МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра економічної кібернетики та інформатики

Вебер Владислав Денисович

Розробка та аналіз системи інтелектуального відеоспостереження для підвищення безпеки господарського об’єкту

спеціальність 124 «Системний аналіз»

освітня програма – «Системний аналіз»

Кваліфікаційна робота

Виконав студент групи САм-21

Вебер. В.Д. _____

Науковий керівник:

к.т.н., ст.викл. Адамів О.П. _____

ТЕРНОПІЛЬ-2024

ЗМІСТ

ВСТУП **5**

1 АНАЛІТИЧНИЙ ДОГЛЯД **8**

1.1 Опис програмного середовища 8

1.2 Аналіз існуючих аналогів 12

РОЗДІЛ 2 АНАЛІЗ ЗАВДАННЯ НА ДИПЛОМНИЙ ПРОЄКТ **21**

2.1 Аналіз методів 21

2.2 Методи застосування 24

3 ОСНОВНА ЧАСТИНА ДИПЛОМНОГО ПРОЄКТУ **28**

3.1 Вибір програмного забезпечення 28

3.2 Проєктування власного додатку 40

3.3 Інструкції користувача 64

ВИСНОВКИ **71**

ДОДАТКИ

ВСТУП

Актуальність теми. У сучасному світі безпека є однією з найважливіших складових функціонування господарських об’єктів. Зростання рівня технологічного прогресу надає нові можливості для покращення засобів відеоспостереження, зокрема за рахунок впровадження інтелектуальних систем. Ці системи дозволяють здійснювати моніторинг у режимі реального часу, аналізувати події, автоматизувати процеси та забезпечувати ефективну взаємодію користувачів із системою. У даній магістерській роботі здійснено розробку та аналіз інтелектуальної системи відеоспостереження для підвищення безпеки господарського об’єкта. Зважаючи на постійне зростання потреб у безпеці та динамічний розвиток технологій, тема розробки інтелектуальних систем відеоспостереження є актуальною. Запропоноване рішення дозволить покращити рівень захисту господарських об’єктів, забезпечити більш раціональне використання ресурсів та скоротити необхідність людського втручання.

Мета дослідження. Метою даної роботи є розробка інтелектуальної системи відеоспостереження, яка забезпечує підвищення рівня безпеки господарського об’єкта за рахунок автоматизації обробки відеоінформації, інтеграції сучасних алгоритмів відеоаналітики та оптимізації використання ресурсів системи.

Завдання дослідження для досягнення поставленої мети було визначено наступні завдання:

- Провести аналіз сучасних підходів до створення інтелектуальних систем відеоспостереження.
- Здійснити порівняльний огляд існуючих систем відеоаналітики та їх застосування у забезпеченні безпеки господарських об’єктів.
- Обґрунтувати економічну доцільність впровадження інтелектуальної системи відеоспостереження.

— Сформулювати вимоги до функціональних і нефункціональних компонентів системи.
— Розробити архітектурну модель системи, обрати відповідний інструментарій для її реалізації.
— Реалізувати програмне забезпечення для інтелектуального відеоспостереження.
— Провести тестування системи та оцінити її ефективність за визначеними критеріями.

Методи роботи. Для виконання завдань дослідження застосовано такі методи наукової роботи:

— Аналіз літературних джерел та інформаційних ресурсів — для вивчення сучасних підходів до розробки інтелектуальних систем відеоспостереження та порівняння існуючих рішень.
— Метод порівняльного аналізу — для оцінки функціональних можливостей аналогічних систем та визначення їх переваг і недоліків.
— Системний підхід — для розробки архітектурної моделі системи, що забезпечує інтеграцію функціональних і нефункціональних компонентів.
— Методи моделювання та проєктування — для створення прототипу інтелектуальної системи відеоспостереження та її компонентів.
— Експериментальний метод — для тестування розробленої системи в умовах, максимально наближених до реальних.
— Методи економічного аналізу — для обґрунтування доцільності впровадження розробленої системи.

Структура роботи. Магістерська робота складається з вступу, трьох розділів, висновків, списку використаних джерел та додатків.

— Перший розділ містить аналітичний огляд сучасних підходів до інтелектуального відеоспостереження, порівняльний аналіз існуючих систем відеоаналітики, а також техніко-економічне обґрунтування доцільності впровадження проєкту.
— Другий розділ присвячений постановці задачі, визначенню вимог до системи, критеріям оцінювання її ефективності та очікуваним результатам.

— Третій розділ охоплює процес проєктування та розробки системи, включаючи вибір архітектури, реалізацію компонентів, інструкцію користувача та оцінку функціональності.

У висновках узагальнено основні результати роботи, сформульовано рекомендації щодо впровадження розробленої системи.

Новизна роботи. Наукова новизна даної роботи полягає у наступному:

— Розроблено інтегровану архітектурну модель інтелектуальної системи відеоспостереження, яка поєднує алгоритми відеоаналітики з можливістю масштабування системи.

— Запропоновано використання сучасного програмного забезпечення для аналізу відеоінформації з метою автоматизації процесів обробки даних.

— Впроваджено функції автоматичного розпізнавання предметів і поведінкових аномалій, що підвищує оперативність реагування на загрози.

— Розроблено ефективний алгоритм оптимізації використання апаратних ресурсів системи без втрати якості роботи.

Практичне значення роботи. Практичне значення роботи полягає у створенні функціонального прототипу інтелектуальної системи відеоспостереження, який може бути використаний для підвищення безпеки в різних сферах: навчальних закладах, виробничих підприємствах, торгових центрах та інших об’єктах. Розроблена система дозволяє здійснювати моніторинг у реальному часі, ідентифікувати загрози, автоматизувати збирання та обробку відеоданих, що знижує ризик людської помилки та підвищує ефективність безпекових заходів. Крім того, запропоновані технічні рішення можуть бути адаптовані для інтеграції з існуючими системами безпеки та масштабування залежно від потреб об'єкта.

1 АНАЛІТИЧНИЙ ДОГЛЯД

1.1 Опис програмного середовища

Очевидно, що в часи стрімкого розвитку науки та технологій, відеопотік та відеоматеріали самі по собі не є ефективним засобом забезпечення публічного порядку, адже в разі необхідності проаналізувати певну подію чи знайти ту чи іншу інформацію, необхідно передивлятися десятки годин відеоматеріалу. Так само, постійне спостереження за відеопотоком не є ефективним витрачанням часу.

У низці відвіданих міст існують окремі посади для операторів, функцією яких є відслідковування та реагування в режимі реального часу. Така робота може бути корисною для оперативного реагування на сигнали системи, але немає потреби в залученні більше однієї штатної одиниці одночасно.

Також більш доцільним є запровадження посади оператора не в структурі органів місцевого самоврядування, а все ж таки в органах Національної поліції, які обслуговують відповідну територію. Зважаючи на це, в багатьох системах відеоспостереження використовується відеоаналітика, яка вирішує найрізноманітніші задачі.

Наразі найпростішою відеоаналітикою, яку може здійснювати система відеоспостереження, є автоматичне розпізнавання державних номерних знаків ТЗ.

Втім, сучасні програмні рішення дозволяють також відслідковувати перетин певних ліній, знаходження об’єктів протягом тривалого часу на визначеній території (детектор залишених предметів), розпізнавати обличчя та навіть предмети з визначеними характеристиками (наприклад, предметом пошуку може бути червона дамська сумочка).

Однією з цікавих функцій, яка зокрема документується щодо системи відеоспостереження, — можливість розпізнавання скупчень людей. Це дозволяє оперативно виявляти надзвичайні події або спонтанні мирні зібрання та належним чином реагувати на них.

Для цього використовуються різноманітні програмні засоби — найчастіше це стандартне програмне забезпечення виробників камер, зокрема HikVision.

Також широко використовуються програмні платформи ULA від одеської компанії LanTec, Milestone від однойменної данської компанії та VEZHA від вінницької компанії Incoresoft.

Додаток призначений для роботи з такими пристроями: цифровими і мережевими відеореєстраторами, відеодомофонами і панелями управління безпекою. Завдяки йому можна спостерігати за об'єктом, який цікавить в реальному часі або переглядати відеозапис в будь-який час і будь-якому зручному місці - вдома, на роботі або в майстерні. Додаток відразу ж направить вам повідомлення при спрацьовуванні сигналу тривоги.

Програма відеореєстрації дозволяє виконувати:

1) Можна проводити контроль над територією в режимі реального часу.
2) Користувач може управлятися камерами відеоспостереження PTZ.
3) Є можливість відтворення відео з архіву.
4) Двоканальний аудиодомофон для двостороннього зв'язку.
5) Користувач отримує на сервіс все оповіщення про тривогу з відео і фото.
6) Можна відповідати на виклики від відеодомофонів і дверних дзвінків.
7) Замінює дистанційну панель управління.
8) Передбачено відбиток пальця для ідентифікації користувача.

Програмне середовище є одним із ключових елементів у розробці будь-якої інтелектуальної системи. Його вибір визначає функціональні можливості, продуктивність, масштабованість та надійність розробленого програмного забезпечення. У рамках даного проєкту було розроблено систему інтелектуального відеоспостереження, яка потребує сучасного програмного середовища для реалізації її функціоналу.

Загальна характеристика програмного середовища

Для розробки системи відеоспостереження було обрано середовище Microsoft Visual Studio, яке є однією з найпопулярніших інтегрованих платформ для створення програмного забезпечення. Visual Studio забезпечує широкий спектр

інструментів для написання, тестування та налагодження коду, що дозволяє оптимізувати процес розробки.

Visual Studio підтримує різні мови програмування, включаючи C#, який було обрано для реалізації проєкту. Ця мова програмування дозволяє ефективно працювати з великими обсягами даних, інтегрувати сторонні бібліотеки та створювати зручний інтерфейс користувача.

Компоненти середовища розробки

1. Редактор коду:
   - Забезпечує зручність написання коду завдяки підсвічуванню синтаксису, автодоповненню та вбудованим підказкам.
   - Інтеграція з системами контролю версій (Git, SVN).
2. Налагоджувач:
   - Дозволяє виявляти та виправляти помилки у коді за допомогою покрокового виконання, встановлення точок зупинки (breakpoints) та моніторингу змінних.
3. Інструменти для тестування:
   - Включають модульне тестування, автоматизовані тести та інструменти для аналізу продуктивності коду.
4. Пакетний менеджер NuGet:
   - Забезпечує доступ до великої кількості бібліотек та модулів, які можна інтегрувати у проєкт для розширення його функціональних можливостей.
5. Засоби створення графічного інтерфейсу:
   - Windows Forms та WPF використовувалися для створення користувацького інтерфейсу системи.

Використані бібліотеки та інструменти.

Для реалізації системи відеоспостереження були використані такі бібліотеки:

- Emgu CV – бібліотека для роботи з комп’ютерним зором, яка дозволяє виконувати розпізнавання облич та об’єктів.
- DirectShowLib – використовується для обробки відеопотоків та роботи з медіа-даними в реальному часі.

— SQL Server – застосовується для зберігання інформації про події, які фіксуються системою відеоспостереження.

Технічні вимоги розробки Microsoft Visual Studio потребує наступних ресурсів:

— Операційна система: Windows 10 або вище.
— Оперативна пам’ять: мінімум 8 ГБ (рекомендовано 16 ГБ для великих проєктів).
— Процесор: багатоядерний процесор із тактовою частотою не менше 2 ГГц.
— Місце на диску: 20–30 ГБ для встановлення та роботи із залежностями.
— Графічна карта: підтримка DirectX 11 або вище.

Процес розробки здійснювалася в кілька етапів:

— Визначення вимог: збір інформації про потреби користувачів та функціональні вимоги до системи.
— Проєктування: створення архітектури системи, яка включає серверну та клієнтську частини.
— Реалізація: написання коду із застосуванням бібліотек Emgu CV та DirectShowLib для роботи з відеопотоками.
— Тестування: перевірка системи на коректність роботи, продуктивність та відповідність вимогам.
— Впровадження: інтеграція системи у середовище реального використання.

Переваги обраного середовища:

— Гнучкість: підтримка різних мов програмування та платформ.
— Інтеграція: можливість легко підключати сторонні бібліотеки через NuGet.
— Зручність: дружній інтерфейс та інтуїтивно зрозумілі інструменти для новачків і професіоналів.
— Підтримка: наявність великої спільноти розробників та документованих ресурсів.

Отже на основі використання Microsoft Visual Studio як основного програмного середовища дозволило забезпечити високу продуктивність розробки та гнучкість інтеграції компонентів системи. Інструменти, доступні в середовищі, спростили процес налагодження та тестування, а використання сучасних бібліотек значно розширило функціональні можливості системи.

1.2 Аналіз існуючих аналогів

Аналіз існуючих аналогів є важливим етапом розробки будь-якого програмного забезпечення, оскільки він дозволяє оцінити доступні рішення, виявити їхні сильні та слабкі сторони і визначити напрямки для вдосконалення. У даному підрозділі представлено аналіз популярних систем відеоспостереження, що використовуються для забезпечення безпеки господарських об’єктів.

Основні критерії аналізу

Для порівняння існуючих аналогів було обрано наступні критерії:

— Функціональні можливості.

— Простота використання.

— Вимоги до апаратного забезпечення.

— Можливості інтеграції з іншими системами.

— Економічна доцільність.

Розглянуті аналоги

1. Xeoma
    - Функціональність: Xeoma є багатофункціональною платформою, яка підтримує розпізнавання облич, автомобільних номерів, аналіз поведінки об'єктів та виявлення аномалій.
    - Простота використання: Програмне забезпечення має інтуїтивно зрозумілий інтерфейс із підтримкою модульного підходу.
    - Вимоги: Потребує мінімум 2 ГБ оперативної пам’яті та процесора із частотою не менше 2 ГГц.
    - Інтеграція: Підтримує роботу з більшістю IP-камер та відеореєстраторів.
    - Ціна: Містить безкоштовну версію з обмеженим функціоналом та платні версії від $50.
2. Milestone XProtect

— Функціональність: Підтримує масштабовані рішення, які можуть охоплювати до тисячі камер. Здійснює автоматичний аналіз відео, має систему управління доступом.
— Простота використання: Інтерфейс орієнтований на професійних користувачів, тому може бути складним для новачків.
— Вимоги: Потребує потужного серверного обладнання для масштабних систем.
— Інтеграція: Підтримує інтеграцію із системами контролю доступу, охоронними системами та іншими програмними продуктами.
— Ціна: Ціни варіюються залежно від масштабів системи, базова версія — від $500.

3. iSpy
— Функціональність: Відкрите програмне забезпечення, яке підтримує розпізнавання рухів, запис подій та трансляцію відео.
— Простота використання: Має базовий інтерфейс із мінімальними налаштуваннями.
— Вимоги: Може працювати на середніх за потужністю комп’ютерах із Windows.
— Інтеграція: Підтримує інтеграцію з хмарними сховищами та сторонніми модулями.
— Ціна: Безкоштовне, із можливістю придбання додаткових модулів.

4. ZoneMinder
— Функціональність: Пропонує можливості моніторингу в реальному часі, запису подій і виявлення руху.
— Простота використання: Інтерфейс є менш інтуїтивно зрозумілим у порівнянні з конкурентами.
— Вимоги: Працює на Linux, потребує середніх апаратних ресурсів.
— Інтеграція: Може бути інтегрована із сторонніми аналітичними модулями.
— Ціна: Безкоштовне, з відкритим кодом.

5. Hikvision IVMS

— Функціональність: Забезпечує комплексне управління камерами Hikvision із розширеними функціями аналітики.
— Простота використання: Оптимізований для використання із продукцією Hikvision, що спрощує налаштування.
— Вимоги: Потребує відповідних пристроїв Hikvision для максимальної продуктивності.
— Інтеграція: Підтримує інтеграцію із системами контролю доступу Hikvision.
— Ціна: Безкоштовне для використання з обладнанням Hikvision.

Програмне забезпечення	Функціональність	Простота використання	Вимоги до обладнання	Інтеграція	Ціна
Xeoma	Розширена аналітика	Висока	Середні	Підтримка IP-камер	Від $50
Milestone XProtect	Масштабовані системи	Складна	Високі	Інтеграція з охоронними системами	Від $500
iSpy	Базові функції	Середня	Низькі	Хмарні сховища	Безкоштовне
ZoneMinder	Відкритий код	Низька	Середні	Модулі аналітики	Безкоштовне
Hikvision IVMS	Оптимізація для Hikvision	Висока	Середні	Інтеграція з обладнанням Hikvision	Безкоштовне

Отже на основі проведеного аналізу можна зробити висновок, що кожне програмне забезпечення має свої сильні сторони та обмеження. Для реалізації даного проєкту найбільш доцільним є використання рішень, які забезпечують масштабованість, інтеграцію з існуючими системами та доступ до розширених функцій відеоаналітики. Найкращим вибором може стати поєднання рішень Emgu CV і DirectShowLib для створення індивідуального продукту, який задовольнятиме специфічні вимоги господарського об’єкта.

Існує багато програмних аналогів програм відеоспостереження. Однак, деякі з них заслуговують більшої уваги і аналізу. Першою такою із них слід виділити Xeoma (рис. 1.2).

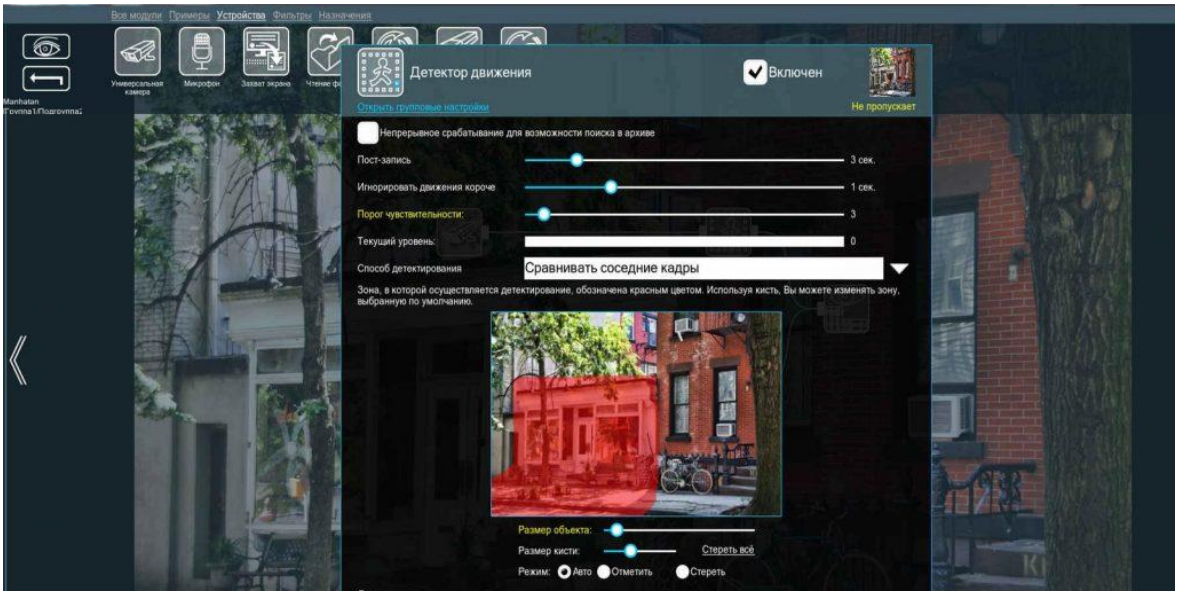

Рисунок 1.2 – Програма для відеоспостереження: Xeoma

Дана програма працює на таких платформах як: Windows, macOS, Linux, iOS, Android.

Найпопулярніше програмне забезпечення (ПЗ) для організації відеоспостереження, яке працює з усіма існуючими камерами. Xeoma запускається на будь-яких комп'ютерах і навіть не вимагає установки.

У програми лаконічний інтерфейс і майже безмежні можливості. Крім звичного детектора руху, в Xeoma реалізовано розпізнавання автомобільних номерів, осіб і навіть емоцій. Всі функції працюють у вигляді модулів, які можна об'єднувати в ланцюжки і дуже тонко налаштовувати.

У безкоштовної версії кількість модулів обмежена лише трьома функціями, якими буде достатньо для проведення не складних операцій. Для більш серйозних завдань є три типи ліцензій, ціна яких залежить від кількості камер.

Неменш важливою функцією сучасного програмного забезпечення являється функція захисного контролю не тільки самого об’єкту який захищає система відеонагляду, а й захист самоконтролю, аби зловмисники не могли добратись до інформації контролю, інформації даних, програмного коду, програмної системи відеонагляду камер і т.д. Одним з таких програмних додатків являється Zoneminder, який неодноразово було чутно на ринках програмних забезпечень відео камер та самих організацій які використовують

дане програмне забезпечення задля захисту контролю відеонагляду того чи іншого об’єкту (рис. 1.3)

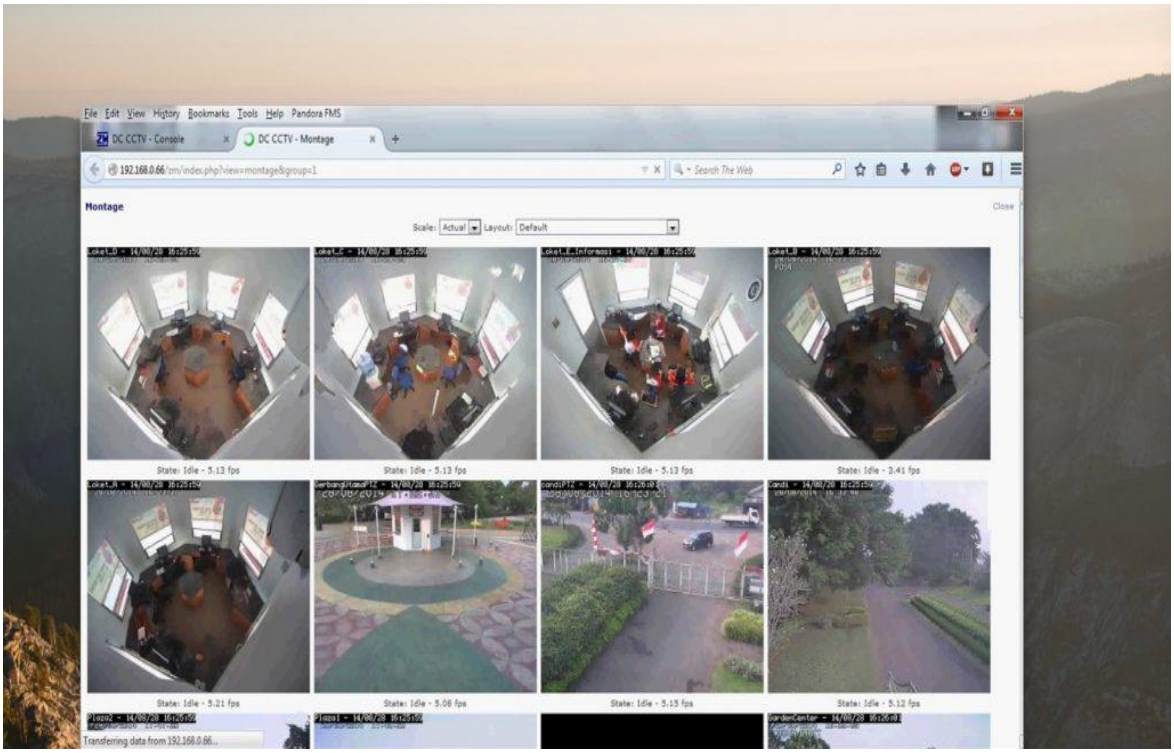

Рисунок 1.3 – Програма для відеоспостереження: Zoneminder

Дане програмне забезпечення працює на таких платформах як:Linux, iOS, Android.

Потужний інструмент з відкритим вихідним кодом і активним спільнотою, який годиться для організації відеоспостереження будь-якої складності. Після настройки переглядати відео можна з комп'ютера або смартфона, з будь-якого браузера.

Zoneminder працює з камерами будь-яких типів, що дає змогу записувати і аналізувати картинку з них. Завдяки розширених налаштувань, для кожної камери можна задати кілька зон визначення рухів і їх чутливість. Уміє відправляти оповіщення на електронну пошту або СМС про задані події.

Додаток повністю безкоштовний як для домашнього, так і для комерційного використання.

Також не можна не відмітити Ispy, яка зображенна на рисунку 1.4.

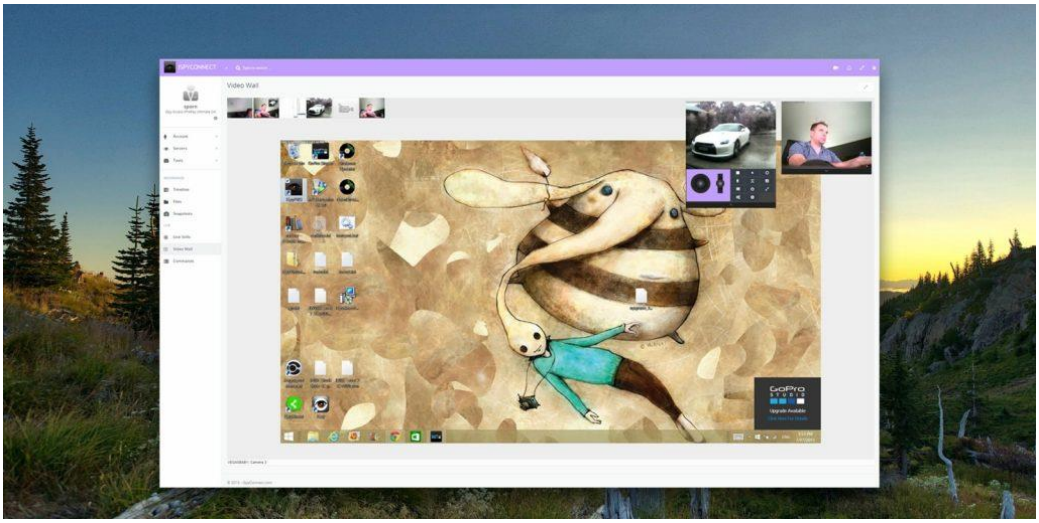

Рисунок 1.4 - Програма для відеоспостереження: iSpy

Дане програмне забезпечення працює на таких платформах як:Windows, iOS, Android.

iSpy має відкритий вихідний код, що надає широкі можливості для модернізації програми і робить детальну настройку дуже зручною. Розширити функціональність можна за допомогою плагінів для розпізнавання автомобільних номерних знаків, накладення тексту, сканування штрих-кодів.

Можна підключити необмежену кількість джерел. Є датчик руху, мережеве мовлення, повідомлення. Крім того, iSpy підтримує завантаження на YouTube, Dropbox або FTP-сервер.

Як джерело можна використовувати не тільки USB- і IP-камери, але і зображення робочого столу.

З урахуванням нових і більш досконалих моделей відеокамер, які постійно виходять на ринок, кожен замовник може створити автоматизований комплекс відеомоніторингу, виходячи зі своїх власних завдань. Дуже зручне рішення - використання принципу визначення руху об'єктів, але потрібно убезпечити себе від випадкових спрацювань. Для цього можна виставити такі критерії, з якими програма не реагувала б на хаотичний рух або засвічення.

Найпростіші охоронні функції поряд з інтуїтивно зрозумілим інтерфейсом зустрічаються практично в кожному обраному програмному забезпеченню, незалежно від виробника. Звичайно, базові програми не можуть похвалитися таким же кількістю функцій, як у професійного софта, який розрахований і може

взаємодіяти з добрим десятком датчиків. Однак виконувати елементарні функції і проводити аналіз зображення вони здатні цілком.

Захищеність цифрової системи, оснащеної віддаленої камерою, досягається тому, що її робота пов'язана з окремим персональним комп'ютером, який знаходиться за межами зони, що охороняється.

Згідно зі статистикою, 2/3 різних правопорушень вдається припинити до їх здійснення і навіть планування. Помітивши підвищену охорону на об'єкті, пов'язану з відеоспостереженням, порушники відмовляються від своїх планів. Необхідно, щоб обраний софт в будь-якому випадку був задуманий, як програмний продукт для відеоконтролю об'єктів будь-якого масштабу. В такому випадку, вдасться створити систему спостереження з повнофункціональним управлінням будь-якими її компонентами. Таке програмне забезпечення буде підтримувати багатосерверних конфігурації і володіти функцією автоматичного контролю працездатності.

Перш ніж розпочати вибір системи відеоспостереження, важливо визначити її основні завдання, зону покриття, яку мають охоплювати камери, а також вимоги до якості зображення. Це дозволить обрати найвідповіднішу систему, від чого, в свою чергу, залежатиме її вартість.

Необхідно врахувати умови роботи камер, такі як рівень освітленості, місце їх розташування, вологість, а також відстань до об’єкта, що спостерігається. Важливим є визначення функцій системи: чи йтиметься лише про спостереження, чи будуть також потрібні функції ідентифікації відвідувачів, контролю входу/виходу, розпізнавання автомобільних номерів та інші.

Аналогові камери зазвичай використовуються для відеоспостереження на невеликих об’єктах, де головним завданням є просто спостереження. Вони легкі в установці та налаштуванні, не потребують особливих навичок чи підтримки системи, а також мають низьку вартість обладнання. Однак такі камери мають застарілу технологію та високу вартість матеріалів.

IP-камери популярні завдяки високій якості зображення та чіткості. Одна з їхніх переваг — можливість об’єднання кількох об’єктів в одну систему,

підключення до камер віддалено через Інтернет без необхідності в відеореєстраторі, що неможливо для аналогових камер. До переваг також відноситься можливість масштабування системи, використання аналітичних модулів та інтеграція з іншими системами, такими як доступ або охорона.

Зберігання даних є важливим елементом в програмному забезпеченні всіх камер. Перш ніж обирати між аналоговими або IP-камерами, необхідно вирішити, який тип відеореєстратора буде використовуватись. Для IP-камер може не знадобитись відеореєстратор взагалі, оскільки відео можна зберігати безпосередньо в мережі.

Також важливо врахувати можливість розширення системи. Рекомендується передбачити резерв по камерах не менше 30%. Щоб система зберігання даних була більш надійною, потрібно використовувати жорсткі диски, які рекомендуються виробником для роботи з відеоспостереженням. Вибір відеореєстратора повинен підтримувати масиви RAID для захисту від втрати інформації через пошкодження жорсткого диска. Крім того, для забезпечення автономності системи слід закласти блок безперебійного живлення і розрахувати необхідну потужність залежно від споживання всієї системи. Важливо обрати правильну кількість і тип підключень до моніторів чи телевізорів.

Виконавши попередні вимоги, можна сміливо починати пошук і закупівлю необхідного обладнання.

Щоб працювати з даним сервісом, необхідно, щоб пристрій підтримував його і цифрова камера повинна бути активною, а в базових налаштуваннях заходи повинні бути прописані: мережеву адресу, шлюз і маска підмережі. Активувати інструмент можна за допомогою локального пристрою GUI, інструменту SADP, через веб-інтерфейс, додаток iVMS-4500 і за допомогою клієнтського ПЗ iVMS-4200. Основні способи підключення до додатку:

Спочатку для використання користувачами обладнання, необхідно створити відповідний код перевірки, для цього потрібно ввести новий код підтвердження і підтвердити, прочитати ліцензійну угоду і натиснути «ОК», щоб зберегти всі налаштування.

Отже, на основі проведеного аналізу було встановлено, сучасних програмних рішень для відеоспостереження, їх функціональність, вартість та інтеграційні можливості. Було визначено, що вибір програмного забезпечення залежить від потреб конкретного господарського об'єкта, а також доступних ресурсів. Запропоновані рішення дозволяють підвищити ефективність відеоспостереження завдяки автоматизації процесів та використанню інтелектуальних алгоритмів обробки відео.

РОЗДІЛ 2 АНАЛІЗ ЗАВДАННЯ НА ДИПЛОМНИЙ ПРОЄКТ

2.1 Аналіз методів

Розробка інтелектуальної системи відеоспостереження потребує використання сучасних методів, що забезпечують ефективність аналізу відеопотоків та автоматизацію процесів розпізнавання. У ході дослідження було розглянуто кілька підходів до обробки відеоінформації, включаючи класичні алгоритми комп’ютерного зору, методи машинного навчання та технології глибокого навчання.Методи комп’ютерного зору дозволяють реалізувати попередню обробку зображень, що включає фільтрацію шумів, покращення контрастності та визначення контурів об'єктів. Вони широко використовуються для автоматичного розпізнавання об’єктів, виявлення змін у кадрах та створення детекторів руху.

На даний момент застосовував такі методи:

— Алгоритми попередньої обробки відеопотоків дозволяють нормалізувати кольори, коригувати освітлення та підвищувати чіткість зображень, що покращує точність подальшого аналізу.

— Методи детекції об’єктів включають застосування контурного аналізу, сегментації зображень та визначення ключових точок, що дозволяє ідентифікувати об'єкти навіть у змінних умовах освітлення.

— Використання глибокого навчання дозволяє створювати нейромережеві моделі для розпізнавання облич, предметів та аномалій у поведінці людей. Особливо ефективними є згорткові нейронні мережі (CNN), які можуть аналізувати великі обсяги даних і забезпечувати високу точність розпізнавання.

— Технології аналізу руху застосовуються для автоматичного відстеження переміщень об'єктів у полі зору камер. Це включає методи оптичного потоку, які

дозволяють визначати швидкість та напрямок руху, що особливо корисно для контролю транспорту або моніторингу громадських місць.

— Застосування методів кластеризації та класифікації дозволяє автоматизувати процеси аналізу відеопотоків, групуючи подібні об’єкти та виділяючи потенційно небезпечні ситуації. Це забезпечує зменшення навантаження на операторів та підвищення ефективності систем безпеки.

Розширений аналіз методів інтелектуальних систем відеоспостереження у сучасних умовах розвитку технологій безпеки застосування методів аналізу відеопотоків є ключовим для забезпечення ефективного моніторингу об’єктів. Розвиток систем штучного інтелекту дозволяє значно розширити можливості відеоспостереження та автоматизувати обробку інформації. В даному розділі розглянуто додаткові методи, які можуть бути застосовані для підвищення точності та швидкодії системи відеоспостереження.

Інтелектуальні системи відеоспостереження базуються на технологіях комп’ютерного зору, які дозволяють розпізнавати обличчя, виявляти рух, визначати об’єкти у відеопотоці та виконувати їхню класифікацію. Одним із найважливіших напрямків є застосування нейронних мереж, які навчаються на великих наборах даних для точного розпізнавання осіб, транспортних засобів, предметів та інших елементів сцени.

— Методи сегментації зображень дозволяють виділяти окремі об’єкти у відеопотоці, що важливо для їх подальшої класифікації. Наприклад, алгоритми Watershed, GrabCut та DeepLabV3+ застосовуються для автоматичної ідентифікації об’єктів у складних умовах освітлення.

— Технології розпізнавання дій та поведінки використовуються для оцінки динаміки об’єктів та їхньої взаємодії. Це дає змогу аналізувати ситуації, що можуть вказувати на потенційну загрозу, наприклад, раптові рухи, натовпи або залишені предмети.

— Оптимізація алгоритмів глибокого навчання відіграє важливу роль у системах відеоспостереження, оскільки обробка відеопотоків вимагає значних обчислювальних ресурсів. Технології, такі як TensorRT, OpenVINO та CoreML, дозволяють прискорити процес розпізнавання об’єктів та зменшити затримки обробки даних.

— Інтеграція з хмарними платформами забезпечує можливість зберігання великих обсягів відеоданих та їхню обробку в реальному часі. Це відкриває нові перспективи для масштабованості систем відеоспостереження та підвищення їхньої ефективності.

Розширені методи застосування

Інтелектуальні системи відеоспостереження можуть бути використані не лише у традиційних сферах, таких як охорона громадських місць, але й у нових напрямах, включаючи аналіз соціальної активності, управління транспортом та автоматизацію промислових процесів.

— Використання в розумних містах: інтелектуальні системи відеоспостереження можуть бути інтегровані в міські інфраструктури для моніторингу дорожнього руху, забезпечення безпеки на транспортних зупинках та оптимізації роботи світлофорів.

— Застосування у медицині: контроль за станом пацієнтів у лікарнях, відстеження динаміки руху в реанімаційних відділеннях та забезпечення швидкого реагування у критичних ситуаціях.

— Використання в промисловості: автоматизований контроль роботи обладнання, запобігання аварійним ситуаціям та моніторинг дотримання техніки безпеки на виробництвах.

Отже, аналіз та впровадження розширених методів комп’ютерного зору, штучного інтелекту та глибокого навчання дозволяють значно підвищити ефективність систем відеоспостереження. Завдяки інноваційним підходам

можливо не лише автоматизувати процеси безпеки, а й інтегрувати такі системи в ширший контекст розумних міст, медицини та промисловості.

2.2 Методи застосування

Інтелектуальні системи відеоспостереження використовуються у широкому спектрі сфер, що потребують автоматизації моніторингу та аналітики.

Основні напрями застосування таких систем включають:

— Безпеку та контроль доступу, що передбачає автоматичне розпізнавання осіб, ідентифікацію транспортних засобів та визначення аномальних ситуацій, таких як залишені предмети або велике скупчення людей.

— Транспортну інфраструктуру, де такі системи допомагають здійснювати моніторинг дорожньої ситуації, виявлення порушень правил дорожнього руху, а також контроль руху транспортних засобів у реальному часі.

— Розумні міста, що використовують відеоаналітику для оптимізації трафіку, контролю екологічної ситуації та моніторингу громадських просторів. Наприклад, автоматичний аналіз руху пішоходів дозволяє регулювати сигнали світлофорів та зменшувати затори.

— Промисловість та виробничі підприємства, де відеоспостереження допомагає запобігати нещасним випадкам, контролювати дотримання техніки безпеки та автоматично виявляти несправності обладнання на основі відеоаналізу.

— Освітні та комерційні заклади, що застосовують відеоаналітику для автоматизованого контролю доступу, аналізу поведінки відвідувачів та покращення рівня безпеки. Це особливо актуально для великих університетських кампусів, торгових центрів та бізнес-центрів.

Крім того, сучасні системи відеоспостереження використовують штучний інтелект для автоматичного розпізнавання нестандартних ситуацій. Наприклад, у банківській сфері такі системи можуть ідентифікувати підозрілу поведінку клієнтів, а в медичних установах — відстежувати стан пацієнтів та запобігати можливим інцидентам.

Розширене застосування інтелектуальних систем відеоспостереження у сучасних умовах відеоспостереження виходить за рамки традиційного моніторингу безпеки та використовується у різних сферах. Завдяки новітнім технологіям ці системи можуть виконувати аналіз поведінки, прогнозування подій та навіть адаптуватися до мінливих умов зовнішнього середовища.

—Безпека у громадських місцях: відеоспостереження застосовується для моніторингу великих скупчень людей, контролю ситуацій на масових заходах та ідентифікації підозрілих осіб.

—Контроль дорожнього руху: інтелектуальні системи можуть автоматично визначати порушення правил дорожнього руху, розпізнавати номерні знаки транспортних засобів та контролювати завантаженість доріг.

—Автоматизація бізнес-процесів: магазини, торгові центри та фінансові установи використовують камери для аналізу поведінки клієнтів, запобігання крадіжкам та оптимізації обслуговування.

—Вплив на екологію: системи відеоспостереження можуть бути інтегровані у проекти з моніторингу рівня забруднення повітря, води та інших екологічних показників.

—Індустріальне використання: контроль за роботою виробничого обладнання, моніторинг дотримання техніки безпеки та виявлення аварійних ситуацій у режимі реального часу.

Інноваційні підходи до застосування

Завдяки розвитку штучного інтелекту та нейромережевих технологій, відеоспостереження стає більш гнучким та функціональним. Розглянемо декілька новітніх підходів до його застосування.

—Аналітика поведінки: аналіз динаміки руху, прогнозування потенційних загроз та оцінка соціальної активності.

—Адаптивні алгоритми: камери можуть змінювати налаштування відповідно до часу доби, погодних умов та рівня освітлення.

—Автоматичне визначення загроз: системи можуть миттєво реагувати на нестандартні ситуації, такі як вибухи, пожежі або агресивна поведінка.

—Інтеграція з іншими цифровими платформами: поєднання відеоспостереження з базами даних, біометричними системами та мережевими аналітичними центрами.

—Хмарні технології та зберігання даних: можливість перегляду записів у реальному часі та доступ до архівів з будь-якої точки світу.

Майбутнє відеоспостереження

Очікується, що найближчим часом відеоспостереження стане ще більш автономним та ефективним. Розвиток технологій обробки зображень у поєднанні з високопродуктивними алгоритмами машинного навчання дозволить системам прогнозувати події та діяти на випередження.

—Розширена відеоаналітика: можливість аналізу емоцій, рухів та стану здоров’я осіб у полі зору камери.

—Автоматизоване прийняття рішень: камери зможуть автоматично викликати екстрені служби або активувати системи захисту у випадку небезпеки.

—Мікросенсорні камери: мінімізація розмірів камер та впровадження їх у предмети повсякденного вжитку, такі як окуляри чи дорожні знаки.

—Кібербезпека у відеоспостереженні: запобігання несанкціонованому доступу та захист персональних даних у відеопотоках.

Отже, розвиток інтелектуальних систем відеоспостереження не лише змінює підходи до безпеки, а й відкриває нові можливості для моніторингу, аналізу та управління процесами у різних сферах людської діяльності.

Отже, на основі проведеного аналізу було встановлено, що інтелектуальні системи відеоспостереження є необхідним елементом сучасних технологій безпеки. Використання методів комп’ютерного зору, нейромережевої аналітики та автоматизованих алгоритмів дозволяє значно підвищити ефективність моніторингу, знизити навантаження на операторів та забезпечити швидке реагування на нестандартні ситуації.

3 ОСНОВНА ЧАСТИНА ДИПЛОМНОГО ПРОЄКТУ

3.1 Вибір програмного забезпечення

Мета дипломного проєкту: розробка проєкту системи відеоспостереження для навчального закладу. Одним з факторів відеоспостереження є записуюче обладнання. Найчастіше немає сенсу купувати окремий DVR рекордер, коли присутній відносно вільний комп'ютер, який може виконувати ті ж функції. Програма для ір камер підійде найкращим чином з цією метою. Данний додаток я буду створювати на мові C# через Unity та Visual Studio.

Microsoft Visual Studio — це серія продуктів від компанії Майкрософт, що включає в себе інтегроване середовище розробки (IDE) та низку інших інструментів для розробки програмного забезпечення. Ці інструменти дозволяють створювати як консольні додатки, так і програми з графічним інтерфейсом, зокрема з підтримкою Windows Forms. Крім того, Visual Studio дозволяє розробляти веб-сайти, веб-застосунки та веб-служби для різних платформ, що підтримуються Microsoft, таких як Windows, Windows Mobile, Windows Phone, Windows CE, .NET Framework, .NET Compact Framework і Microsoft Silverlight.

Таблиця 3.1 – Інформаційне поле Microsoft Visual Studio

Visual Studio	
1	2
Тип	інтегроване середовище розробки
Розробник	Microsoft

Версії	macOS: 2019 8.9.0.1651 (2 березня 2021) і Microsoft Windows: 2019 16.9.0 (3 лютого 2021)
Операційна система	Microsoft Windows
Мова програмування	C++ та C#
Доступні мови	англійська, китайська (спрощена і традиційна), іспанська, італійська, корейська, німецька, португальська, російська, французька, японська
Ліцензія	Microsoft EULA
Вебсайт	visualstudio.com

Кінець таблиці 3.1

Visual Studio 97, що мала кодову назву "Boston," стала першою версією інтегрованого середовища розробки (IDE) від Microsoft, яка об'єднала кілька інструментів для створення програмного забезпечення в єдиній платформі. Випущена у двох варіантах — Professional та Enterprise — вона включала Visual Basic 5.0, Visual C++ 5.0, Visual J++ 1.1, Visual FoxPro 5.0 та ASP-середовище розробки Visual InterDev. Уперше Microsoft спробувала створити універсальне середовище для різних мов програмування. Visual C++, Visual J++, Visual InterDev і MSDN використовували спільне середовище Developer Studio, тоді як Visual Basic і Visual FoxPro залишалися на окремих платформах.

Visual Studio 6.0 (кодова назва "Aspen") з'явилася в червні 1998 року і стала останньою версією, що підтримувала Windows 9x. Вона набула популярності серед розробників, які активно використовували Visual Basic. Протягом наступних чотирьох років ця версія залишалася основним інструментом для створення програм під Windows до появи платформи .NET. Visual Studio 6.0 також була останньою версією, що підтримувала COM-реалізацію Visual Basic і мову програмування Visual J++. Випускалася у версіях

Professional та Enterprise, причому остання включала додаткові інструменти, такі як Application Performance Explorer, Automation Manager, Visual Modeler, RemAuto Connection Manager і Visual Studio Analyzer.

Visual Studio .NET 2002 (кодова назва "Rainier") побачила світ у лютому 2002 року. Це був перший випуск, який інтегрував платформу .NET Framework 1.0, що дозволяло компілювати програми у проміжний код — Microsoft Intermediate Language (MSIL), також відомий як Common Intermediate Language (CIL). Завдяки цьому кодування стало кросплатформним, хоча виконання програм було можливим лише на платформах із підтримкою Common Language Infrastructure (CLI). Visual Studio .NET 2002 запропонувала чотири редакції: Academic, Professional, Enterprise Developer і Enterprise Architect. Уперше було представлено мову програмування C#, а також спадкоємця Visual J++, який отримав назву Visual J#. За допомогою ASP.NET розробники могли створювати веб-додатки та сервіси.

Visual Studio .NET 2003 (кодова назва "Everett") вийшла у квітні 2003 року і включала .NET Framework 1.1. Ця версія вперше дозволила розробляти програми для мобільних пристроїв завдяки використанню .NET Compact Framework. Visual Studio .NET 2003 випускалася в тих самих редакціях, що й попередня версія, але версія Enterprise Architect містила додатковий інструмент — Microsoft Visio 2002 для побудови UML-моделей. Оновлення для цієї версії було випущено у вересні 2006 року.

Visual Studio 2005 (кодова назва "Whidbey") стала доступною наприкінці жовтня 2005 року, включаючи підтримку .NET Framework 2.0. У цій версії були представлені Express-редакції (Visual C++ 2005 Express, Visual Basic 2005 Express, Visual C# 2005 Express тощо), які стали безкоштовними у квітні 2006 року. Visual Studio 2005 підтримувала розробку 64-бітних додатків, включаючи 64-бітні версії стандартних бібліотек. Додатково були представлені Visual Studio Tools for Applications (VSA) та Visual Basic for Applications (VBA) із підтримкою

Microsoft Office 2007. Згодом було додано підтримку таких технологій, як WPF, WCF, WF та LINQ.

Visual Studio 2008 (кодова назва "Orcas") дебютувала в листопаді 2007 року разом із .NET Framework 3.5. Ця версія була орієнтована на розробку додатків для Windows Vista, Microsoft Office 2007 і веб-додатків. Нововведення включали Windows Presentation Foundation для візуальної розробки, новий HTML/CSS-редактор і понад 250 нових функцій. Було додано підтримку AJAX/JSON для веб-додатків і нові елементи управління ASP.NET. Крім того, .NET Framework 3.5 приніс функції для сервіс-орієнтованої архітектури (SOA) і розробки Web 2.0-додатків.

Visual Studio 2010, представлена 12 квітня 2010 року, інтегрувала .NET Framework 4.0 і запропонувала нову мову програмування — F#. Було повністю перероблено інтерфейс із використанням Windows Presentation Foundation (WPF). У цій версії були впроваджені інструменти для хмарної розробки, підтримка C++0x, ASP.NET нового покоління і багатопоточних додатків. Visual Studio Ultimate 2010 (кодова назва "Rosario") стала платформою для командної розробки із сучасними засобами управління проектами.

Visual Studio 2012, випущена 2 серпня 2012 року, принесла підтримку .NET Framework 4.5, Windows Runtime і C++ AMP для GPGPU-програмування. У цій версії було впроваджено новий інтерфейс і тип проектів для створення нативних застосунків під Windows 8. Оновлення інструментів включали покращений Solution Explorer та засоби тестування. Visual Studio 2013, що вийшла у жовтні 2013 року, продовжила вдосконалення з акцентом на командну роботу, діагностику, мобільну та веб-розробку, а також представила нові інструменти для аналізу продуктивності та роботи з пам'яттю.

Таблиця 3.2 - Протягом розробки мови C # було випущено кілька його версій

Версія	Специфікація мови			Дата	.NET	Visual Studio
	ECMA	ISO/IEC	Microsoft			
1	2	3	4	5	6	7
C# 1.0	Грудень 2002	Квітень 2003	Січень 2002	Січень 2002	.NET Framework 1.0	Visual Studio .NET (2002)
C# 1.1 C# 1.2			Жовтень 2003	Квітень 2003	.NET Framework 1.1	Visual Studio .NET 2003

Продовження таблиці 3.2

1	2	3	4	5	6	7
C# 2.0	Червень 2006	Вересень 2006	Вересень 2005	Листопад 2005	.NET Framework 2.0 .NET Framework 3.0	Visual Studio 2005
C# 3.0	Відсутнє		Серпень 2007	Листопад 2007	.NET Framework 2.0 (кроме LINQ) .NET Framework 3.0 (кроме LINQ)	Visual Studio 2008

					.NET Framework 3.5	
C# 4.0			Квітень 2010	Квітень 2010	.NET Framework 4.0	Visual Studio 2010
C# 5.0	Грудень 2017	Грудень 2018	Червень 2013	Серпень 2012	.NET Framework 4.5	Visual Studio 2012
C# 5.0	Грудень	Грудень	Черпень 2013	Серпень 2012	.NET Framework 4.5	Visual Studio 2012

Кінець таблиці 3.2

1	2	3	4	5	6	7
C# 7.0	Відсутнє	Відсутнє	Березень 2017 .NET Framework 4.6.2 .NET Framework 4.7		Visual Studio 2017 15.0	Березень 2017
C# 7.1	Відсутнє	Відсутнє	Серпень 2017 .NET Core 2.0		Visual Studio 2017 15.3	Серпень 2017

C# 7.2	Відсутнє	Відсутнє	Листопад 2017	Visual Studio 2017 15.5	Листопад 2017
C# 7.3	Відсутнє	Відсутнє	Травень 2018	. NET Core 2.1	Травень 2018
C# 8.0	Відсутнє	Відсутнє	Вересень 2019 .NET Core 3.0 .NET Core 3.1 .NET Framework 4.8	Visual Studio 2019 16.3	Серпень 2019
C# 9.0	Відсутнє	Відсутнє	Вересень 2020	.NET 5.0	Visual Studio 2019 16.8

Таблиця 3.3 - Загальна інформація за версіями

Версія	Нововведення
1	2
C# 2.0	• Часткові типи • Узагальнені типи (generics) • Ітератори і ключове слово yield • Анонімні методи • Оператор null-об'єднання • Nullable-типи

C# 3.0	• Запити, інтегровані в мову (LINQ) • Ініціалізатор об'єктів і колекцій • Лямбда-вирази • дерева виразів • Неявна типізація і ключове слово var • Анонімні типи • методи розширення • автоматичні властивості
C# 4.0	• Динамічне зв'язування і ключове слово dynamic • Іменовані і опціональні аргументи • Узагальнена коваріантність і контрваріантность • Бібліотека TPL, концепція завдань і класи Task, Parallel • клас MemoryCache • Класи паралельних колекцій
Ċ# 5.0	• шаблон TAP • Асинхронні методи async і await

Продовження таблиці 3.3

1	2
C# 6.0	• Компілятор як сервіс • Імпорт членів статичних типів в простір імен • фільтри винятків • await в блоках catch / finally • ініціалізатор автосвойств • Автосвойства тільки для читання • null-умовні операції (?. і? []) • інтерполяція рядків • оператор nameof

	• ініціалізатор словника • Функції стислі до виразів
C# 7.0	• Компілятор як сервіс • Імпорт членів статичних типів в простір імен • фільтри винятків • await в блоках catch / finally • ініціалізатор автосвойств • Автосвойства тільки для читання • null-умовні операції (?. і? []) • інтерполяція рядків • оператор nameof • ініціалізатор словника • Функції стислі до виразів
C# 8.0	• Члени інтерфейсу за замовчуванням

Закінчення таблиці 3.3

1	2
C# 9.0	• Оператор об'єднання з null • Порожні параметри для лямбда-виразів • Native Int: nint, nuint • диз'юнктоне об'єднання • Додано with-вирази • новий модифікатор initonly

Unity — це універсальний рушій і платформа для розробки відеоігор і додатків, що підтримує широкий спектр різних пристроїв і платформ. З його допомогою можна створювати програми для настільних комп'ютерів, мобільних

гаджетів, ігрових консолей, а також для технологій віртуальної та доповненої реальності. Unity дає змогу розробникам працювати з різноманітними типами графіки, включаючи двовимірні та тривимірні зображення, підтримує DirectX, OpenGL і OpenGL ES.

Основний інтерфейс Unity є гнучким, з можливістю налаштовувати розташування вікон, що забезпечує зручну роботу з проектами. Ключові елементи інтерфейсу включають оглядач ресурсів, інспектор для управління поточними об'єктами, попередній перегляд, а також оглядач сцени та ієрархії. Проект в Unity складається з окремих сцен, кожна з яких містить свій набір ігрових об'єктів, скриптів та налаштувань.

Unity підтримує фізику твердих тіл і тканини, а також спеціальні ефекти типу Ragdoll (ганчіркова лялька). Ігрові об'єкти можуть мати різноманітні компоненти, такі як трансформації, рендеринг геометрії та анімацію. Для роботи з графікою використовуються шейдери, текстури, матеріали, а також часткові системи для відображення диму, рідин і інших ефектів.

Unity підтримує інтеграцію з популярними програмами для створення графіки, такими як 3ds Max, Maya, Blender, ZBrush та іншими. Це дозволяє імпортувати моделі і налаштовувати їх безпосередньо в Unity. Для написання шейдерів використовуються GLSL або Cg, а для скриптів — UnityScript, C# або Boo. У версії 5.2 була інтегрована підтримка Visual Studio для редагування скриптів.

Система контролю версій Unity дозволяє ефективно управляти великими проектами, забезпечуючи збереження і перегляд історії змін ресурсів. Це забезпечує безперебійний робочий процес, навіть у разі переміщення чи перейменування файлів, і дозволяє працювати з великими файлами без проблем.

.

Сервер ресурсів доступний як для Mac OS X Installer, так і для Linux RPMs . Підтримка декількох платформ забезпечує гнучкість у впровадженні Сервера ресурсів Unity у наявну IT-інфраструктуру на таблиці 3.4.

Таблиця 3.4 – Підтримка декілько платформ

Тип ліцензии	Дохід компанії в рік	Екран привітання	Розраховані на багато користувачів функції	Ціна
1	2	3	4	5
Personal	До 100 000 до лларів	Призначена	20 CCUs	Бесплатно
Plus	До 200 000 до лларів	Призначена для користувача анімація і / або «Made With Unity»	50 CCUs	399 в рік
Pro	Не обмежений	Призначена для користувача анімація і / або «	200 CCUs	1800 в рік или 150 ежемесячно
Enterprise	Не обмежений	Призначена	Користуваць кий	200$ в місяц

Unity — це потужний багатофункціональний інструмент для створення відеоігор і програм, а також для розробки застосунків, які працюють на різних платформах. Його інтуїтивно зрозумілий редактор, оснащений простим Drag & Drop інтерфейсом і підтримкою плагіна KALI, дозволяє швидко налаштовувати проект. Unity підтримує написання скриптів на C#, а раніше була доступна підтримка Boo (Python-подібний діалект), а також UnityScript, яке більше не використовується.

Фізика в Unity розраховується через потужний рушій PhysX від NVIDIA. Відеоадаптери використовують графічний API, зокрема DirectX (від версії DX 11 і 12). Кожен проект в Unity організовано у вигляді сцен (рівнів), кожна з яких містить окремі світи, об'єкти, сценарії та налаштування.

Об'єкти в Unity можуть бути як з фізичною моделлю, так і "порожніми" (без моделі), які виконують специфічні функції, такі як тригери або точки збереження. Кожен об'єкт має компоненти, з якими взаємодіють скрипти. Наприклад, обов'язковий компонент **Transform** визначає позицію, поворот і

масштаб об'єкта. Для моделей також використовується **Mesh Renderer**, що робить їх видимими. Окрім того, Unity підтримує різноманітні типи колізій (коллайдери) для фізичних взаємодій.

Існує система успадкування об'єктів, при якій дочірні елементи автоматично приймають зміни своїх батьків. Коли імпортується текстура, Unity дозволяє генерувати альфа-канал, міпрівні та різні карти, але текстуру не можна безпосередньо прикріпити до моделі — замість цього створюється матеріал, який потім асоціюється з модельною геометрією.

Щодо анімації, Unity підтримує вбудовані інструменти для створення анімацій, а також дозволяє імпортувати анімацію з зовнішніх 3D-редакторів. Важливою функцією є **Level Of Detail (LOD)**, де на великих відстанях від гравця високодеталізовані моделі замінюються менш деталізованими, що покращує продуктивність. Також підтримується **Occlusion culling**, що дозволяє оптимізувати відображення геометрії та колізій на об'єктах, які не знаходяться в полі зору камери.

Unity підтримує різні формати для ресурсів, включаючи моделі, звуки, текстури та матеріали, які можна упаковувати у формат **.unitypackage**. Ці пакети можна передавати іншим розробникам або використовувати для публікації в **Unity Asset Store**. В Asset Store можна знайти як безкоштовні, так і платні елементи для гри.

Для багатокористувацької розробки в Unity можна використовувати різні інструменти для контролю версій, такі як Tortoise SVN або Git. Підтримка серверів спільної роботи також є через **Unity Asset Server**, що полегшує процеси розробки в команді.

Основними перевагами Unity є візуальне середовище розробки і підтримка різних платформ. Це дозволяє розробникам створювати проекти для ПК, мобільних пристроїв, консолей та інших платформ, використовуючи однакові інструменти. Крім того, Unity використовує модульну систему, що дозволяє створювати ігрові об'єкти з комбінованих функціональних блоків.

Проте є деякі недоліки: обмеження у візуальному редагуванні складних сцен, проблеми з посиланнями на зовнішні бібліотеки і необхідність самостійної

налаштування для командної роботи. Також WebGL-версія має проблеми з продуктивністю і пам'яттю на мобільних пристроях через специфічну архітектуру трансляції коду.

3.2 Проєктування власного додатку

Як для першого так і для другого власного додатку потрібно було завантажити середовище на яких створювалася і тестувалася програми, а також додаток через який я міг би перетворити проєкт в файл встановлення. Для цього перед початком встановлення Visual Studip 2019 потрібно:

1. Перевірити вимоги до системи. Так можна дізантись, чи підтримує комп'ютер Visual Studio 2019.
2. Застосував актуальні оновлення Windows. Ці оновлення гарантують, що на комп'ютері встановлено останні оновлення для системи безпеки і необхідні системні компоненти для Visual Studio.
3. Перезавантажив систему. Перезавантаження гарантує, що очікують установки або оновлення компоненти не будуть перешкоджати встановленню Visual Studio.
4. Звільнив місце. Видалив непотрібні файли і додатки з системного диска. Наприклад, запустив додаток очищення диска.

Потім запустив інсталятор, тобто файл vs_Community.exe.
Спочатку установника необхідно виконати підготовчі дії, натиснувши «Продовжити», тим самим ми також погоджуємося з умовами ліцензійної угоди які показані на рисунку 3.1.

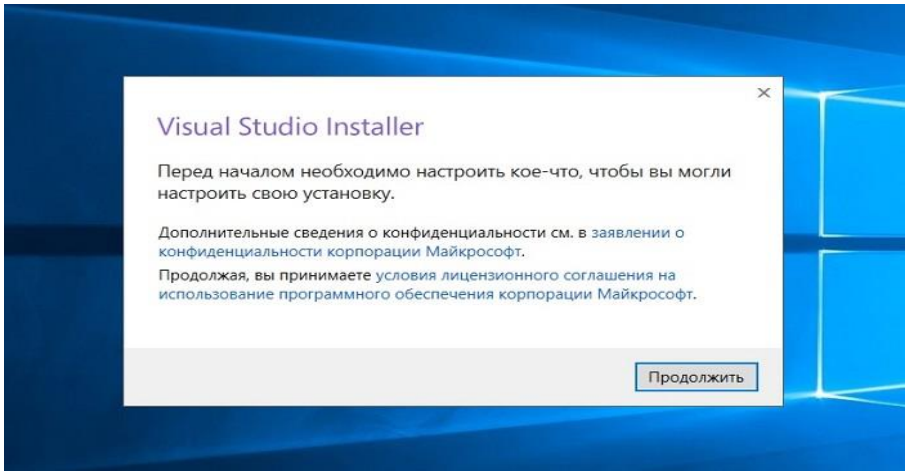

Рисунок 3.1 – Скачанний інсталятор

Після чого буде завантажено і встановляться необхідні файли установника.

Після того як установник виконає всі необхідні попередні заходи, він запуститься. Потім вибравши «Робочі навантаження», тобто що створити за допомогою Visual Studio 2019 Community, вибравши розробку як класичних додатків під комп'ютер, так і розробку Web-додатків.

Після цього можна відразу натискати «Встановити», але в разі потреби можна більш детально налаштувати установку, для цього є додаткові вкладки: «Окремі компоненти», «Мовні пакети» і «Розташування установки» які зображенно на рисунку 3.2.

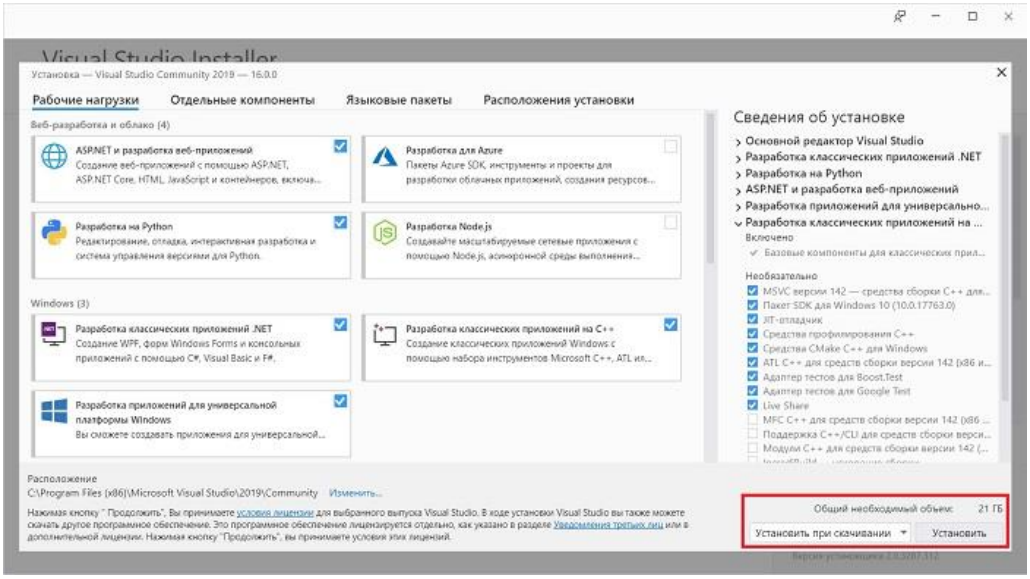

Рисунок 3.2 – Додатки Visual Studio 2019

Вкладка «Окремі компоненти» - якщо є така необхідність, конкретизував компоненти, які необхідно встановити, на цій вкладці які зображено на рисунках 3.3-3.5.

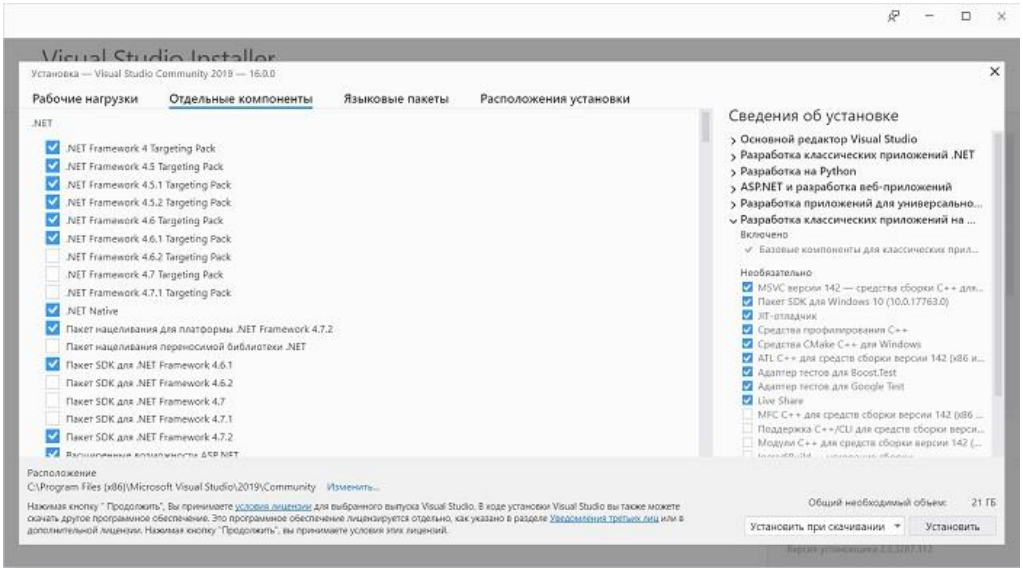

Рисунок 3.3 – Завантаження необхідних компонентів

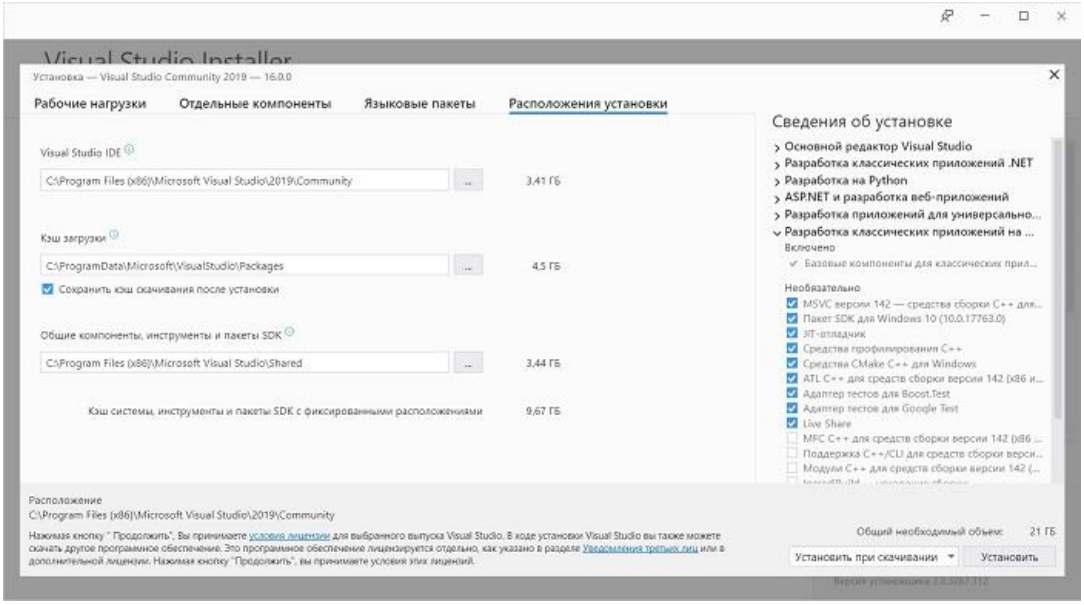

Рисунок 3.4 – Завантаження необхідних компонентів

Після того натиснувши на кнопку «встановити», почнеться процес завантаження і установки всіх обраних компонентів. Залежно від обсягу компонентів, швидкості інтернету і характеристик ПК тривалість даного процесу буде відрізнятися, приблизно у 20-30 хвилин.

Коли з'явиться наступне вікно, установка буде завершена, натискаємо «Перезавантажити» на рисунку 3.5.

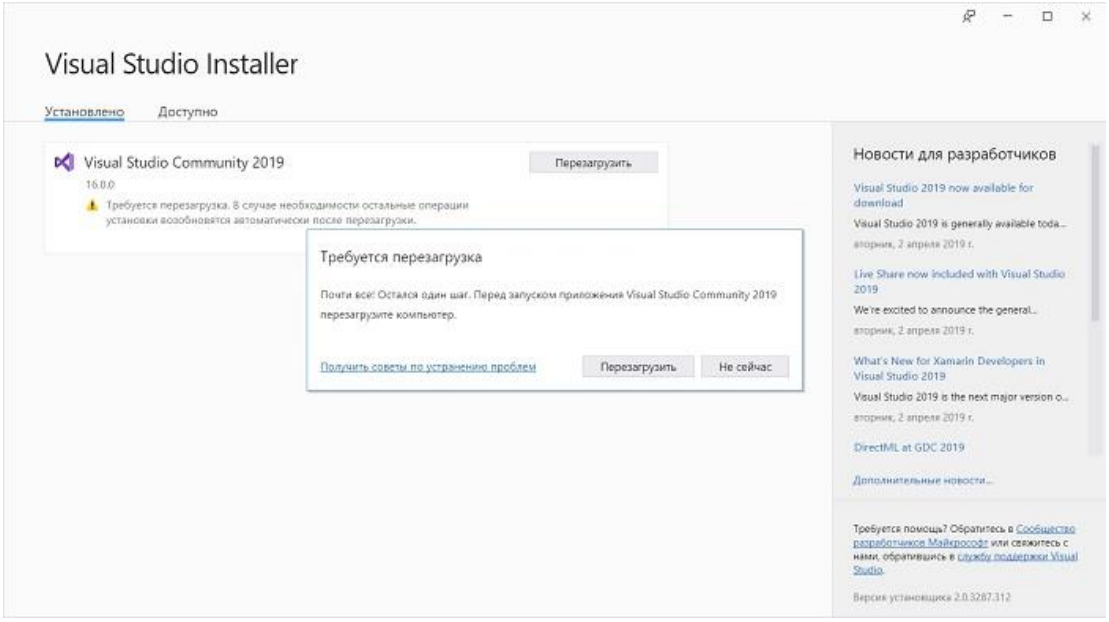

Рисунок 3.5 – Перезавантаження ноутбука

Запустивши Visual Studio Community 2019, і подивився, як вона виглядає, і для прикладу створивши проєкт програми, і запустивши його на виконання.При першому запуску запропонували увійти в обліковий запис, у який ввійшов вже існуючий на рисунку 3.6.

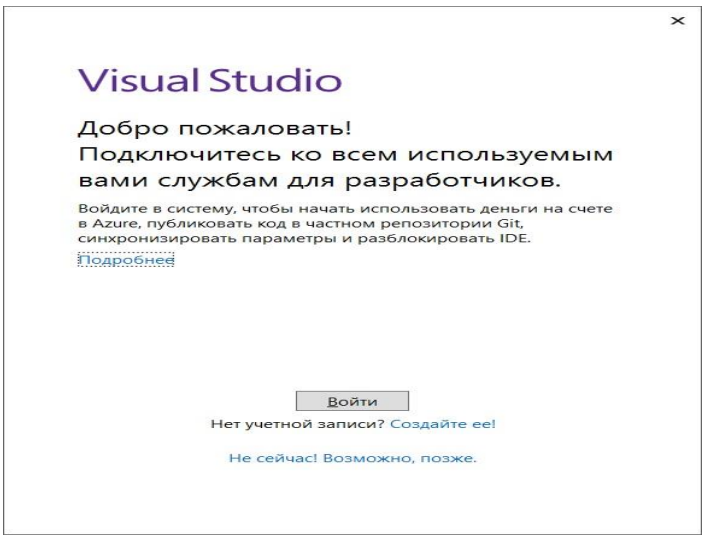

Рисунок 3.6 – Вхід в обліковий запис

Потім вибиравши колірну схему оформлення середовища Visual Studio і натиснув «Запуск Visual Studio» на рисунку 3.7.

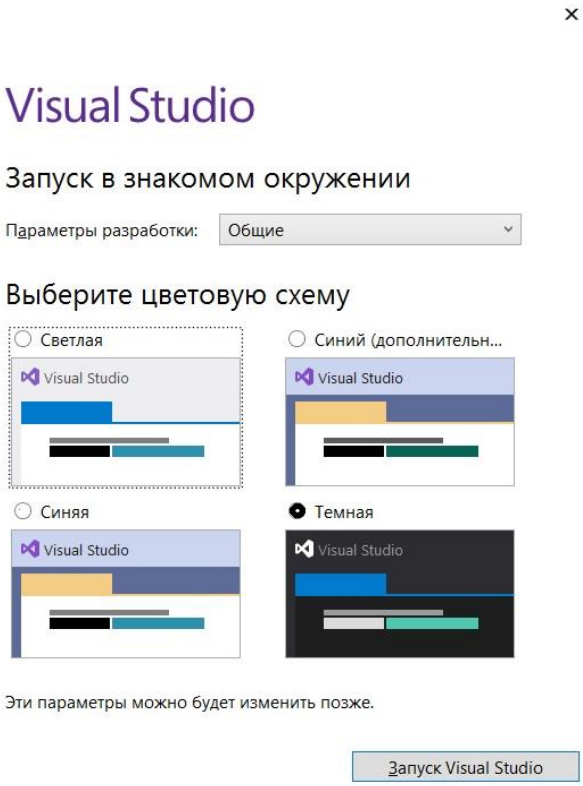

Рисунок 3.7 – Вибір колірної теми

Для прикладу відразу створивши проєкт, як шаблон проєкту вибрав «Windows Forms App (.NET Framework)». Натиснув «Далі» вказавши назву проєкту і розташування файлів цього проєкту назвавши проєкт «DIPLOM» і натиснув “Create” щоб створити, після цього висвітилась заватажувальна лінія для створення проєкту яка триває до 5-ти хвилин і вже після цього створився пустий проєкт ,дані процеси показані на рисунках 3.8 – 3.12.

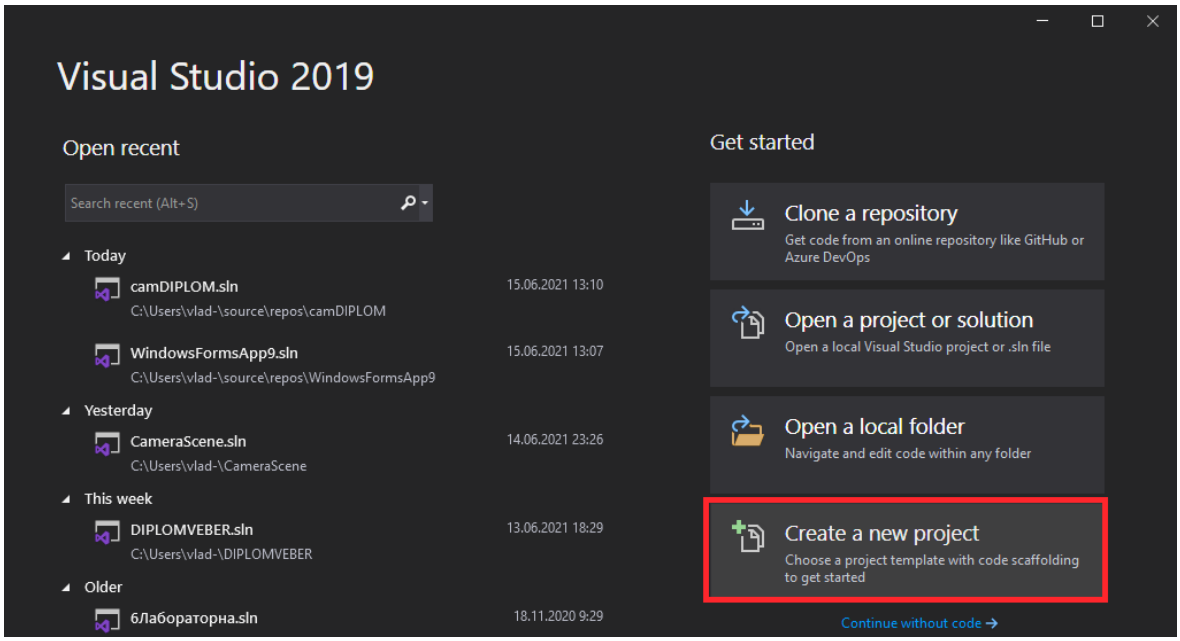

Рисунок 3.8 – Створення проєкту

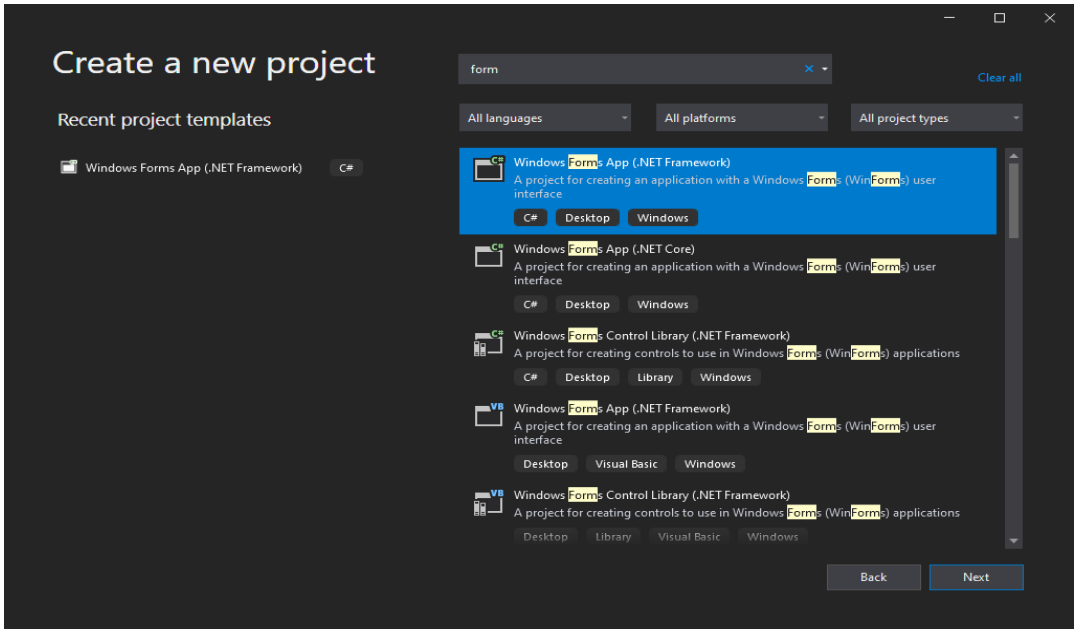

Рисунок 3.9 – Створення проєкту

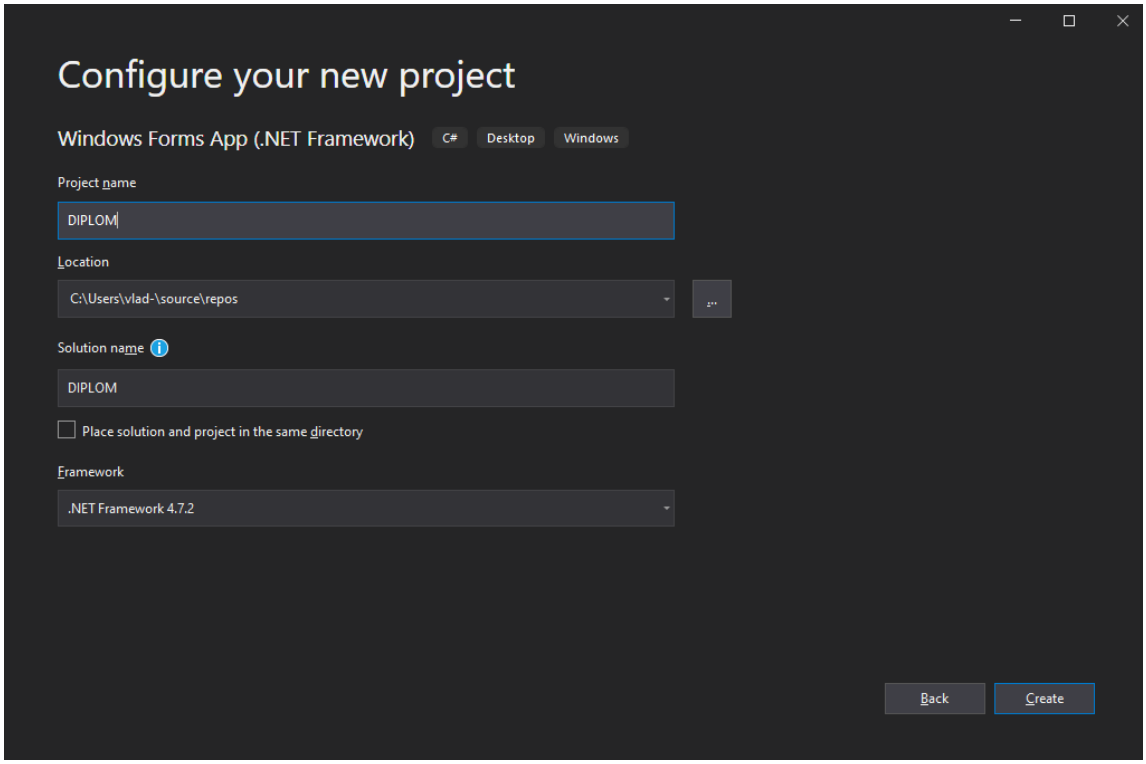

Рисунок 3.10 – Створення проєкту

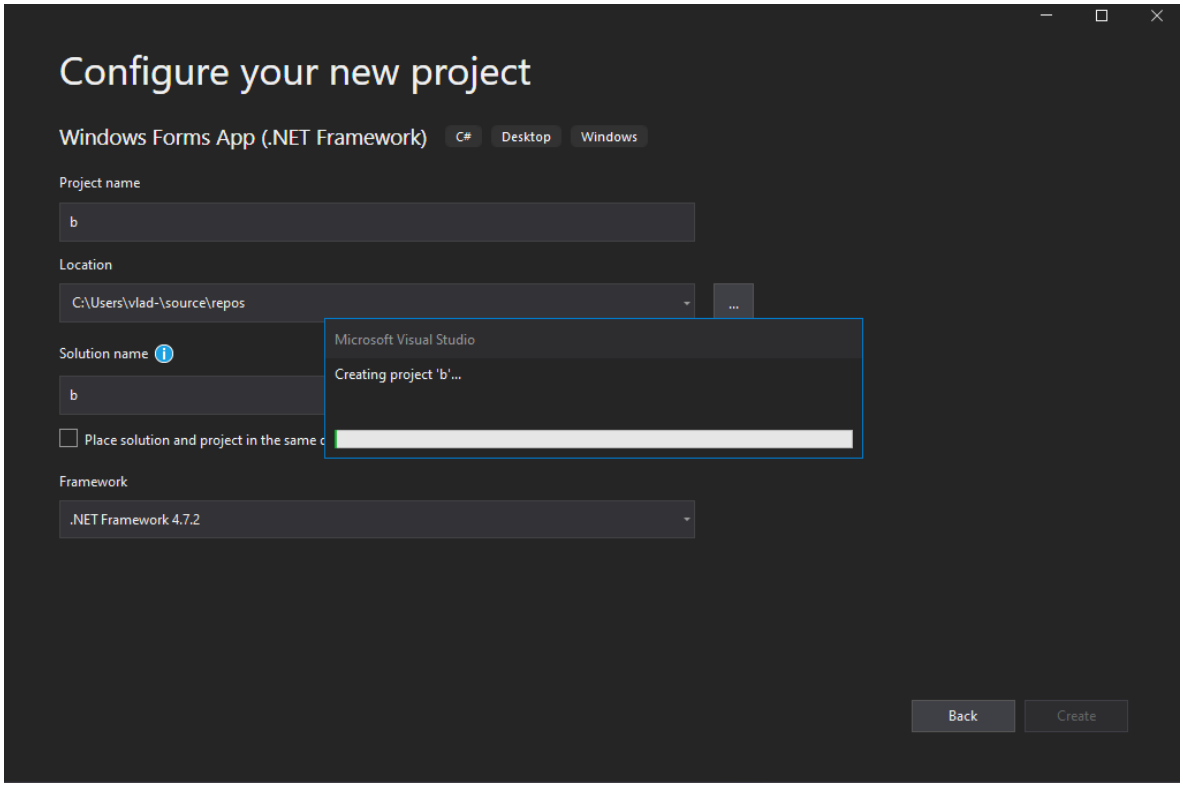

Рисунок 3.11 – Створення проєкту

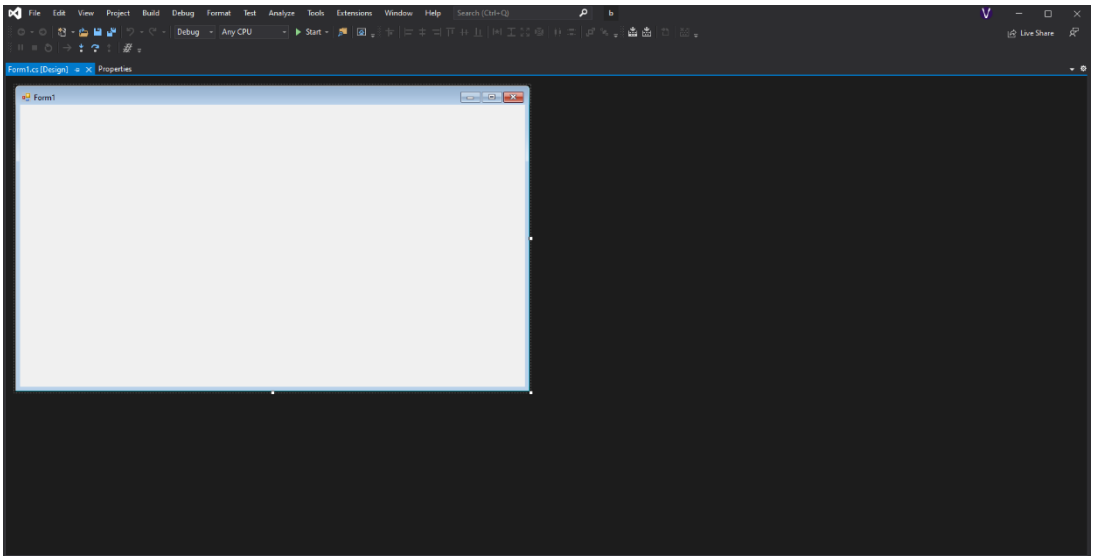

Рисунок 3.12 – Створення проєкту

Після створення проєкту, добавивши властивості, добавилася стіна інструментів «Tool box» і стіна властивостей «Properties».Дану форму одразу ж перемейнував в «WV Cam». В стіні інструментів вставив у форму «Menustrip» який назвав “Файл”, а також “toolstrip” і перейменував на «Камери».В під пункті «toolstrip» вибрав «ComboBox» для вибору камер, а також добавив біля нього 4 кнопки «Button2» і перемейнував на «Перегляд», «Пауза», «Стоп», «Зробити

скріншот» для подальших дій. Задля показу камер які будуть відображатись, також добавив «Picture Box» і підпункті замінив на функцію zoom,яка ж одразу розширилась по ширині та довжині нижче кнопок, дані діх показані на рисунку 3.13.

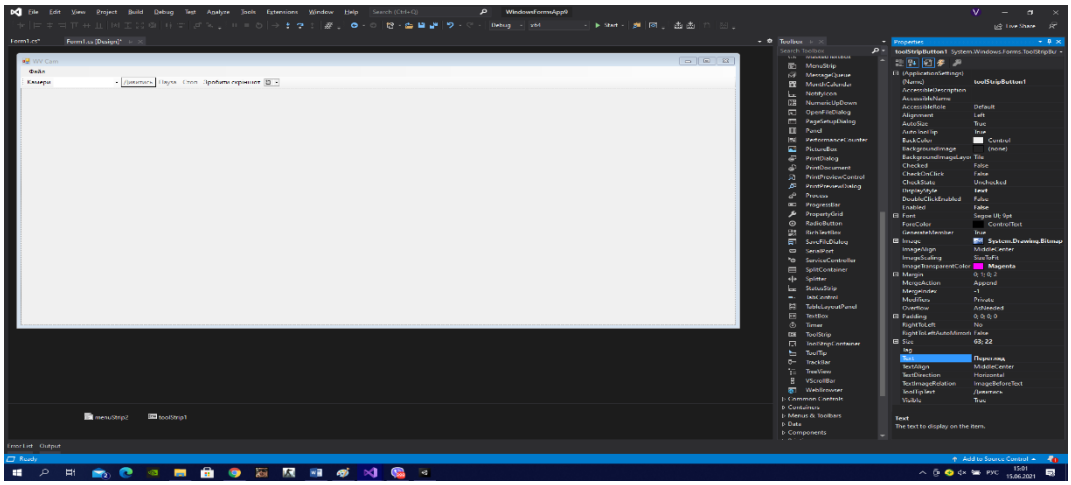

Рисунок 3.13 – Вигляд форми додотку

В меню пакетів програм «NuGet», завантажив пакет «Emgu.CV» та «DirectShowLib», рисунку 3.14.

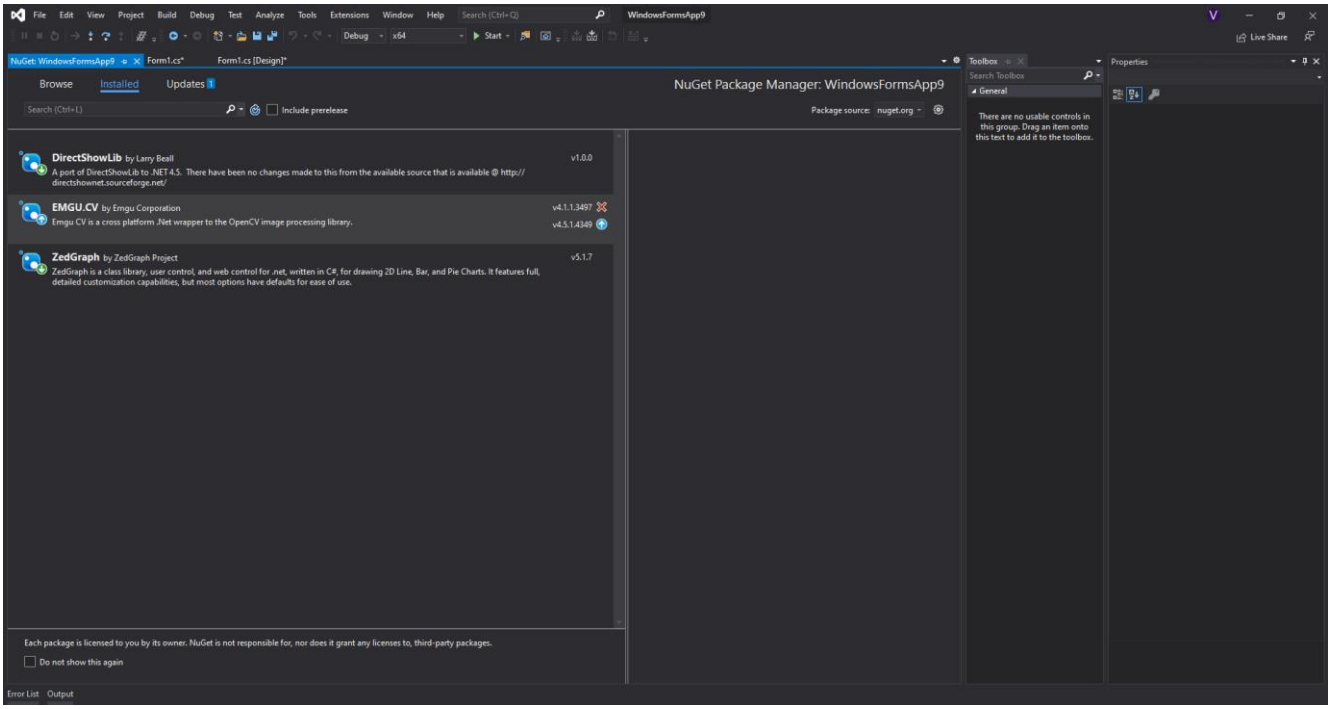

Рисунок 3.14 – Завантаженні пакети

Аби працював додаток, потрібно було прописати код. Для початку прописав бібліотеки які необхідні для подальших функцій

```
using System;
```

```
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using Emgu;
using Emgu.CV;
using Emgu.CV.Util;
using Emgu.CV.Structure;
using Emgu.Util;
using DirectShowLib;
```

Для того щоб відображалась форма при запуску прописав наступне:

```
private void Form1_Load(object sender, EventArgs e)
    {
        webCams = DsDevice.GetDevicesOfCat(FilterCategory.VideoInputDevice);
        for (int i = 0; i < webCams.Length; i++)
        {
            toolStripComboBox1.Items.Add(webCams[i].Name);
        }
    }
    private void toolStripLabel1_Click(object sender, EventArgs e)
    {
    }
    private void menuStrip1_ItemClicked(object sender,
ToolStripItemClickedEventArgs e)
    {
    }
```

```
    private void toolStripComboBox1_SelectedIndexChanged(object sender,
EventArgs e)
    {
          selectedCameraId = toolStripComboBox1.SelectedIndex;
      }
```

Для того щоб була змога переглядати доступні камери прописав наступне:

```
private void Capture_ImageGrabbed(object sender, EventArgs e)
    {
      try
      {
        Mat m = new Mat();
        capture.Retrieve(m);pictureBox1.Image = m.ToImage<Bgr,
byte>().Flip(Emgu.CV.CvEnum.FlipType.Horizontal).Bitmap;
      }
      catch (Exception ex)
      {
        MessageBox.Show(ex.Message, "Помилка |", MessageBoxButtons.OK,
MessageBoxIcon.Error);
      }
    }
private void toolStripButton1_Click_1(object sender, EventArgs e)
    {
      try
      {
        if (webCams.Length == 0)
        {
          throw new Exception("Немає доступних камер!");
        }
        else if (toolStripComboBox1.SelectedItem == null)
        {
```

```
                throw new Exception("Необхідно вибрати камеру!");
            }
            else if (capture != null)
            {
                capture.Start();
            }
            else
            {
                capture = new VideoCapture(selectedCameraId);
                capture.ImageGrabbed += Capture_ImageGrabbed;
                capture.Start();
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message, "Помилка |", MessageBoxButtons.OK,
MessageBoxIcon.Error);
        }
    }
```

Для того щоб була змога поставити на паузу камери прописав наступне:

```
//пауза
        private void toolStripButton2_Click(object sender, EventArgs e)
        {
            try
            {
                if (capture != null)
                {
                    capture.Pause();
                }
            }
```

```
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message, "Помилка |",
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
        }
        private void button1_Click(object sender, EventArgs e)
        {
        }
        private void pictureBox1_Click(object sender, EventArgs e)
        {

        }
        private void label1_Click(object sender, EventArgs e)
        {
        }
```

Для того щоб була змога нажати на стоп прописав наступне:

```
        private void toolStripButton3_Click(object sender, EventArgs e)
        {
            try
            {
                if (capture != null)
                {
                    capture.Pause();
                    capture.Dispose();
                    capture = null;
                    pictureBox1.Image.Dispose();
                    pictureBox1.Image = null;
                    selectedCameraId = 0;
                }
            }
```

```
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message, "Помилка |", MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
        }
        private void файлBToolStripMenuItem_Click(object sender, EventArgs e)
        {
        }
        private void вихідToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Application.Exit();
        }
```

Щоб була змога робити скріншот прописав наступне:

```
        private void toolStripButton4_Click(object sender, EventArgs e)
        {
            try
            {
                Mat m = new Mat();

                capture.Retrieve(m);

                MakeScreeenShot screenShotForm = new MakeScreeenShot(m.ToImage<Bgr, byte>().Flip(Emgu.CV.CvEnum.FlipType.Horizontal));

                screenShotForm.Show();
            }
            catch (Exception ex)
            {
```

```
                MessageBox.Show(ex.Message, "Помилка |",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                }
            }
        }
    }
```

Добавив власний іконку та логотип, добавив кольори спокійного відтінку і кінцева частина додатку виглядає наступним чином на рисунку 3.15.

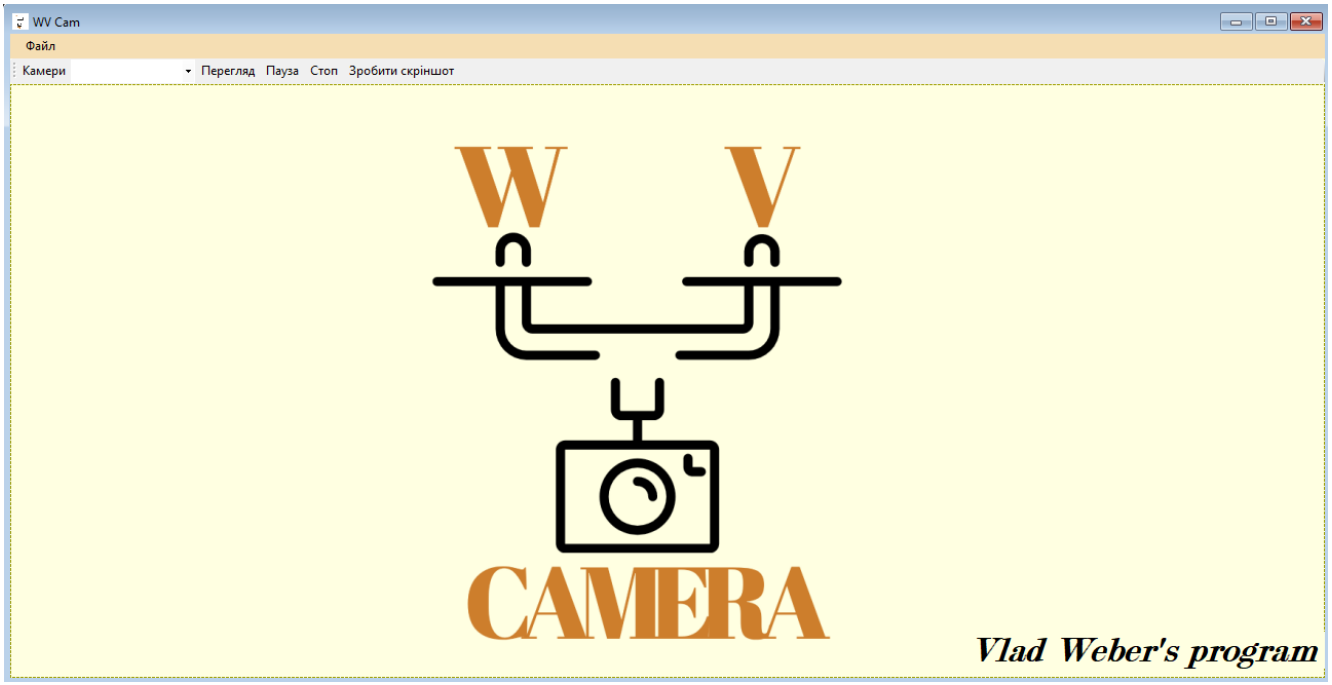

Рисунок 3.15 – Кінцевий вигляд додатку

Для додатку 2 потрібно було встановити Unity. Для початку, перейшов на сайт розробника і завантажив дистрибутив Unity, вірніше так званий web установник[7]. На головній сторінці сайту вам необхідно натиснув на кнопку «Завантажити Unity». Отже, завантажив і запустив інсталятор. Дії показані на рисунках 3.16-3.18

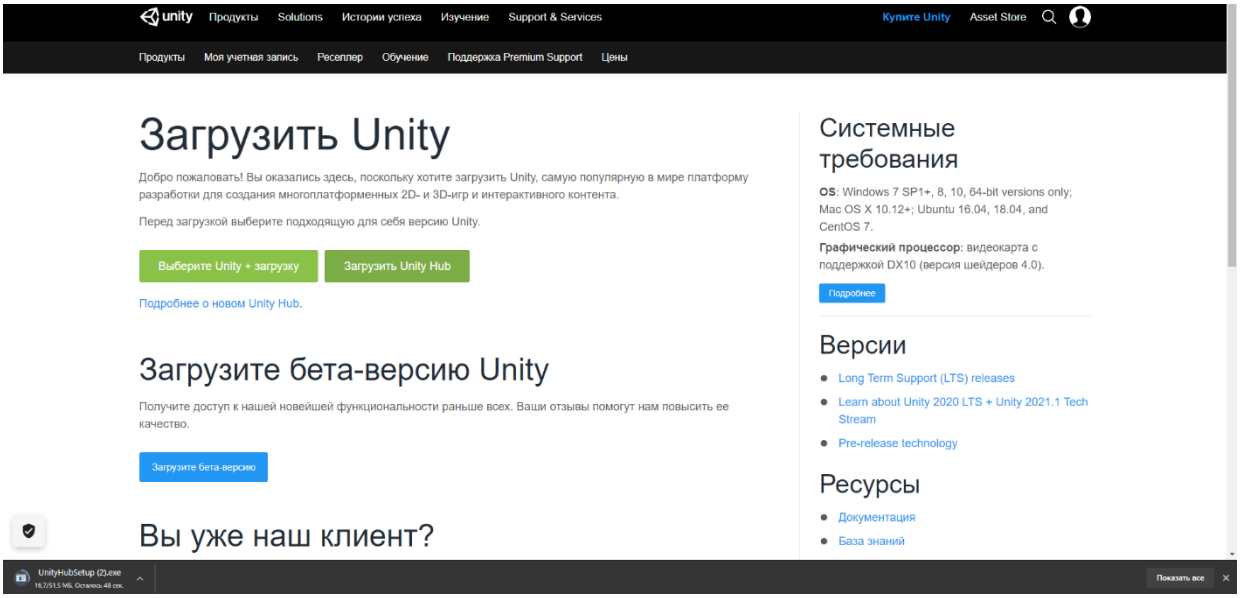

Рисунок 3.16 – Завантаження Unity

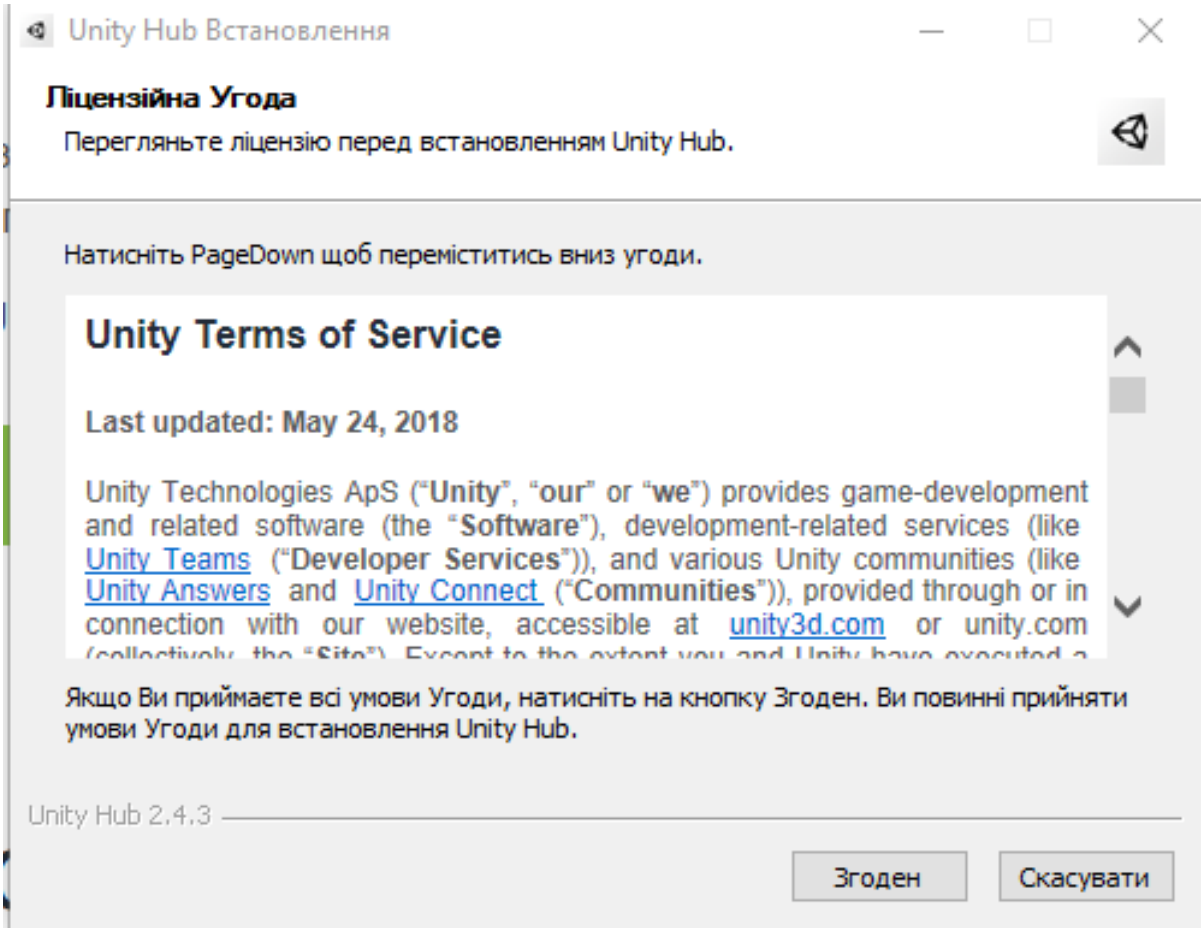

Рисунок 3.17 – узгодження з правами при завантаженні Unity

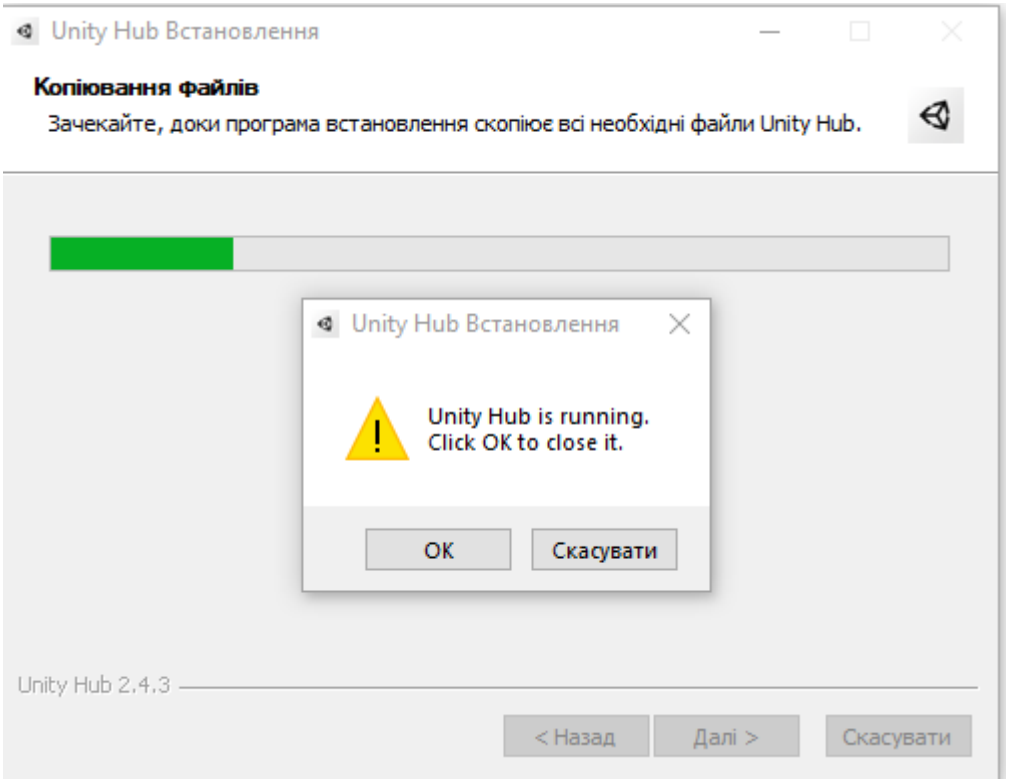

Рисунок 3.18 - Завантаження Unity

Визначившись з вибором компонентів, вибирав місце куди встановиться Unity і зачекав поки все це завантажиться.

Після всіх пройдених етапів завантаження і установки, на робочому столі з'явиться ярлик Unity, установник закрив.

Запустив двіжок і подивився що він із себе представляє. Як написано на офіційному сайті, системні вимоги для запуску Unity недостатньо великі, з підтримуваних операційних систем це «Windows 7 SP1 +, 8, 10; Mac OS X 10.8+. Windows XP і Windows Vista не підтримуються; серверні версії Windows і OS X, не пройшло перевірку. Графічний процесор: графічна карта з підтримкою DX9 (шейдерная модель 2.0). Повинні працювати будь-які карти, випущені з 2004 року. Решта залежить, головним чином, від складності проєктів ».

Далі вікно вибору проєкту, як струмових проєктів його потрібно було створити, натиснувши на кнопку NEW. Ввів назву проєкту і місце його розташування, так само вибрати 3D або 2D (за замовчуванням стоїть 3D) і натиснув Create project. Створений мною новий проєкт з'явиться у вікні Project, для його запуску досить натиснути на нього і у вас запуститься Unity, дії показанні на рисунках 3.19-3.21.

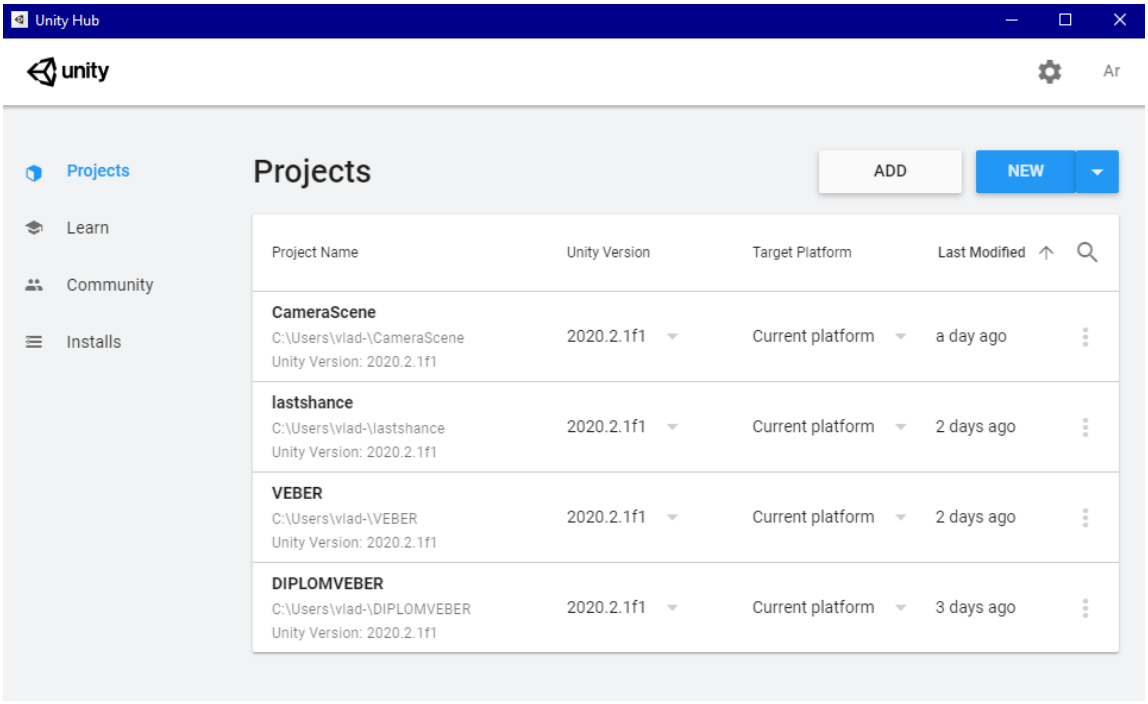

Рисунок 3.19 – Створення нового проєкту в Unity

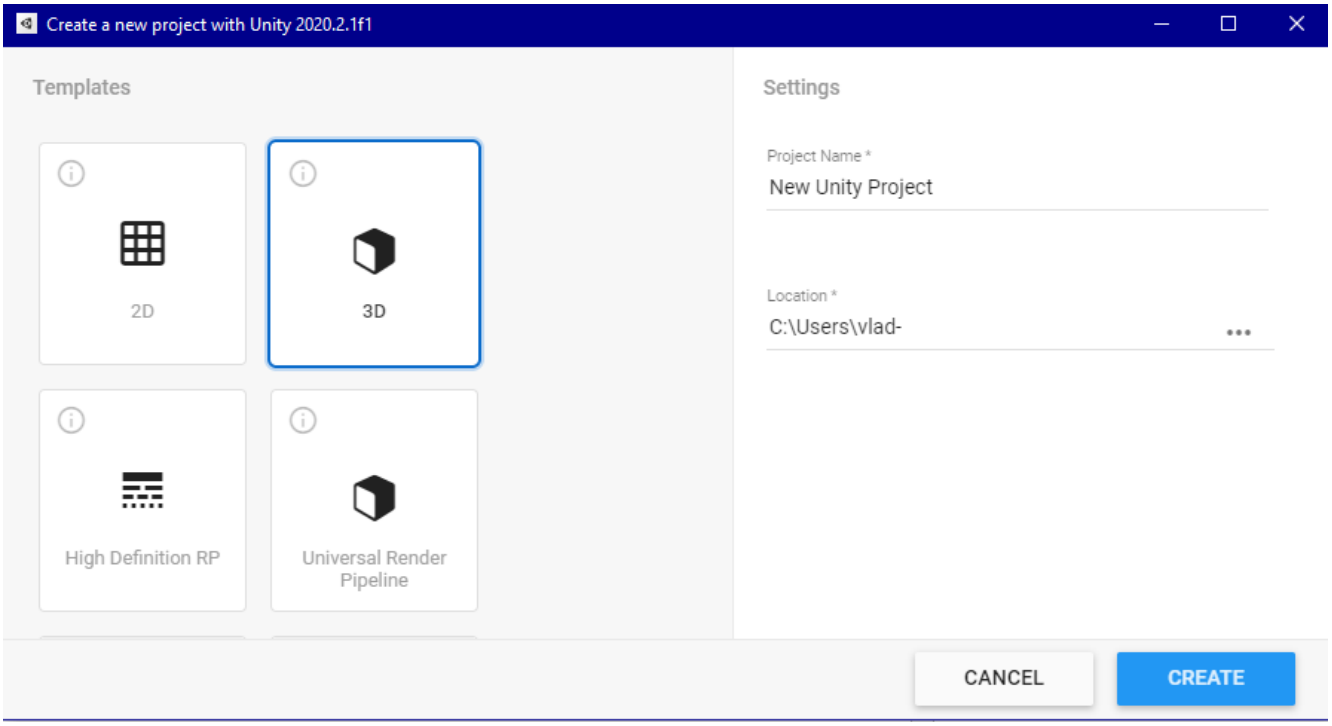

Рисунок 3.20 - Створення нового проєкту в Unity

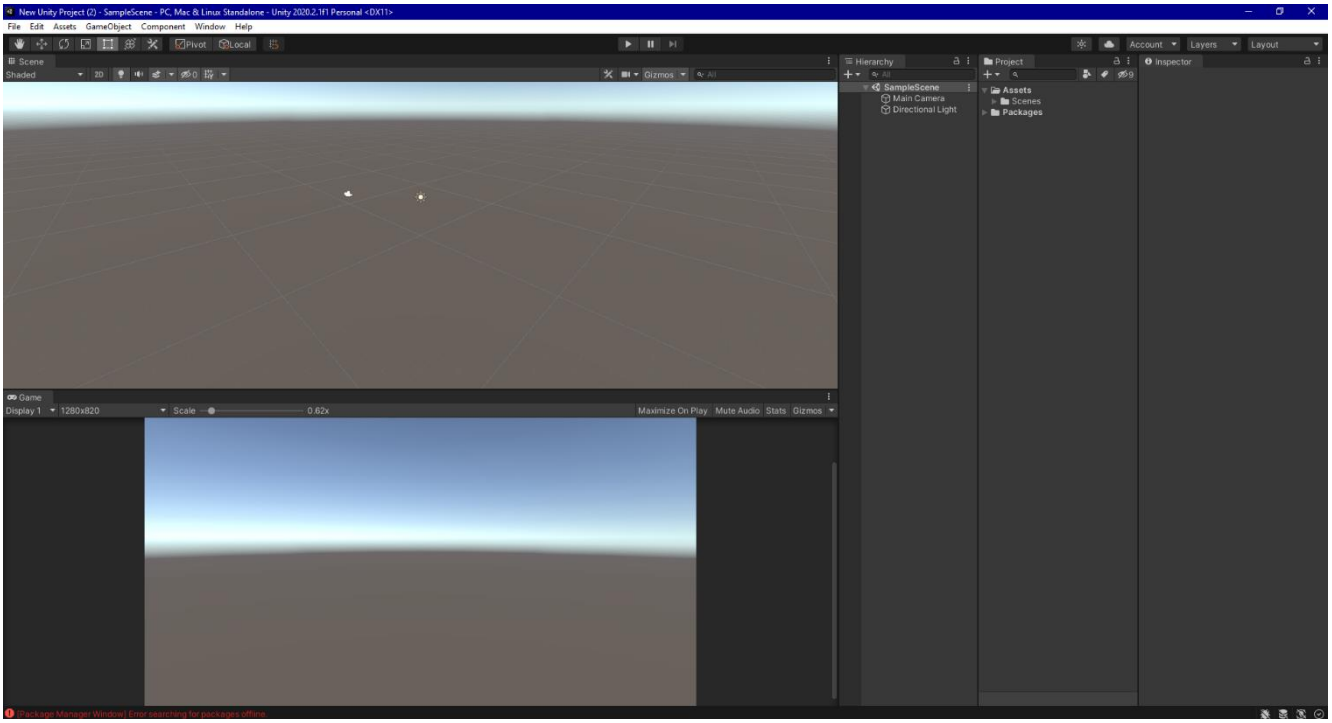

Рисунок 3.21 - Вигляд нового проєкту в Unity

Одразу ж після створення проєкту, сцену поставив в «2D» режимі аби було зручно користуватися і добавляти кнопки. Видалив «SampleScenes» в папці «Scenes» так як він не потрібний. В «Ієрархії функцій» натиснув правою кнопкою миші по таблу, і довавив в рядку «UI» «Canvas», який зразу ж показався в зображенні. Також в «Canvas», добавив 2 кнопки «Button», і перейменував їх на «Camera On» «Switch Camera». Також через праву кнопку миші добавив фон, та «Raw image» який буде показувати зображення картинки через камеру телефону. У папці «Assets» добавив нову папку з назвою «Script» і вставив C# код який назвав «CameraScript». У коді прописав бібліотеки:

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
```

Також прописав код для виводу зображення з камери телефону:

```
public class NewBehaviourScript : MonoBehaviour
{
    private bool camAvailable;
    private WebCamTexture backCam;
    private Texture defaultBackground;
```

```
    public RawImage background;
    public AspectRatioFitter fir;
    privte void Start()
    {
        defaultBackground = background.texture;
        WebCamDevice[] devices = WebCamTexture.devices;
        if(devices.Length == 0)
        {
            Debug.Log("No camera detected");
            camAvailable = false;
            return;
        }
        for (int i = 0; i < devices.Length; i++)
        {
            if (!devices[i].isFrontFacing)
            {
                backCam = new WebCamTexture(devices[i].name, Screen.width,
Screen.height);
            }
        }
        if (backCam == null)
        {
            Debug.Log("Unable to find back camera");
            return;
        }
        backCam.Play();
        background.texture = backCam;
        camAvailable = true;
    }
    private void Update()
    {
```

```
        if (!camAvailable)
            return;
        float ratio = (float)backCam.width / (float)backCam.height;
        fit.aspectRatio = ratio;
        float scaleY = backCam.videoVerticallyMirrored ? -1f : 1f;
        background.rectTransform.localScale = new Vector3(1f, scaleY, 1f);
        int orient = -backCam.videoRotationAngle;
        background.rectTransform.localEulerAngels = new Vector3(0, 0, orient);
    }
}
```

Кінцевий результат вигляду програми на Unity виглядає так як на рисунку 3.22.

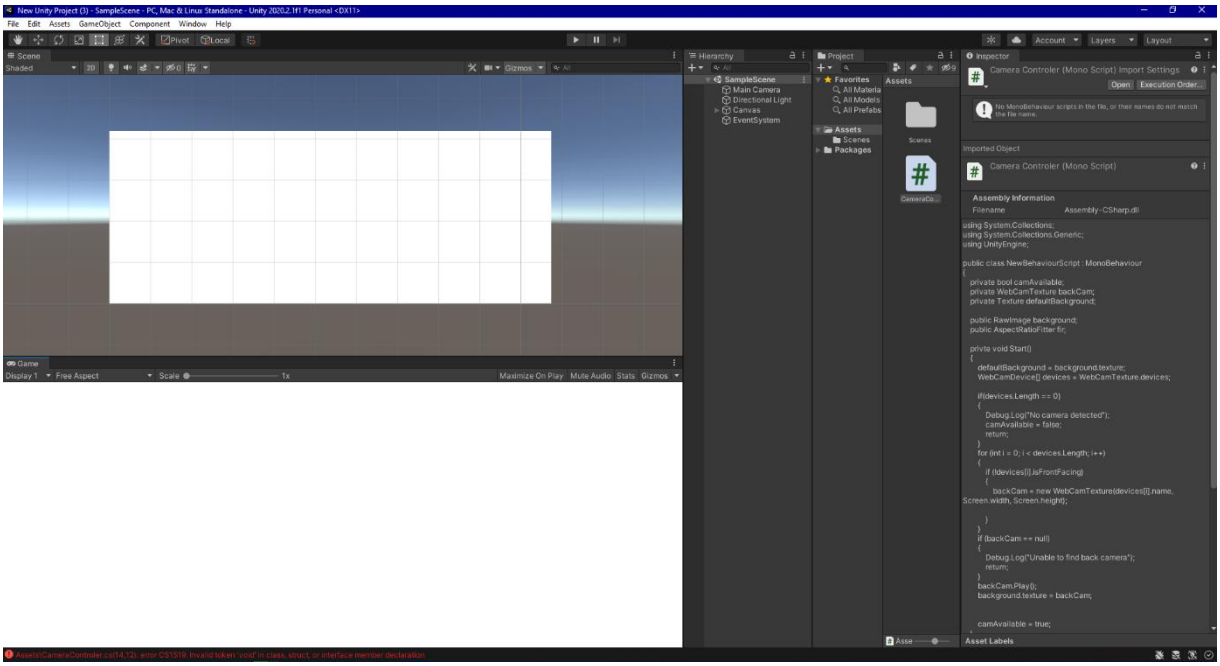

Рисунок 3.22 – Кінцевий вигляд програми на Unity

Щоб перетворити форми в додаток в файл завантаження, було перейдено до створення установочного файлу з програмою (Setup), з можливістю його

подальшої установки на різних комп'ютерах. На першому етапі створення установочного файлу, було названо програмний продукт «WV Camera», задано його версію – v 1.0, та розробника цієї програми – WBR, на рисунку 3.23.

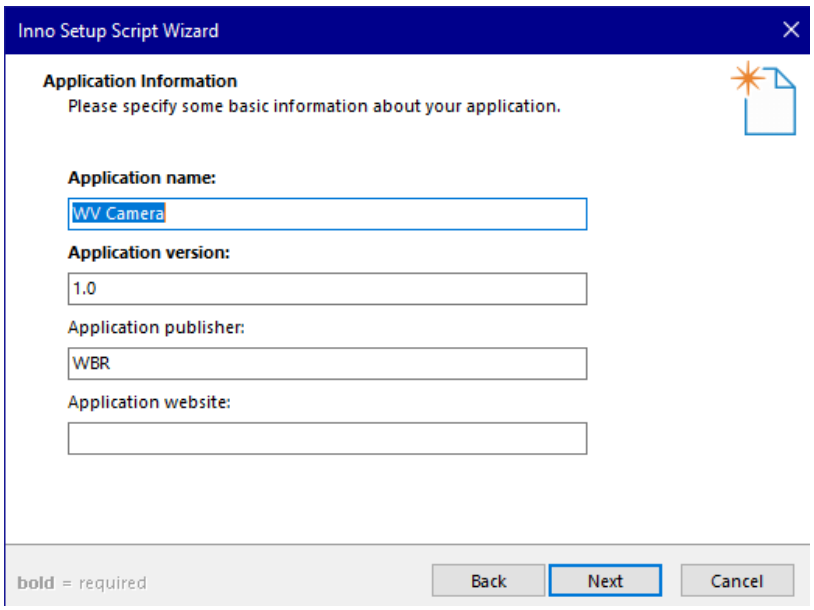

Рисунок 3.23 – Початок створення установочного файлу

Далі вводилося ім'я додатку та розташування та вибір файлу .exe в папці з програмою на рисунку 3.24.

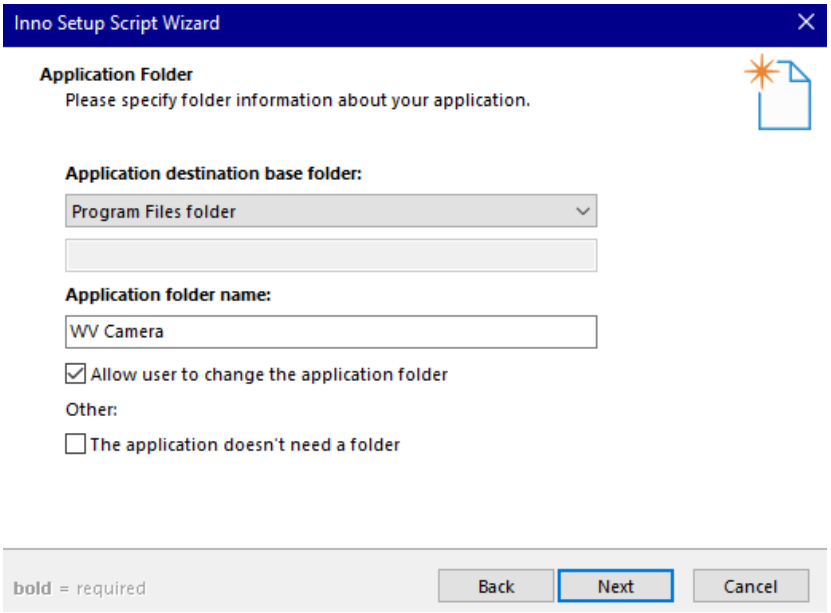

Рисунок 3.24 – Другий етап створення установочного файлу

Після цього в установщику вибрав файл .exe програми з якого програма повина була б запускатись після установки на комп'ютері та папка, де знаходиться програма на рисунку 3.25.

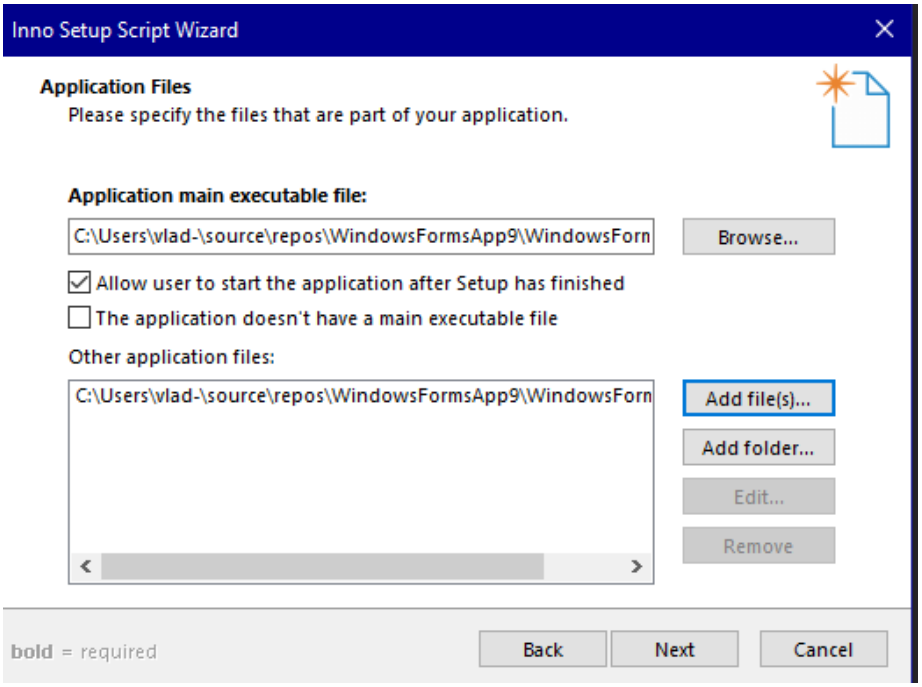

Рисунок 3.25 – Вибір файлу .exe, та папки з програмою

Після вибору .exe програми з'являється вікно в, якому потрібно вибрати папку «Resourses», яка знаходиться в папці з програмою на рисунку 2.26.

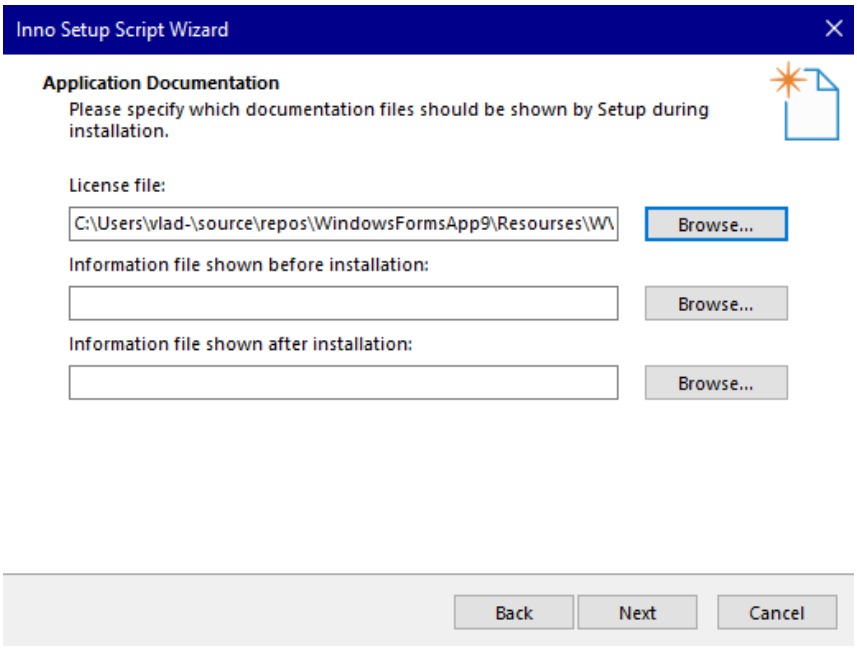

Рисунок 3.26 – Вибір папки Resources програми

В наступному вікні вибино з підпунктів, інсталяцію для адміністратора програми на рисунку 2.27

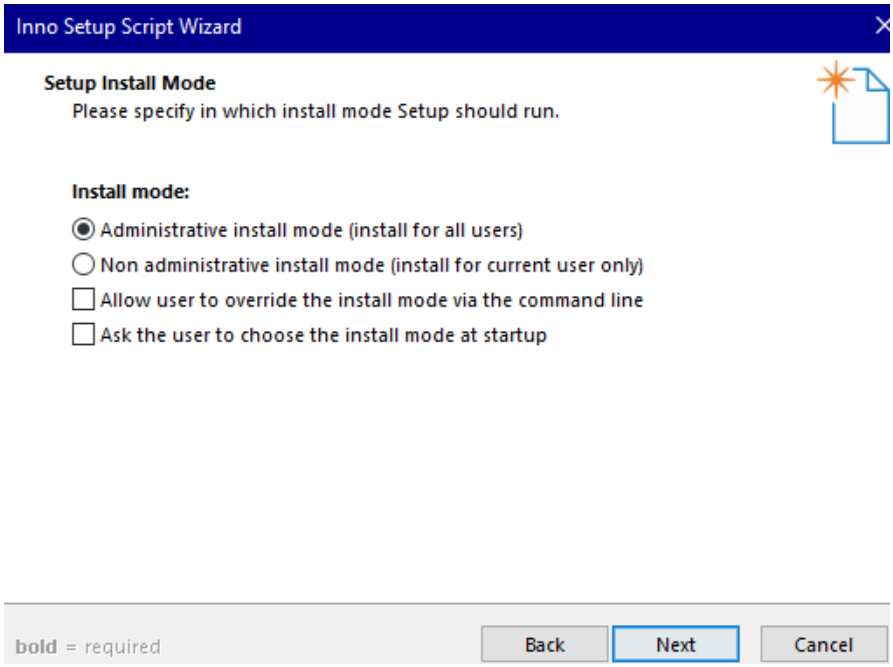

Рисунок 3.27 – Вибір інсталяції для адміністратора програми

Наступним етапом було вибір мови установочного пакета, вибрано українську,російську та англійську мови на рисунку 2.28.

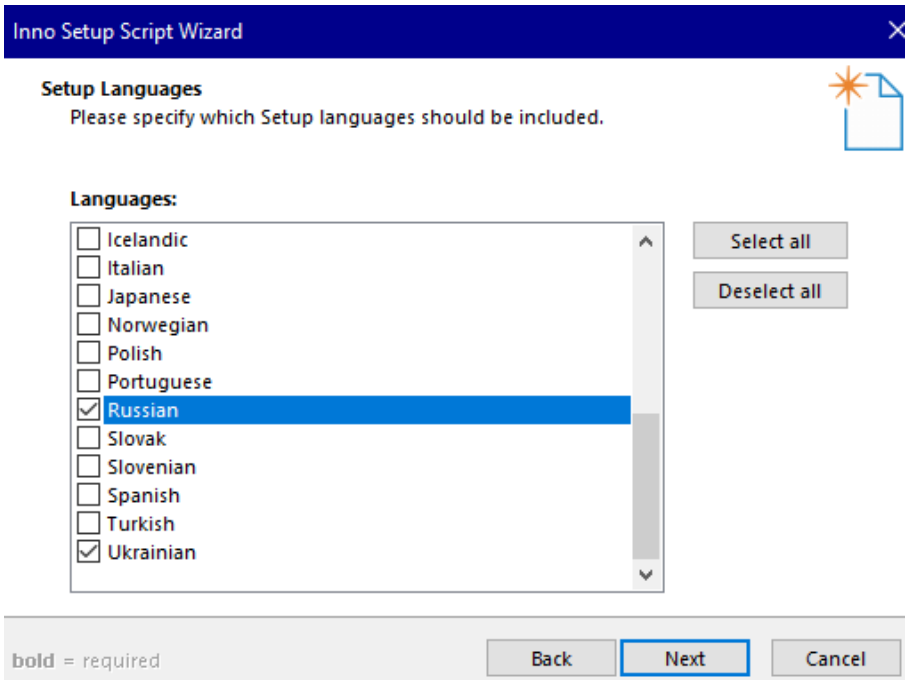

Рисунок 3.28 – Вибір мови установочного пакету

В наступному вікні вибирав файл з логотипом, який буде використовуватись в програмі і завершив установку на рисунках 2.29,2.30.

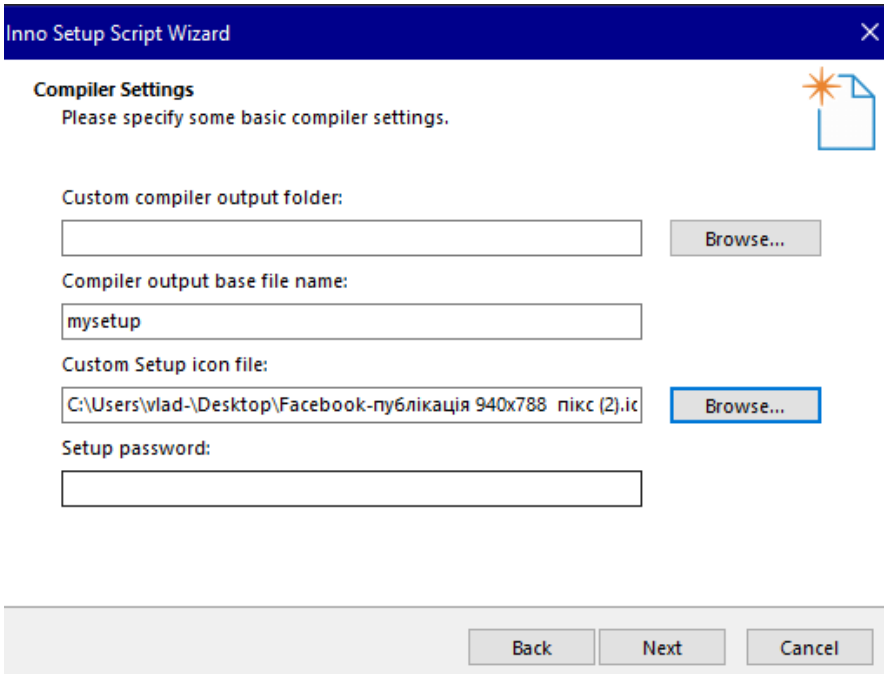

Рисунок 3.29 – Вибір логотипу установки

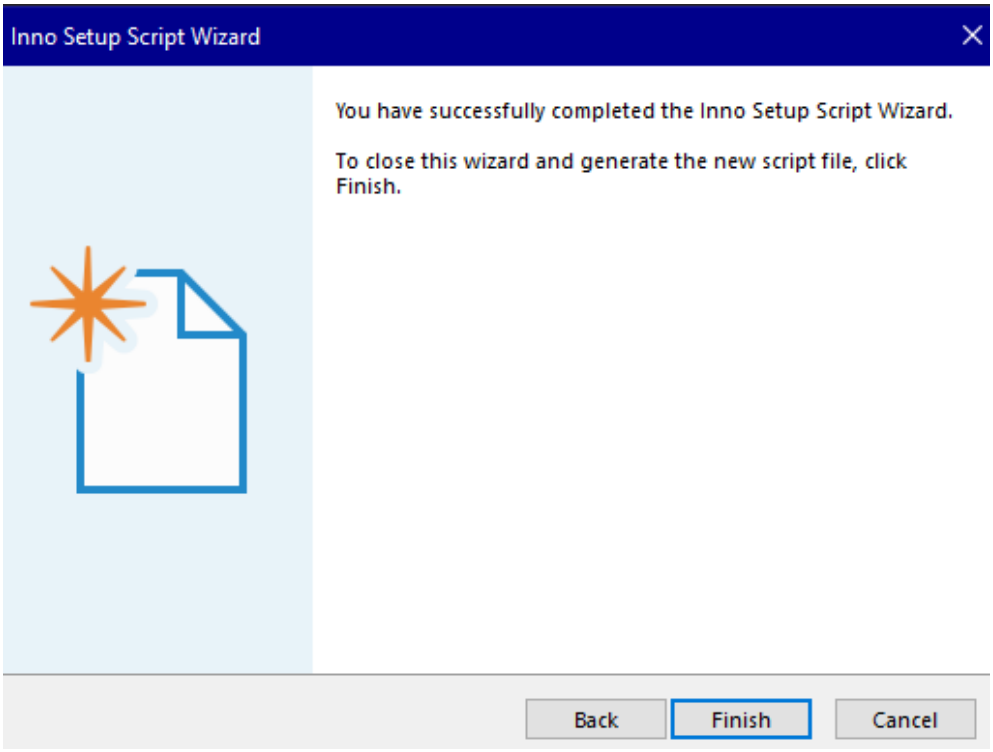

Рисунок 3.30 – Завершення установки

Далі в програмі було запущено скрипт на виконання, в результаті якого на робочому столі з’явився установочний файл з назвою setup на рисунку 3.31.

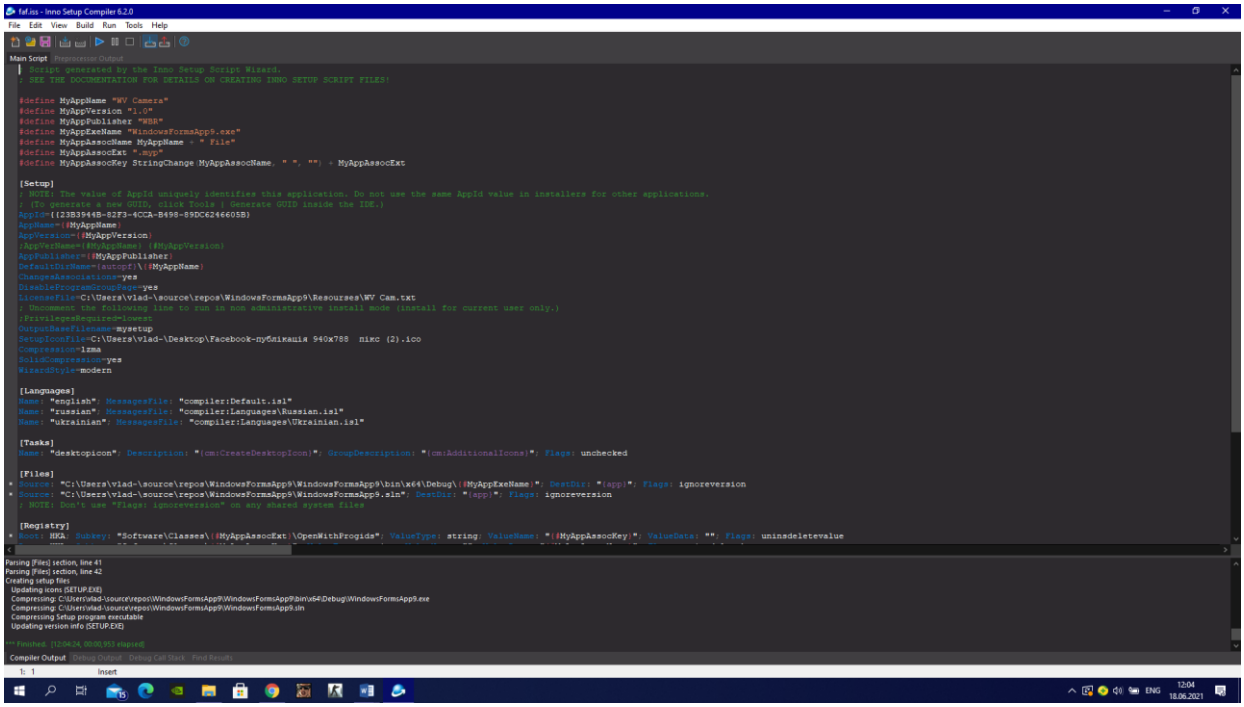

Рисунок 3.31 – Вигляд програми установки

3.3 Інструкції користувача

Для роботи з додатком для тестування – «WV Camera», потрібно завантажити установочний файл з програмою – «setupprogram», та відкрити його, з’явиться вікно для встановлення програми. Слідкуючи інструкціям установщика, у має успішно програма завантажитись та запуститись. Першим з’явится вікно установщика, у якому потрібно буде вибрати мову встановлення програми на рисунку 3.32.

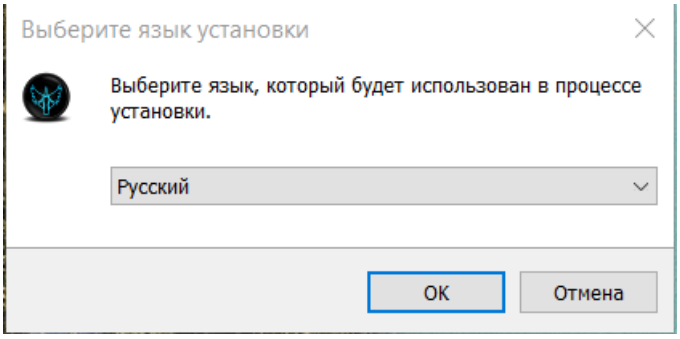

Рисунок 3.32 – Вікно вибору мови установки

Наступним з’явиться вікно, яке попросить прочитати ліцензійну угоду, та вибрати пункт, щоб прийняти умови ліцензійної угоди, та перейти далі до установки програми. У разі не прийняття ліцензійної угоди, можна продовжувати установку програмного забезпечення. Ліцензійна угода на даний етап розробки програми доступна на любій мові. Ліцензійна угода звучить так:

Дякуємо за встановлення програми замовлення – WV Camera.

Програма передбачає автоматизоване управління замовленнями.

Копіювати, передавати або встановлюйти окремі частини, або всю програму заборонено третім особам, без відома розробника на рисунку 3.33.

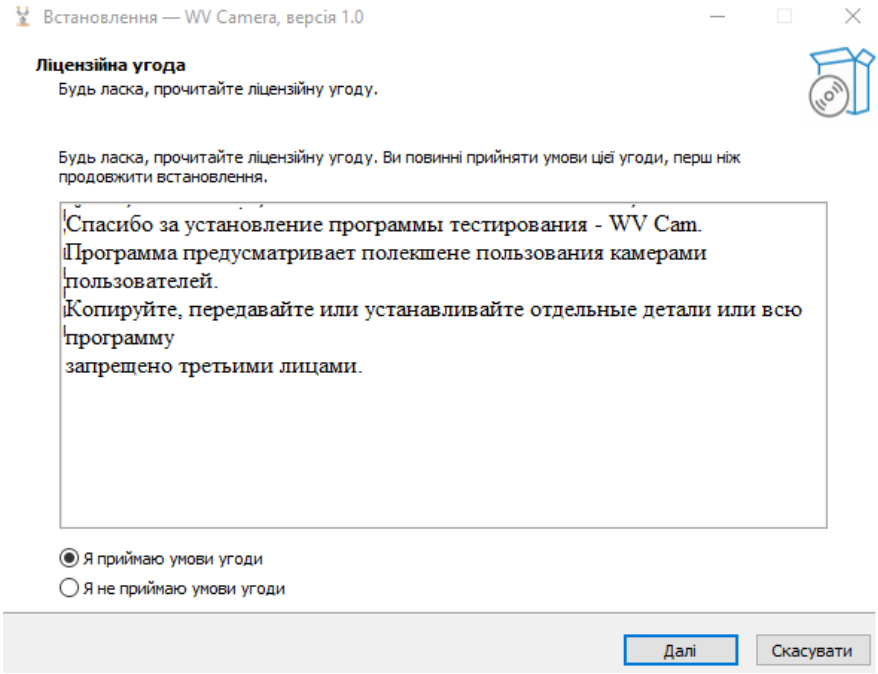

Рисунок 3.33 – Підтвердження ліцензійної угоди

В наступному вікні вибираєм папку, куда буде встановлена програма. І показано скільки місця буде вона займати на комп’ютері на рисунку 3.34.

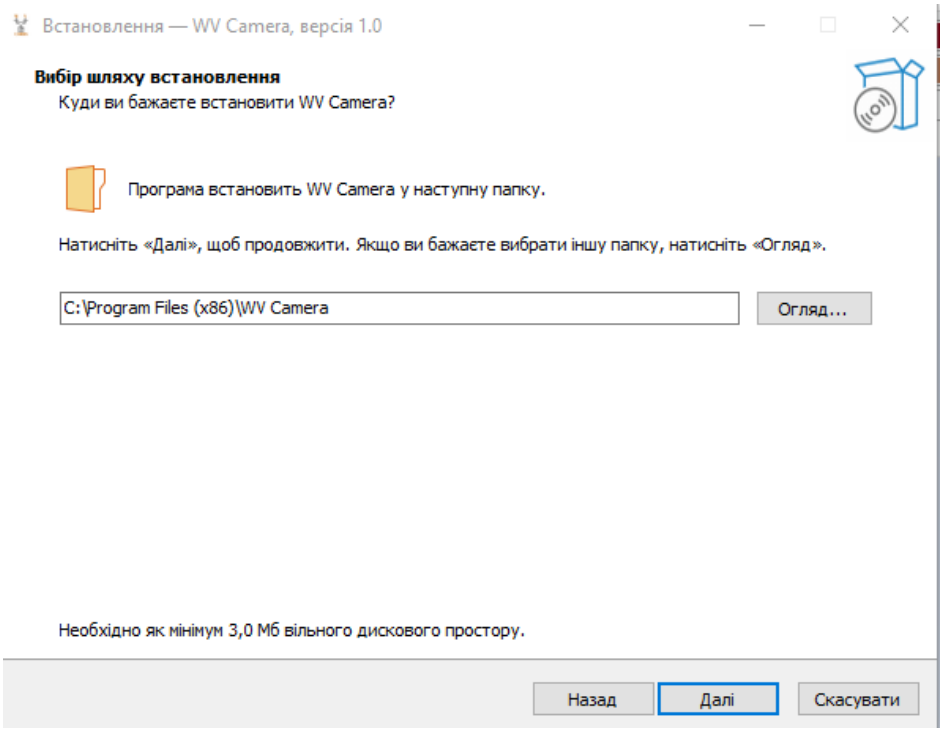

Рисунок 3.34 – Вибір папки для створення програми

Далі з’являється вікно де запитується чи потрібно створювати папку в меню «Пуск» на рисунку 3.35.

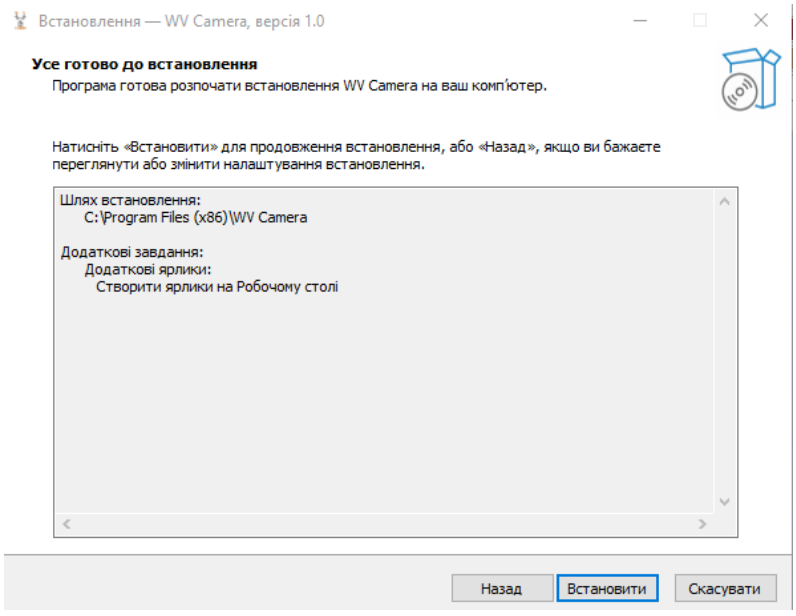

Рисунок 3.35 – Вибір створення ярлика в меню «Пуск»

Після прийняття ліцензійної угоди, з’явитись вікно, яке дозволить створити ярлик .exe файлу запуска програми, на робочому столі. У разі

відхилення цієї функції, можна запустити програму з папки на диску, який буде вибраний у процесі установки на рисунку 3.36.

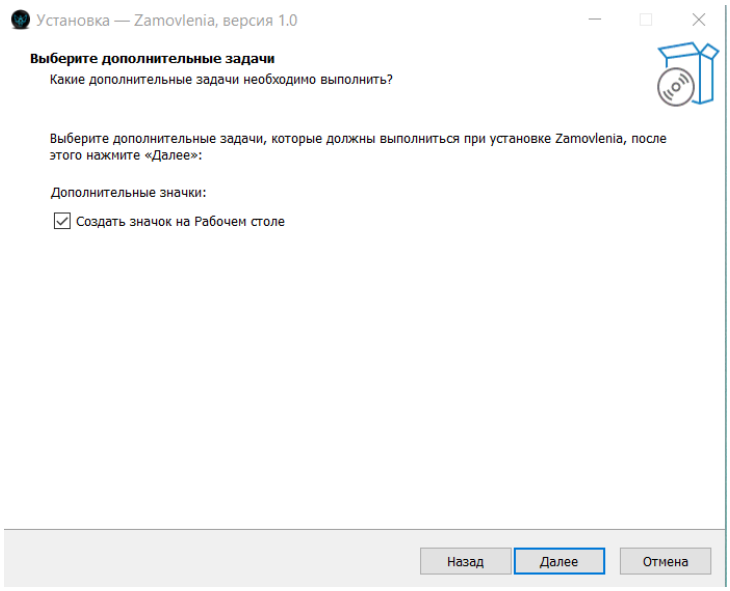

Рисунок 3.36 – Вибір створення ярлика на робочому столі

Після цих дій було вибрано назву установочного пакету – setup Zamovlenia, та вибрано раніше створену іконку для програми в форматі .ico, та завершено створення установочного пакету з програмою на рисунку 3.37-3.38.

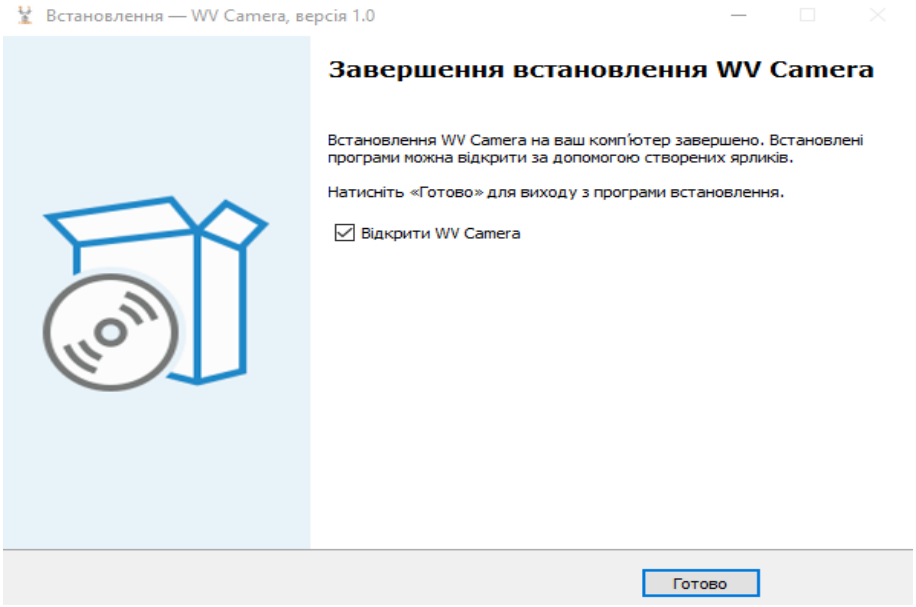

Рисунок 3.27 – Завершення створення установочного пакету

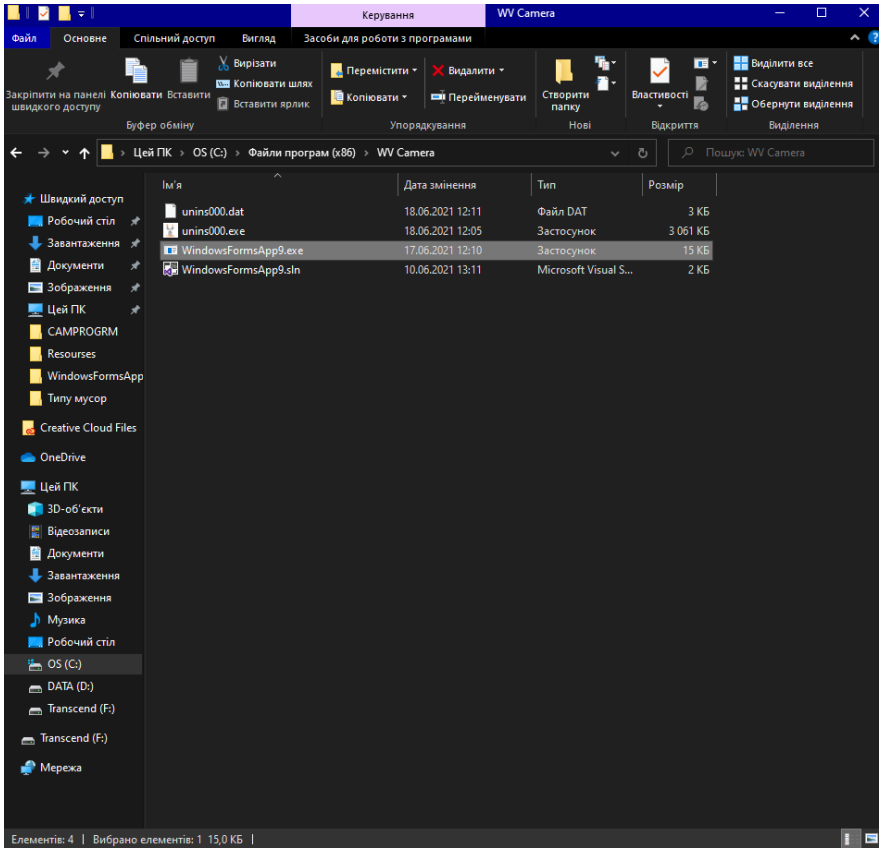

Рисунок 3.38 – Папка, в якій була встановлена програма

Після запуску програми з'являється головне меню додатка (рис. 3.39). Перехід до даних відбувається при натисканні кнопок меню (Перегляд, Пауза, Стоп, Зробити скріншот). На кожній з форм знаходиться таблиця або поля з даними та відповідні кнопки управління: перехід по записам, додавання даних, видалення, збереження та пошук запису.

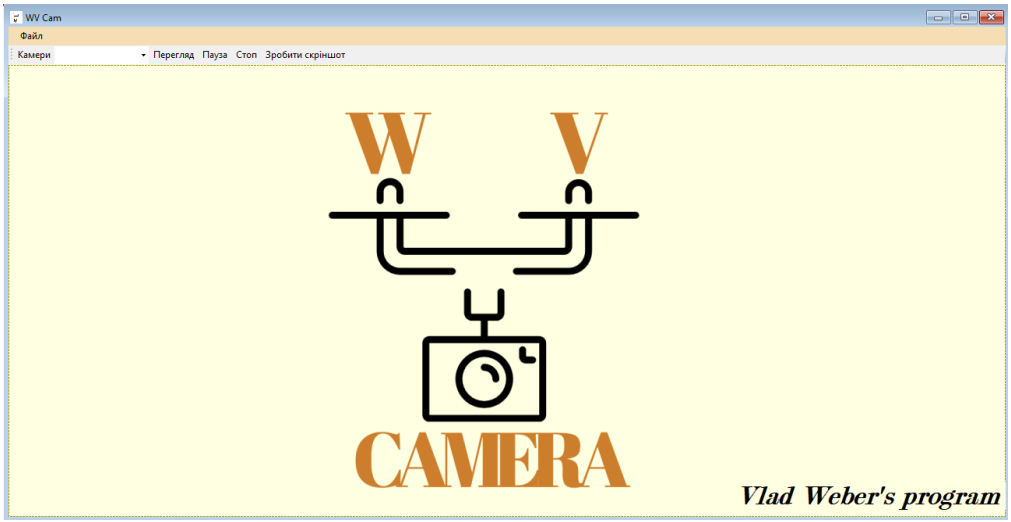

Рисунок 3.39 – Головне меню програми

Меню додатку для камер містить такі пункти: Перегляд камер, пауза, стоп та зробити скріншот.

Пункт меню «Камера», який служить для вибору камер які підключенні до ноутбуку. Після відкриття додатку, натиснути на підпункт що знаходиться біля «Камери», та вибираємо підключену до вашого ноутбуку камеру.Після цього натиснувши на «Перегляд» відобразиться зображення на рисунках 3.40-3.41.

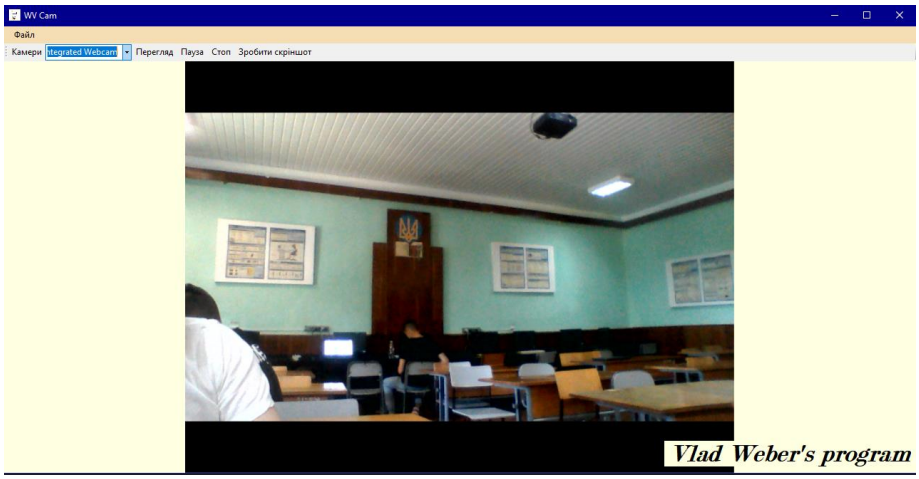

Рисунок 3.40 – Перегляд першої камери

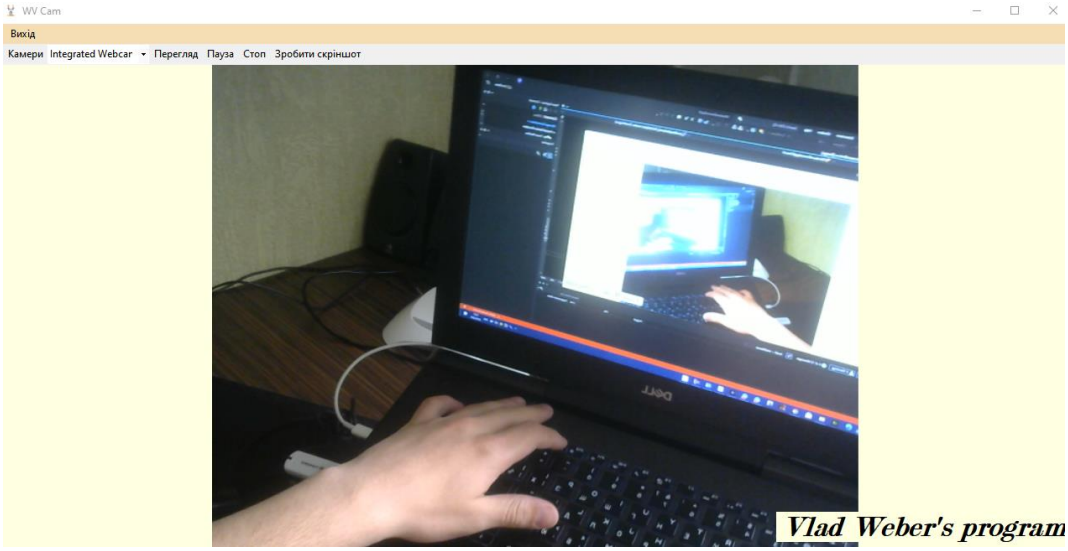

Рисунок 3.41 – Перегляд другої камери

Кнопка «Пауза», дозваляє зупинити картинку яка відображалась.Кнопка «Стоп», зупиняє програму та перекидає в головне меню.Кнопка «Зробити скріншот» робить скріншот зображення.

Для другого додатку по аналогії те ж саме, тільки для цього потрібно завантажити додаток від розробника для телефону «Unity» і підключити через

usb – кабель. Давши дозвіл на використання камер і картинку буде зображено з камери телефону.

У цьому розділі було розглянуто процес розробки власного програмного рішення для відеоспостереження, вибір відповідного середовища розробки та тестування. Реалізоване програмне забезпечення дозволяє ефективно здійснювати моніторинг, автоматизувати процеси аналізу відео та підвищити рівень безпеки об’єкта. Описані технології та інструменти забезпечують стабільність роботи системи та її адаптацію до різних умов експлуатації.

ВИСНОВКИ

Підсумком виконання дипломного проєкту стала розробка системи віддаленого відеоспостереження через телефон та інтернет для ВСП «Кам'янець-Подільський фаховий коледж харчової промисловості НУХТ».

При вирішенні завдання розробки програмного забезпечення, було досліджено і проглянуто велика кількість існуючих аналогів на помилках яких було створенний власний додаток, який зумовлює покращення та полегшення у використанні власних камер для відеоспостереження або у стрімингових сервісах для прямого ефіру. Програми з простим та зрозумілим інтерфейсом дозволить легко впорюватись навіть в надскладних ситуаціях, а її реалізація в житті зумовлює підключати любий тип камер.

Сучасний потоковий тип відеонагляду дозволить показувати абсолютно все : від новин в прямому ефірі, зон де перебуває відеоспостережня камеп до класичних фільмів і останніх серіальних новинок, у будь-який час, з будь-якого пристрою.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Kapp, K. M. The gamification of learning and instruction: Game-based methods and strategies for training and education. Pfeiffer, 2012. 352 p.
2. Werbach, K., & Hunter, D. For the win: How game thinking can revolutionize your business. Wharton Digital Press, 2012. 150 p.
3. Domínguez, A., Saenz-de-Navarrete, J., de-Marcos, L., Fernández-Sanz, L., Pagés, C., & Martínez-Herráiz, J. J. Gamifying learning experiences: Practical implications and outcomes. Computers & Education, 2013, vol. 63, pp. 380–392.
4. Запорожець, О.І. Гра як засіб мотивації у освіті. Наукові записки ЗНУ, 2020, т. 45, с. 34–39.
5. Kim, S. The effects of gamification on motivation and engagement in e-learning. British Journal of Educational Technology, 2015, vol. 46, no. 5, pp. 1000–1010.
6. Zichermann, G., & Cunningham, C. Gamification by design: Implementing game mechanics in web and mobile apps. O'Reilly Media, Inc., 2011. 244 p.
7. Детердинг, С., Діксон, Д., Халед, Р., & Наке, Л. Елементи дизайну гри як основа гейміфікації. Науковий вісник цифрових медіа, 2011, с. 15–21.
8. Hamari, J., Koivisto, J., & Sarsa, H. Does gamification work? A literature review of empirical studies on gamification. Proceedings of the 47th Hawaii International Conference on System Sciences, IEEE, 2014, pp. 3025–3034.
9. Ніколсон, С. Користувацький підхід до осмисленої гейміфікації. Ігри + Навчання + Суспільство 8.0, 2012, 15 с.
10. Lee, J. J., & Hammer, J. Gamification in education: What, how, why bother? Academic Exchange Quarterly, 2011, vol. 15, no. 2, p. 1.
11. Przybylski, A. K., Rigby, C. S., & Ryan, R. M. A motivational model of video game engagement. Review of General Psychology, 2010, vol. 14, no. 2, pp. 154–166.

12. Ryan, R. M., & Deci, E. L. Intrinsic and extrinsic motivations: Classic definitions and new directions. Contemporary Educational Psychology, 2000, vol. 25, no. 1, pp. 54–67.
13. Huizinga, J. Homo ludens: A study of the play-element in culture. Routledge, 2014. 220 p.
14. Пренскі, М. Навчання через цифрові ігри. Видавництво "Освіта майбутнього", 2015. 280 с.
15. Загородня, Н.М. Вплив гейміфікації на мотивацію студентів. Вісник ЗНУ, 2018, т. 12, с. 98–105.
16. Kapp, K. M., & O’Driscoll, T. Learning in 3D: Adding a new dimension to enterprise learning and collaboration. Pfeiffer, 2010. 396 p.
17. Білобровко, Т.І. Гейміфікація у сучасній освіті. Освітні студії, 2019, т. 2, с. 15–20.
18. Khan Academy Case Study. Stanford Social Innovation Review. URL: https://hbr.org/podcast/2024/11/khan-academy-a-case-study-in-scaling-a-start-up (дата звернення: 07.10.2024).
19. ClassDojo: A Case Study in Gamification for Education. EdSurge. URL: https://www.edsurge.com/news/2014-07-23-classdojo-adds-shared-classes-and-students-features (дата звернення: 11.10.2024).
20. Gamification in Duolingo: How it Helps Millions Learn Languages. Duolingo Blog. URL: https://medium.com/@digintostartups/duolingo-revolutionizing-language-learning-through-gamification-9e8ae25f2bf8 (дата звернення: 19.10.2024).
21. Hakulinen, L., Auvinen, T., & Korhonen, A. The effect of gamification on students’ learning and motivation: A systematic literature review. Educational Psychology Review, 2015, vol. 27, no. 4, pp. 723–752.
22. Литвиненко, К.О. Роль гейміфікації в цифровому навчанні. Освітня платформа України, 2020, с. 45–52.
23. Seaborn, K., & Fels, D. I. Gamification in education: What, how, why bother? Simulation & Gaming, 2015, vol. 46, no. 2, pp. 147–172.

ДОДАТКИ

Додаток А

Лістинги програми

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
```

Також прописав код для виводу зображення з камери телефону:

```
public class NewBehaviourScript : MonoBehaviour
{
   private bool camAvailable;
   private WebCamTexture backCam;
   private Texture defaultBackground;
   public RawImage background;
   public AspectRatioFitter fir;
   privte void Start()
   {
      defaultBackground = background.texture;
      WebCamDevice[] devices = WebCamTexture.devices;
      if(devices.Length == 0)
      {
         Debug.Log("No camera detected");
         camAvailable = false;
         return;
      }
      for (int i = 0; i < devices.Length; i++)
      {
         if (!devices[i].isFrontFacing)
         {
```

```
            backCam = new WebCamTexture(devices[i].name, Screen.width, Screen.height);
        }
    }
    if (backCam == null)
    {
        Debug.Log("Unable to find back camera");
        return;
    }
    backCam.Play();
    background.texture = backCam;
    camAvailable = true;
  }
  private void Update()
  {
    if (!camAvailable)
        return;
    float ratio = (float)backCam.width / (float)backCam.height;
    fit.aspectRatio = ratio;
    float scaleY = backCam.videoVerticallyMirrored ? -1f : 1f;
    background.rectTransform.localScale = new Vector3(1f, scaleY, 1f);
    int orient = -backCam.videoRotationAngle;
    background.rectTransform.localEulerAngels = new Vector3(0, 0, orient);
  }
}
```