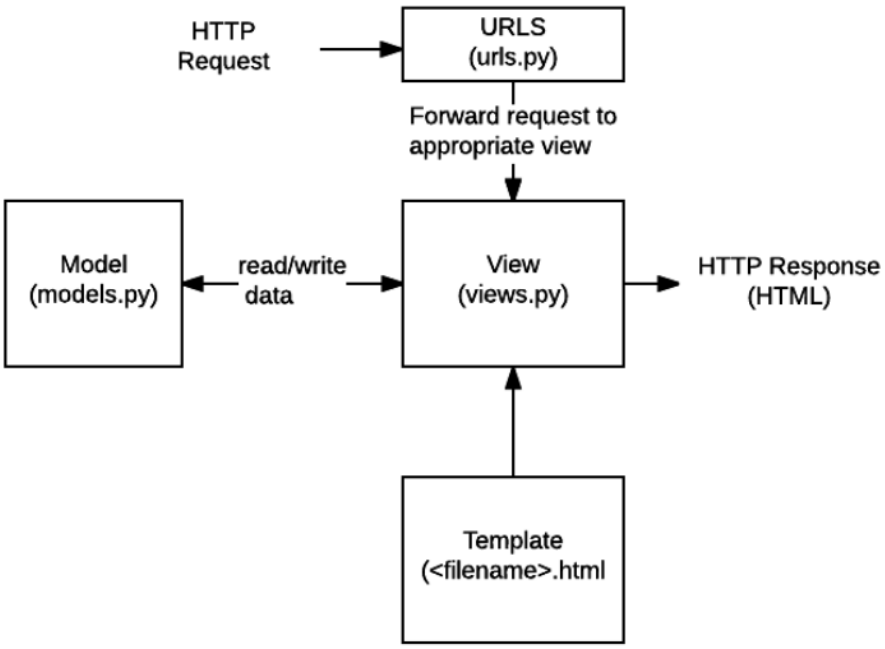

Рисунок 2.6 – Монолітний архітектурний стиль

Монолітний архітектурний стиль є одним з підходів до розробки програмного забезпечення, де весь код програми розглядається як один великий блок, який монолітний за своєю природою. Цей стиль розробки дозволяє швидко створювати прототипи та розвивати програмне забезпечення, а також спрощує управління кодом та деплоїться. В рамках монолітного підходу можна створити фреймворки або бібліотеки, які допоможуть створити web-систему, що базується на клієнт-серверній архітектурі з підтримкою SPA (Single Page Application). Для побудови такої системи можна розглядати наступні ключові компоненти.

Для створення SPA можна використовувати популярні фреймворки, такі як Angular, React або Vue.js. Вони дозволяють створювати динамічні веб-сторінки, які взаємодіють з сервером через API. API для взаємодії. Щоб забезпечити спілкування між клієнтом та сервером, необхідно встановити API. Це може бути RESTful API або GraphQL, залежно від ваших вимог та вподобань. Бізнес-логіка та база даних – у монолітному підході бізнес-логіка зазвичай розміщується на сервері. Вона включає в себе усі правила та операції, які обробляють дані. База даних також може бути частиною серверної частини, і для неї може бути використана будь-яка реляційна або нереляційна база даних [10].

Ці компоненти дозволять створити веб-систему, яка підтримує як повністю оновлення сторінки, так і часткове оновлення без перезавантаження сторінки, що є основними принципами SPA (рис. 2.7).

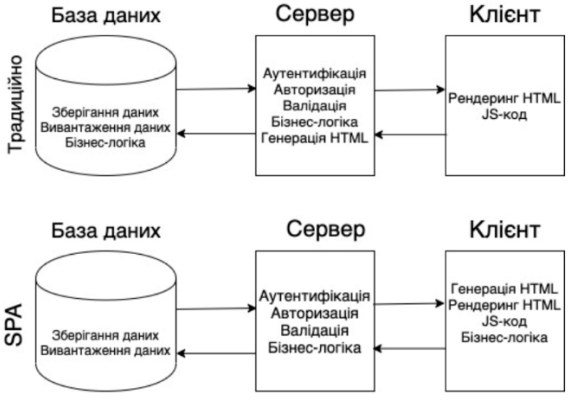

Рисунок 2.7 – Програмна реалізація системи автоматичного позиціонування камер відеоспостереження

Розглянуто програмну реалізацію системи автоматичного позиціонування камер відеоспостереження. Структура системи, приклад вирішення задач, архітектура програми, послідовність роботи алгоритму, діаграма прецедентів і послідовностей.

РОЗДІЛ 3
РОЗРОБКА СИСТЕМИ АВТОМАТИЧНОГО РОЗТАШУВАННЯ КАМЕР ВІДЕОСПОСТЕРЕЖЕННЯ У ПРИМІЩЕННІ

3.1 Практична реалізація об’єкта проектування

Для реалізації авторизації користувача з логіном та паролем, а також створення облікового запису на сайті, нам знадобиться функціональність для обробки введених даних користувача і взаємодії з базою даних. Ось можливий алгоритм дій.

Створення форми входу (Login Form), авторизація користувача (рис. 3.1).

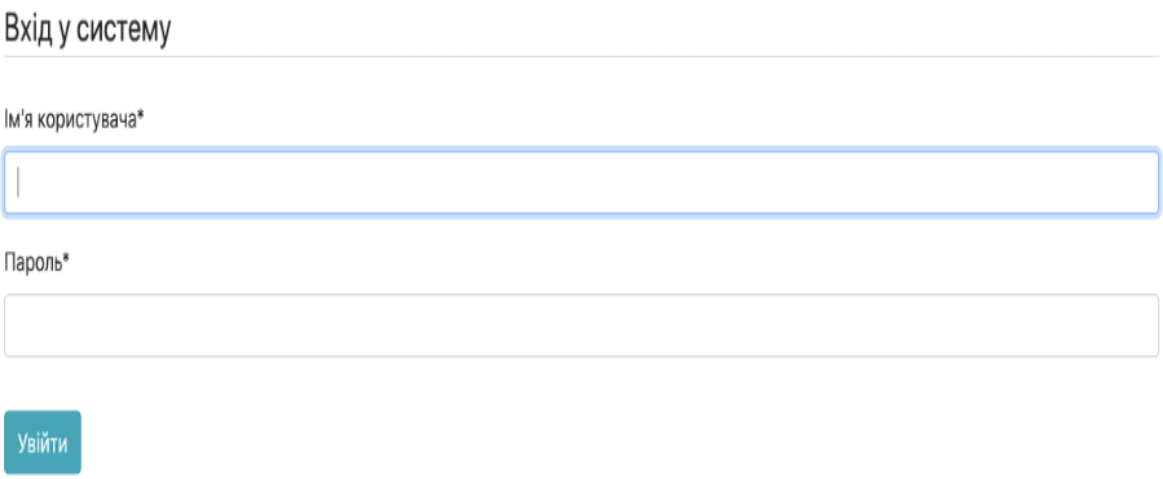

Рисунок 3.1 – Авторизація користувача у систему

При отриманні даних форми входу, перевіряємо, чи існує користувач з таким логіном та паролем у базі даних. Якщо дані введено правильно, виконуємо вхід користувача до системи. Використовуємо сесії або токени для зберігання ідентифікатора сеансу аутентифікації. Створення форми реєстрації (Registration Form) – створюємо веб-форму для реєстрації нового користувача, де потрібно ввести ім’я, електронну пошту та пароль (рис. 3.2).

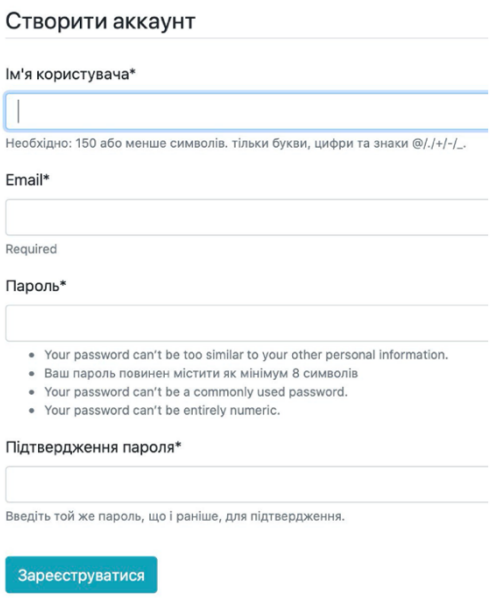

Рисунок 3.2 – Реєстрація користувача

Перед додаванням нового користувача потрібно перевірити, чи не існує вже користувач з такою самою електронною поштою. Зашифруємо пароль перед зберіганням у базі даних. Після валідації та перевірки унікальності даних, зберігається інформацію про нового користувача у базі даних.

Додамо можливість надсилати електронний лист для підтвердження реєстрації або надсилати повідомлення з підтвердженням на сторінку. Додаємо обробку помилок для випадків, коли введені дані неправильні або користувач з таким логіном вже існує. У разі якщо користувач забув дані для авторизації, було добавлено функцію відновлення паролю за допомогою листа на електронну пошту (рис. 3.3).

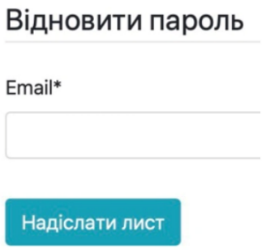

Рисунок 3.3 – Відновлення паролю за допомогою електронного листа

На введену пошту приходить лист, за допомогою якого користувач може відновити пароль для авторизації в системі (рис. 3.4).

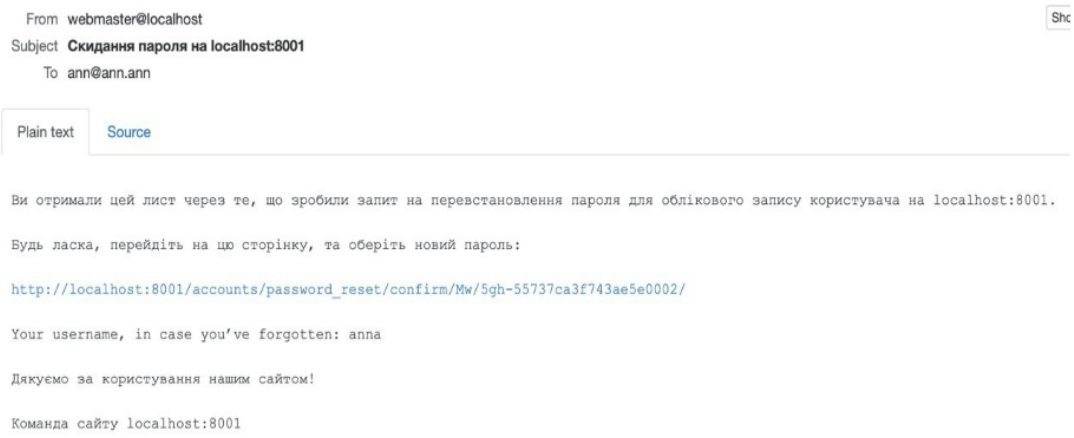

Рисунок 3.4 – Лист для відновлення паролю

Для розуміння роботи програми було додано загальну характеристику, щоб користувачу було більш зрозуміло, яку саме роль відіграє та які функції виконує система (рис. 3.5).

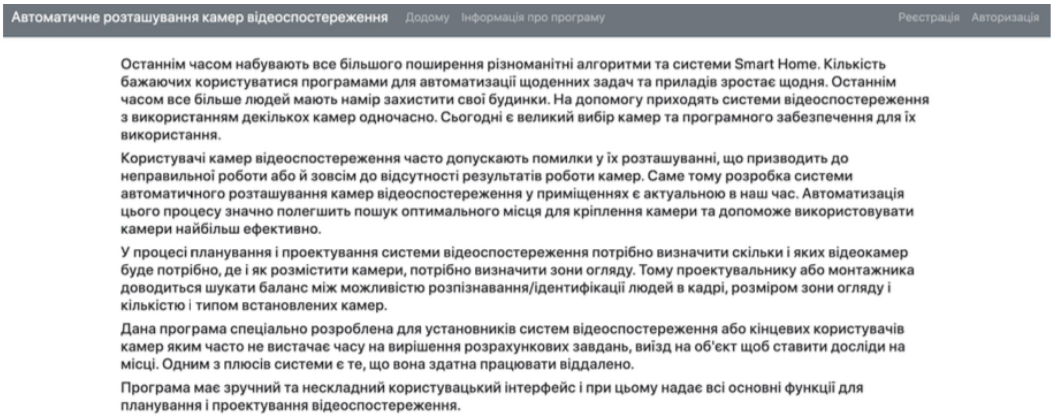

Рисунок 3.5 – Загальна характеристика системи

Було додано функцію перегляду наявних камер відеоспостереження (рис. 3.6), для того щоб користувач міг обрати камеру, яка відповідає його вимогам. Користувач може передивитися набір характеристик та ціну, щоб вибрати найбільш оптимальний варіант (рис. 3.7).

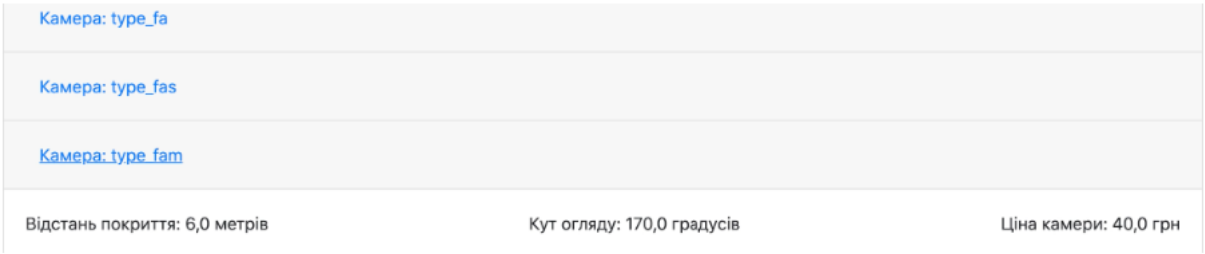

Рисунок 3.6 – Наявність камер відеоспостереження

Рисунок 3.7 – Характеристика камер

Користувач може ввести розміри приміщення з яким працює (рис. 3.8) і тоді програма сама запропонує перелік камер, які будуть найбільш оптимальними (рис. 3.9).

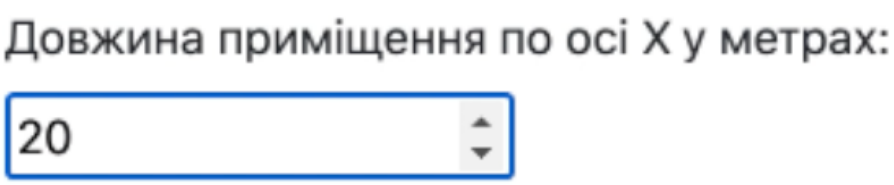

Рисунок 3.8 – Довжина приміщення

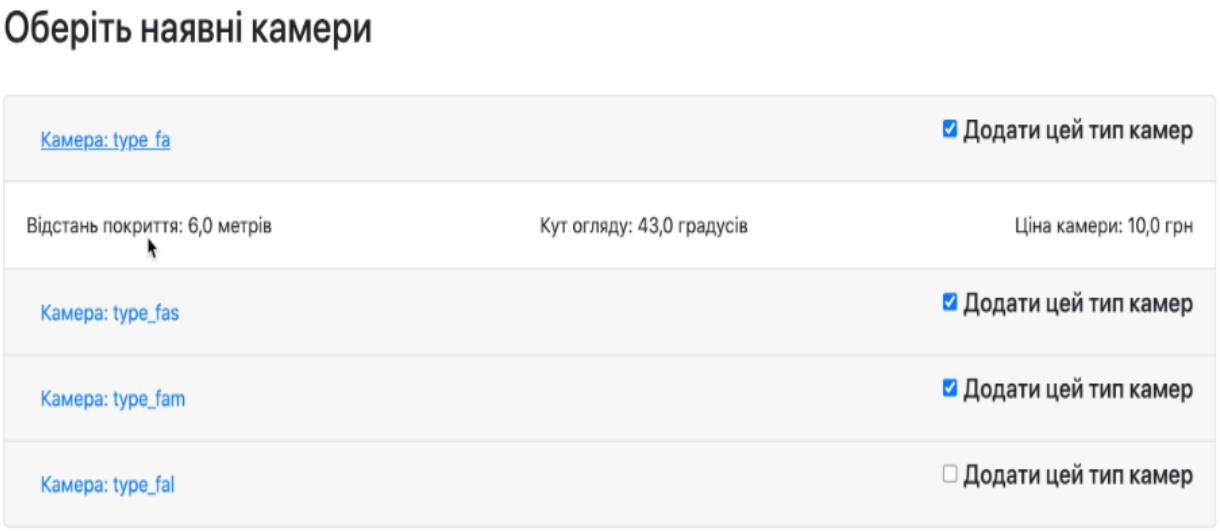

Рисунок 3.9 – Наявні камери

Користувач може обрати мету встановлення камер спираючись на максимальне покриття з мінімальною кількостю камер або спираючись на грошові витрати (рис. 3.10).

Рисунок 3.10 – Вибір камер

Додано вхідний план приміщення, щоб користувач міг додати перешкоди огляду, такі як стіни, меблі тощо. Це допоможе більш оптимально покрити приміщення камерами відеоспостеження оминаючи сліпі зони (рис. 3.11).

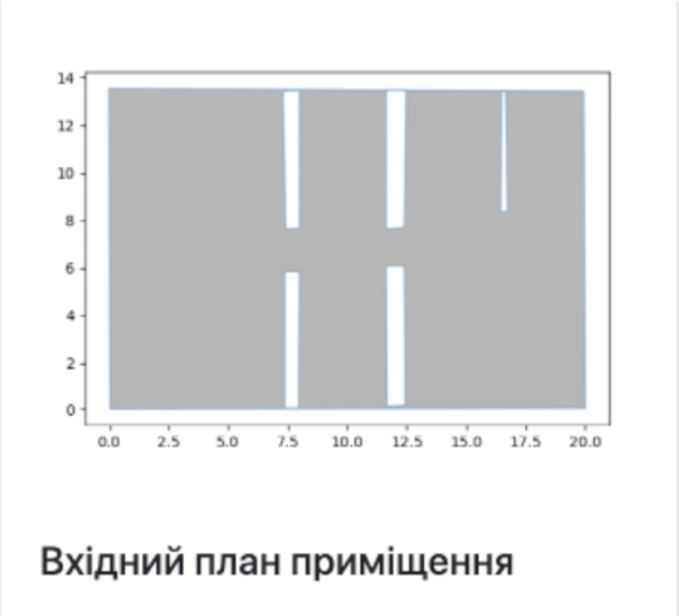

Рисунок 3.11 – Вхідний план приміщення

Додано вивід результатів роботи системи (рис.3.12), щоб користувач міг зрозуміти скільки камер і коштів потрібно для встановлення камер відеоспостереження а також можливість наглядно побачити розсташування камер у приміщенні, параметри якого були попередньо введені (рис 3.13).

Результати роботи системи автоматичного розміщення обраних камер відеоспостереження у заданому приміщенні

Тип камери	Вартість однієї камери	Кількість камер даного типу	Сума вартості камер даного типу
type_fas	45,0 грн	5	225,0 грн
type_fa	10,0 грн	4	40,0 грн

Рисунок 3.12 – Результат роботи системи

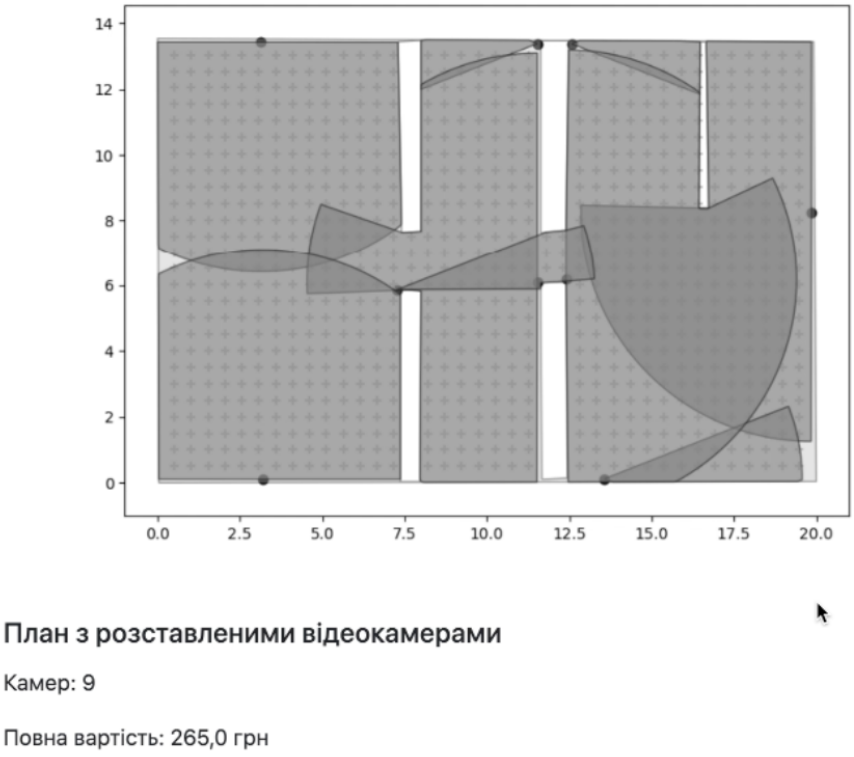

Рисунок 3.13 – План розставлених камер

3.2 Тестування та налагодження інформаційно-комп’ютерної системи

Основна мета QA – забезпечити, що програмний продукт відповідає вимогам замовника і відповідає встановленим стандартам якості. Ось деякі ключові аспекти забезпечення якості.

Визначення стандартів якості, яким повинен відповідати програмний продукт. Ці стандарти можуть включати в себе вимоги до функціональності, безпеки, ефективності, надійності та інших аспектів продукту. Проведення різних видів тестування, таких як функціональне тестування, тестування продуктивності, тестування безпеки тощо, для перевірки, чи відповідає продукт вимогам і стандартам якості. Внутрішній контроль якості. Реалізація процесів контролю якості під час розробки програмного продукту. Це включає в себе перевірку коду, рев’ю коду, автоматизовані тестування тощо. Управління конфігурацією – систематичний підхід до керування змінами в програмному забезпеченні. Забезпечення цілісності і стабільності продукту під час змін.

Постійне вдосконалення – проведення аналізу результатів тестування та інших процесів QA з метою постійного вдосконалення якості продукту та оптимізації процесів розробки. Комунікація зі замовником – забезпечення взаєморозуміння замовника щодо його вимог до продукту та регулярне звітування про хід розробки та якість продукту. Забезпечення якості є важливою складовою розробки програмного забезпечення і допомагає забезпечити успішну поставку якісного продукту замовнику (рис. 3.14).

Рисунок 3.14 – Етапи розробки

Забезпечення якості (Quality Assurance – QA) є важливою складовою процесу розробки програмного забезпечення. В основі цього процесу лежить бажання забезпечити, щоб програмний продукт відповідав потребам та очікуванням замовника і був відповідною якості. Основні принципи та завдання QA включають:

– створення та виконання тестів: розробники та QA-інженери створюють тестові сценарії для перевірки функціональності, відповідності вимогам, надійності та продуктивності програмного забезпечення;

– виявлення та виправлення дефектів: QA-інженери виявляють та документують будь-які дефекти у програмному забезпеченні, а розробники виправляють їх;

– стандарти якості: QA визначає стандарти якості, які повинні бути виконані усіма учасниками процесу розробки. Ці стандарти можуть включати в себе якісні метрики, процедури тестування, а також процедури оцінки та підтримки якості;

– автоматизація тестування: для поліпшення ефективності тестування та зменшення людських помилок, може бути використана автоматизація тестування, наприклад, за допомогою інструментів для автоматизації тестування;

– вдосконалення процесів розробки: QA допомагає вдосконалити процеси розробки, шляхом ідентифікації та вирішення проблем, які виникають під час розробки, тестування та випуску продукту;

– навчання та підтримка: QA забезпечує навчання розробників та інших учасників команди щодо стандартів якості та процедур тестування, а також надає підтримку в процесі випуску продукту.

Загалом, QA допомагає забезпечити, що програмне забезпечення відповідає вимогам замовника, є надійним та відповідає встановленим стандартам якості (рис. 3.15).

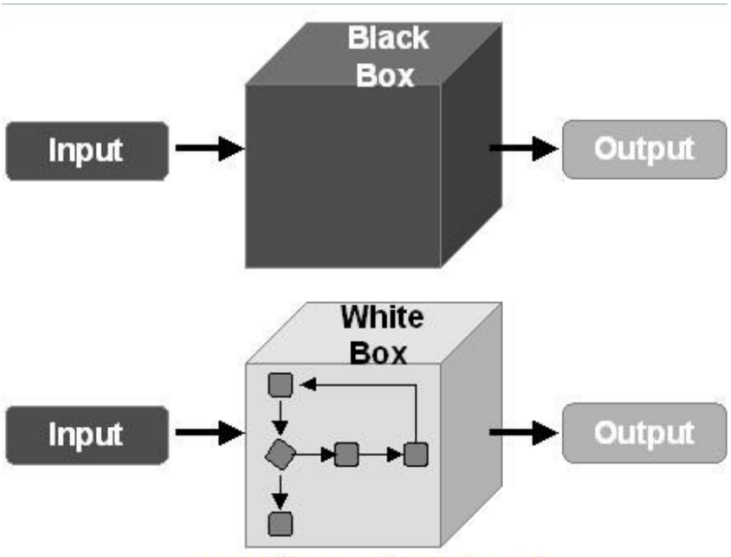

Рисунок 3.15 – Black Box Testing та White Box testing [11]

Забезпечення якості (Quality Assurance, QA) як діяльність, спрямовану на забезпечення того, щоб програмний продукт або послуга, що надається замовникам, були найкращої якості. QA зосереджується на поліпшенні процесів,

щоб забезпечити доставку якісних продуктів, що відповідають встановленим стандартам якості. Ось деякі ключові аспекти забезпечення якості:

– визначення вимог якості: QA починається з чіткого визначення вимог до програмного продукту або послуги. Це може включати функціональні, продуктивнісні, безпекові та інші вимоги;

– виконання тестів: QA забезпечує виконання тестів для перевірки функціональності, надійності, продуктивності та інших аспектів програмного продукту. Це може включати ручне тестування, автоматизоване тестування та інші методи;

– виявлення та виправлення дефектів. QA відстежує та документує виявлені дефекти в програмному продукті і відповідає за їх виправлення до випуску продукту;

– стандарти якості та процеси: QA встановлює стандарти якості та процеси, які дозволяють забезпечити високу якість продукту та ефективність процесів розробки;

– неперервне поліпшення: QA забезпечує постійне навчання та вдосконалення процесів, щоб підвищити якість продукту та ефективність розробки. Загалом, забезпечення якості відіграє ключову роль у забезпеченні успішного випуску програмного продукту та задоволення потреб клієнтів.

Функціональне та користувацьке тестування проведено успішно. Помилки було знайдено та виправлено. Проведено тестування налагоджень, що показало коректність роботи системи. Продукт функціонує правильно і готовий до використання.

3.3 Розробка документації для програмого продукту

Документації програмного застосунку є важливим інструментом для розробників та тестувальників. Документаціє надає інформацію про функціональність, архітектуру та інші аспекти програмного застосунку

Django має широкий функціонал завдяки якому ми маємо можливість реалізувати велику кількість утиліт.

Фреймворк REST містить вбудовану підтримку для створення схем OpenAPI, які можна використати з інструментами, які дозволяють створювати документацію API.

Налаштування віртуального середовища, встановлення залежностей та створення локальної документації зображено в лістингу 3.1.

Лістинг 3.1 – Команди для налаштування командного середовища

```
$ git clone https://github.com/django/django.git
$ python -m venv .venv
$ source .venv/bin/activate
$ python -m pip install -r docs/requirements.txt
$ cd docs
$ make html
```

Кінець лістингу 3.1

Для створення першої моделі бази даних, необхідно імпортувати один з модулів бібліотеки django.db, та ініціалізувати необхідний клас (лістинг 3.2).

Лістинг 3.2 – Приклад створення моделі бази даних

```
from django.db import models
class Reporter(models.Model):
    full_name = models.CharField(max_length=70)
    def __str__(self):
        return self.full_name
class Article(models.Model):
    headline = models.CharField(max_length=200)
    reporter = models.ForeignKey(Reporter, on_delete=models.CASCADE)
    def __str__(self):
        return self.headline
```

Кінець лістингу 3.2

Для роботи з базою даних, можна скористатися графічним інтерфейсом користувача, зображеним на рисунку 3.16.

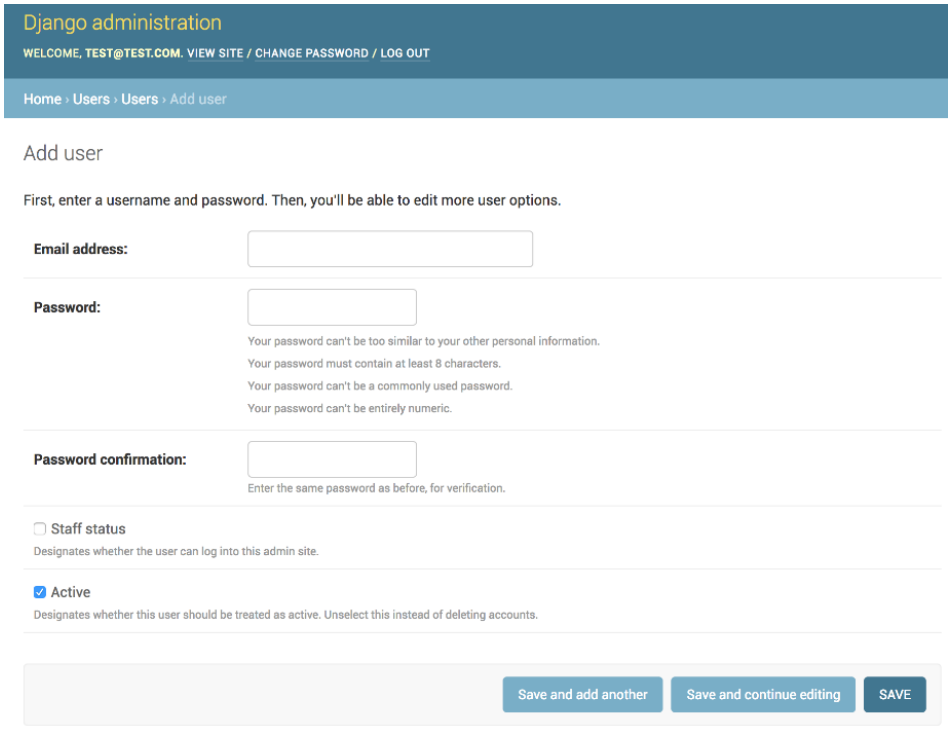

Рисунок 3.16 – Графічний інтерфейс користувача

ВИСНОВКИ

Робота над системою автоматичного розташування камер відеоспостереження у приміщенні дозволила досягти важливих результатів і сформувати ряд висновків, що мають практичне і теоретичне значення.

Здійснено аналіз існуючих аналогів розміщення камер в приміщенні, таких як CAD5D та IP Video System Design Tool, що дало зрозуміти які функції вони виконують, а які ні.

Створено математичну модель приміщення для аналізу можливих розташувань камер. Використання передових технологій, таких як машинне навчання та комп’ютерний зір, сприяє створенню інноваційних рішень для відеоспостереження. Впровадження нових алгоритмів дозволяє постійно вдосконалювати систему та адаптуватися до нових умов експлуатації.

Розроблені алгоритми автоматизованого розташування камер забезпечують оптимальне покриття простору, мінімізуючи сліпі зони і покращуючи загальну ефективність системи відеоспостереження. Використання технологій машинного навчання та комп'ютерного зору дозволяє точно аналізувати просторові характеристики приміщень і адаптуватися до змін у конфігурації простору. Автоматизація процесу планування розташування камер значно знижує витрати на проектування та встановлення системи відеоспостереження. Оптимізація використання обладнання забезпечує ефективне розташування меншої кількості камер, що зменшує витрати на придбання та обслуговування обладнання.

Створено програмне забезпечення для автоматичного розташування камер. Розроблена система демонструє високу гнучкість і може легко адаптуватися до змін у конфігурації приміщень та вимог безпеки. Система здатна масштабуватися для використання в різних типах і розмірах приміщень, від невеликих офісів до великих промислових об'єктів. Ефективне розташування камер підвищує здатність системи виявляти та реагувати на інциденти, знижуючи ризики пропуску важливих подій. Система відеоспостереження з

оптимальним розташуванням камер служить стримуючим фактором для потенційних злочинців.

Забезпечено інтуїтивно зрозумілий інтерфейс для користувачів системи автоматичного розташування камер відеоспостереження у приміщенні представляє собою значний крок вперед у сфері безпеки та моніторингу. Вона забезпечує підвищення ефективності відеоспостереження, економію ресурсів, гнучкість та адаптивність до змінних умов, а також покращує загальний рівень безпеки об'єктів різного типу і розміру. Автоматизація процесу розташування камер, інтеграція з існуючими системами та використання передових технологій роблять цю систему важливим інструментом для забезпечення безпеки в сучасному світі.

Проведено тестування системи для підтвердження правильності роботи, забезпечення високої якості та надійності програмного продукту.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Lakhey M. Word2vec Made Easy. URL: https://towardsdatascience.com/word2vec-made-easy-139a31a4b8ae (дата звернення: 16.04.2024).

2. Sharma A. Understanding Activation Functions in Neural Networks. URL: https://medium.com/the-theory-of-everything/understanding-activation- functions-in-neural-networks-9491262884e0 (дата звернення: 16.04.2024).

3. Karani D. Introduction to Word Embedding and Word2Vec. URL: https://towardsdatascience.com/introduction-to-word-embedding-and-word2vec-652d0c2060fa (дата звернення: 22.04.2024).

4. How is GloVe different from word2vec? URL: https://www .quora.com/How-is-GloVe-different-from-word2vec (дата звернення: 27.04.2024).

5. What are the advantages and disadvantages of Word2vec and GloVe? URL: https://www.quora.com/What-are-the-advantages-and-disadvantages-of-Word2vec-and-GloVe (дата звернення: 27.04.2024).

6. Allam A., Nagy M., Thoma G., Krauthammer M. Neural networks versus Logistic regression for 30 days all-cause readmission prediction. URL: https://arxiv.org/pdf/1812.09549.pdf (дата звернення: 01.05.2024).

7. Mohammadi M., Mundra R., Socher R. CS 224D: Deep Learning for NLP. URL: https://cs224d.stanford.edu/lecture_notes/notes4.pdf (дата звернення: 06.05.2024).

8. Woodford C. How neural networks work. URL: https://www.explainthatstuff.com/introduction-to-neural-networks.html (дата звернення 16.05.2024).

9. What is TensorFlow? Introduction, Architecture & Example. URL: https://www.guru99.com/what-is-tensorflow.html (дата звернення: 16.05.2024).

10. Get to know TensorFlow.js in 7 minutes. URL: https://www.freecodecamp.org/news/get-to-know-tensorflow-js-in-7-minutes-afcd0dfd3d2f/ (дата звернення: 16.05.2024).

11. What is White Box Testing? URL: https://instatus.com/blog/white-box-testing (дата звернення 17.05.2024).