

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЗАПОРІЗЬКА ПОЛІТЕХНІКА»

Факультет комп'ютерних наук та технологій
Кафедра комп'ютерних систем та мереж

Пояснювальна записка

до дипломного проекту (роботи)

магістра

(ступінь вищої освіти)

на тему: Віртуальна комп'ютерна лабораторія кафедри комп'ютерних
систем та мереж
(назва теми)

Виконав: студент 2 курсу, групи КНТ-612м
Спеціальності 123 Комп'ютерна інженерія
(код і найменування спеціальності)

Освітня програма (спеціалізація)
«Спеціалізовані комп'ютерні системи»

ЛАВРИК В.Р.

(ПРИЗВИЩЕ та ініціали)

Керівник ТЯГУНОВА М.Ю.

(ПРИЗВИЩЕ та ініціали)

Рецензент САМОЙЛИК С.С.

(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет _____ комп'ютерних наук і технологій
 Кафедра _____ комп'ютерних систем та мереж
 Ступінь вищої освіти _____ магістерський
 Спеціальність _____ 123 Комп'ютерна інженерія
 (код і найменування)
 Освітня програма (спеціалізація) _____ Спеціалізовані комп'ютерні системи
 (назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри _____ Кудерметов Р.К.

«_____» _____ 2023 року

З А В Д А Н Н Я
НА МАГІСТРЕСЬКУ РОБОТУ СТУДЕНТА

Лаврика Владислава Руслановича

(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) _____ Віртуальна комп'ютерна лабораторія кафедри
 комп'ютерних систем та мереж

керівник проєкту (роботи) _____ канд.техн.наук, доцент Тягунова Марія Юріївна
 (науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від « 24 » жовтня 20 23 року № 400

2. Строк подання студентом проєкту (роботи) _____ 11 грудня 2023 р.

3. Вихідні дані до проєкту (роботи) _____ Апаратна база аудиторії №52: процесор -
 AMD Ryzen 5 PRO 4650G; материнська плата ASUS TUF GAMING A-520M -
 PLUS; оперативна пам'ять - Kingston HYPERX 2 планки по 16Gb; пам'ять SSD -
 Samsung 970EVO Plus; пам'ять HDD - Toshiba PC P300 1TB; робот-пилосос
 iRobot Create 2 Roomba 600.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Аналіз предметної області;

2. Розробка завдань для виконання у віртуальній лабораторії;

3. Розробка віртуальної лабораторії;

4. Тестування роботи лабораторії.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, кількість слайдів, плакатів)

6. Консультанти розділів проєкту (роботи)

Розділ	ПРИЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4	ТЯГУНОВА М.Ю., доцент каф. КСМ		
нормоконтроль	ЩЕРБАК Н.В., ст. викл. каф. КСМ		

7. Дата видачі завдання « 04 » вересня 2023 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Аналіз предметної області, визначення мети і завдань віртуальної лабораторії	10.09.2023 р.	
2	Аналіз існуючих технологічних рішень	15.09.2023 р.	
3	Аналіз вимог до віртуальної лабораторії	22.09.2023 р.	
4	Розробка завдань для виконання у віртуальній лабораторії	05.10.2023 р.	
5	Рекомендації з виконання завдань	12.10.2023 р.	
6	Розробка віртуальної лабораторії	19.10.2023 р.	
7	Вибір програмного та апаратного забезпечення системи	27.10.2023 р.	
8	Налаштування роботи системи	10.11.2023 р.	
9	Тестування роботи лабораторії	18.11.2023 р.	
10	Оформлення отриманих результатів у ПЗ	01.12.2023 р.	

Студент


(підпис)

Владислав ЛАВРИК
(ім'я, ПРИЗВИЩЕ)

Керівник проєкту (роботи)


(підпис)

Марія ТЯГУНОВА
(ім'я, ПРИЗВИЩЕ)

РЕФЕРАТ

ПЗ: 76 с., 57 рис., 2 таблиці, 23 посилання, 4 додатки.

ВІДДАЛЕНИЙ ДОСТУП, ВІРТУАЛЬНА ЛАБОРАТОРІЯ, ВИКЛАДАЧ,
ПРОГРАМУВАННЯ, РОБОТ-ПИЛОСОС, СТУДЕНТ, IROBOT CREATE

Об'єкт дослідження - віртуальні комп'ютерні лабораторії.

Предмет дослідження - програмне та апаратне забезпечення для створення та роботи віртуальної лабораторії кафедри комп'ютерних систем та мереж.

Мета роботи - допомогти студентам продовжувати навчання та отримувати практичні навички у будь-якій локації: за кордоном, чи у бомбосховищі, чи на тимчасово окупованій території України шляхом розробки віртуальної лабораторії кафедри комп'ютерних систем та мереж.

У першому розділі було проведено аналіз предметної області, визначено цілі та задачі, які потрібно розглянути для створення віртуальної лабораторії. Також було проаналізовано програмне забезпечення для віддаленого доступу до лабораторії.

У другому розділі безпосередньо розроблено завдання, які можна виконувати у створеній віртуальній лабораторії та які корелюють з напрямками освітніх програм, за якими відбувається навчання на кафедрі комп'ютерних систем та мереж.

У третьому розділі було розроблено структурну та описано функціональну схему віртуальної лабораторії, спираючись на напрями розроблених завдання. Також було запропоновано та обґрунтовано вибір апаратної та програмної складової віртуальної лабораторії в межах університету та у віддалених користувачів, розроблено та описано алгоритм функціонування системи загалом.

У четвертому розділі було продемонстровано результати виконання розроблених завдань для виконання у віртуальній лабораторії та проаналізовано отримані результати.

ABSTRACT

EN: 76 p., 57 pictures, 2 tables, 23 references, 4 appendices.

**IROBOT CREATE, PROGRAMMING, REMOTE ACCESS, ROBOT VACUUM
CLEANER, STUDENT, TEACHER, VIRTUAL LABORATORY**

Object of research - virtual computer laboratories.

The subject of research - software and hardware for creating and operating a virtual laboratory of the Department of Computer Systems and Networks.

The purpose of the work is to help students continue their studies and gain practical skills in any location: abroad, in a bomb shelter, or in the temporarily occupied territory of Ukraine by developing a virtual laboratory of the Department of Computer Systems and Networks.

The first section analyzes the subject area, identifies the goals and objectives that need to be considered for the creation of a virtual laboratory. The software for remote access to the laboratory was also analyzed.

In the second section, the tasks that can be performed in the created virtual laboratory and that relate to the areas of educational programs that are studied at the Department of Computer Systems and Networks were developed.

In the third section, we developed a structural and functional diagram of the virtual laboratory based on the areas of the developed tasks. Also, the choice of hardware and software components of the virtual laboratory within the university and at remote users was proposed and justified, and the algorithm of the system's functioning in general was developed and described.

In the fourth section, the results of the developed tasks for the virtual laboratory were demonstrated and the results were analyzed.

ЗМІСТ

Вступ.....	8
1 Аналіз предметної області.....	10
1.1 Визначення мети і завдань віртуальної лабораторії.....	10
1.2 Аналіз існуючих технологічних рішень	11
1.2.1 Платформа віддаленого доступу TeamViewer	11
1.2.2 Віддалений робочий стіл Chrome	13
1.2.3 Платформа віддаленого доступу AnyDesk	14
1.2.4 Результати аналізу програмного забезпечення	16
1.3 Аналіз вимог до віртуальної лабораторії.....	17
1.3.1 Урахування потреб користувачів	17
1.3.2 Вимоги доступу до віртуальної лабораторії.....	18
2 Розробка завдань для виконання у віртуальній лабораторії.....	18
2.1 Завдання у програмному середовищі віртуальної лабораторії	19
2.2 Використання специфічного обладнання віртуальної лабораторії.....	27
2.2.1 Апаратне забезпечення	28
2.2.3 Програмне забезпечення.....	34
2.2.4 Рекомендації з виконання завдань.....	39
3 Розробка віртуальної лабораторії	47
3.1 Структурна схема лабораторії	47
3.2 Функціонування лабораторії.....	51
3.3 Вибір програмного та апаратного забезпечення системи.....	52
3.3.1 Апаратна складова в межах університету	52
3.3.2 Додаткове устаткування	53
3.3.3 Апаратна складова для користувача (студент/викладач).....	54
3.4 Збереження цілісності даних.....	56
3.5 Налаштування роботи системи	58
4 Тестування роботи лабораторії.....	60

Висновки	73
Перелік джерел посилання	75
ДОДАТОК А	77
ДОДАТОК Б	78
ДОДАТОК В	79
ДОДАТОК Г	80

ВСТУП

У світі, де темпи змін і технологій стрімко прискорюються, навчання є більш важливим і актуальним, ніж будь-коли раніше. Нещодавні зміни в освітньому секторі докорінно змінили парадигму навчання, зокрема дві ключові події: пандемія COVID-19 та початок війни в Україні у 2022 році. Ці виклики вимагають нових та інноваційних рішень у навчальному процесі.

Коли світ охопила пандемія COVID-19, багато навчальних закладів були змушені перейти на дистанційне навчання. Заборони та обмеження унеможливили традиційні очні лекції та практичні заняття в аудиторіях. Така ситуація спричинила серйозні труднощі для викладачів та студентів, вимагаючи нових та інноваційних підходів до навчання та комунікації [1].

Тим часом ситуація в навчальних закладах значно ускладнилася через початок війни в Україні у 2022 році. Ризики для студентів і викладачів, пов'язані з перебуванням в університетських кампусах під час конфлікту, зросли, що порушило важливі питання про те, як проводити і викладати заняття під час війни.

Спільним фактором, який відіграв важливу роль у вирішенні цих проблем, стала наявність гаджета, який є у майже кожного: мобільного телефону. Цей пристрій став мостом між студентами та викладачами, дозволяючи проводити лекції та практичні заняття в режимі онлайн навіть тоді, коли фізичний доступ до університету був обмежений.

У деяких регіонах ситуація повернулася до більш-менш нормального стану, і університети відновили очне навчання, але це залишається ризикованим варіантом для всіх, доки триває війна. Проте освіта - це скарб, який ніколи не втратить своєї цінності. Саме тому було вирішено створити віртуальну комп'ютерну лабораторію на кафедрі комп'ютерних систем та мереж, де кожен студент може отримати не лише теоретичні знання, а й практичні навички програмування.

Віртуальні лабораторії стають відповіддю на сучасні виклики у навчанні та надають студентам можливість отримати якісну освіту в умовах обмежень та кризи. Віртуальні лабораторії дозволяють студентам розвивати навички та вдосконалювати свої знання, не виходячи з дому або з безпечного місця. Це інноваційні рішення, які підтримують навчальний процес і сприяють підвищенню якості освіти.

Таким чином, віртуальні комп'ютерні лабораторії є необхідним елементом освітнього простору, який може гарантувати навчання та розвиток учнів навіть у найскладніших ситуаціях. Вони є відповіддю на виклики, що стоять перед сучасним освітнім середовищем, і готові надати учням знання та навички, необхідні для досягнення успіху в сучасному світі. Відповідно, обрана тема магістерської роботи є актуальною.

Мета роботи - допомогти студентам продовжувати навчання та отримувати практичні навички у будь-якій локації: за кордоном, чи у бомбосховищі, чи на тимчасово окупованій території України шляхом розробки віртуальної лабораторії кафедри комп'ютерних систем та мереж.

Для досягнення поставленої мети необхідно виконати наступні завдання:

- проаналізувати засоби створення віртуальних лабораторій;
- визначити вимоги до розроблюваної віртуальної лабораторії з урахуванням наявних вже апаратних та програмних засобів на кафедрі комп'ютерних систем та мереж НУ «Запорізька політехніка»;
- розробити комплекс завдань, узгоджений з напрямом освітніх програм, що викладаються на кафедрі;
- спроектувати віртуальну лабораторію та налаштувати її роботу;
- проаналізувати отримані результати.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Визначення мети і завдань віртуальної лабораторії

У сучасному суспільстві навчання та освіта завжди відігравали важливу роль у житті людей. Однак останні події в нашій країні принесли серйозні виклики освітньому процесу. Обмеження та виклики, пов'язані з епідеміями та військовими діями, змушують нас переосмислити підхід до навчання та запропонувати нові технологічні рішення. Одним із таких рішень може стати створення віртуальної лабораторії. Тож роздивимося більш детально, які цілі та завдання стають її перед створенням.

Цілі віртуальної лабораторії.

- надати студентам можливість активно розвивати свої навички програмування шляхом відпрацювання практичних навичок;
- надати студентам доступ до реальних завдань і проблем програмування для навчання та розвитку;
- надати студентам інтерактивне та візуальне середовище, де вони можуть бачити, як працюють їхні програми в реальному часі;
- мотивувати студентів до навчання за обраним комп'ютерним освітнім напрямом, надаючи їм можливість бачити практичні результати своєї роботи;
- надати студентам можливість користуватися спеціалізованим комп'ютерним обладнанням, навіть якщо вони не можуть відвідувати навчальний заклад.

Завдання віртуальної лабораторії:

- створити середовище віртуальної лабораторії, де студенти можуть розв'язувати реальні задачі з комп'ютерного напрямку, моделювати різні сценарії та тестувати свій код;
- розробити інтерактивне візуальне середовище, де студенти можуть вводити, змінювати та тестувати код безпосередньо з власних пристроїв віддалено;

- забезпечити можливість відстежувати прогрес студентів, оцінювати роботу та надавати зворотній зв'язок викладачам;
- дати можливість студентам співпрацювати один з одним, обмінюватися ідеями та вирішувати проблеми в команді;
- забезпечити безпеку даних та конфіденційність інформації, що використовується у віртуальній лабораторії;
- розвивати масштабованість та розширювати лабораторію для задоволення зростаючого попиту на освіту з інформаційних технологій;
- надавати підтримку та навчати викладачів використовувати віртуальної лабораторії для ефективного навчання за ІТ-напрямом.

1.2 Аналіз існуючих технологічних рішень

Для вивчення доступних рішень треба розуміти, що віддалена освіта та робота стають все більш поширеними, а тому важливо розглянути та вибрати належні технологічні рішення для створення віртуальної лабораторії на кафедрі комп'ютерних систем та мереж. Цей розділ присвячено розгляду різних технологій та платформ, які дозволять здійснювати віддалений доступ до комп'ютерів, розташованих в аудиторіях університету, та визначити найкращий вибір для створення віртуальної лабораторії. Розділ містить аналіз існуючих програмних продуктів, відкритих джерел, апаратних рішень і систем віртуалізації.

Було проаналізовано наступні найпопулярніші платформи для віддаленого доступу, а саме: TeamViewer, віддалений робочий стіл Chrome, Anydesk.

1.2.1 Платформа віддаленого доступу TeamViewer

TeamViewer - це комплексне і сучасне програмне рішення, яке забезпечує віддалений доступ, підтримку і спільну роботу в режимі реального часу між користувачами на різних пристроях і операційних системах (рисунок 1.1). Інструмент зарекомендував себе як лідер на ринку додатків для віддаленого

управління і використовується мільйонами людей по всьому світу для вирішення технічних проблем, надання підтримки та ефективної роботи незалежно від відстані.

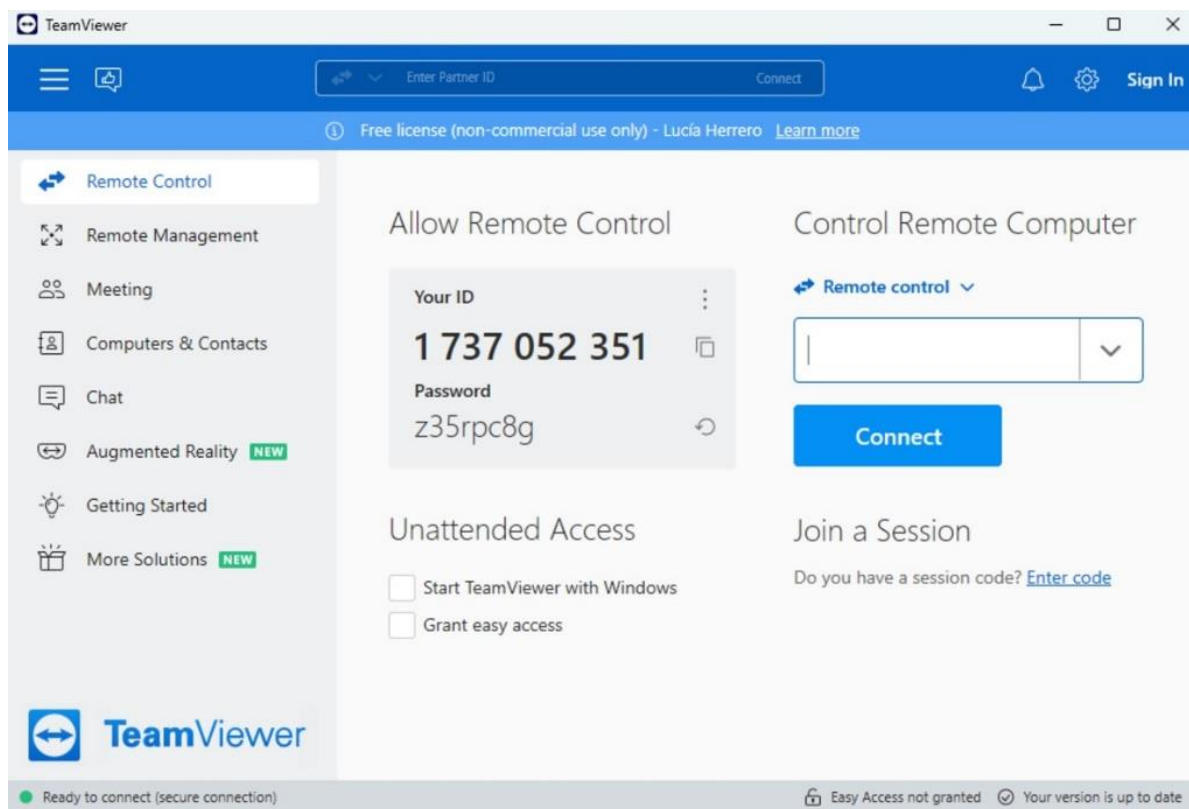


Рисунок 1.1 – Візуалізація програми TeamViewer

TeamViewer забезпечує безпечне і зашифроване з'єднання між комп'ютерами та іншими пристроями, такими як смартфони і планшети. Користувачі можуть керувати своїми пристроями на відстані так, ніби вони знаходяться перед ними, і виконувати такі завдання, як налаштування програмного забезпечення, діагностика проблем, передача файлів або спільна робота над проєктами. Крім того, TeamViewer пропонує функції чату, відеоконференцій і голосових дзвінків, що полегшує спілкування і спільну роботу на відстані.

Однією з унікальних особливостей TeamViewer є підтримка багатьох операційних систем і пристроїв: підтримуються Windows, macOS, Linux, Android і iOS, що дозволяє встановлювати з'єднання між пристроями на різних платформах і забезпечує оптимальну продуктивність. TeamViewer також доступний для

Windows, macOS, Linux, Android та iOS. Крім того, TeamViewer також доступний у портативній версії, яка не потребує встановлення, що полегшує його використання в надзвичайних ситуаціях або на зовнішніх пристроях.

Безпека є важливим аспектом TeamViewer і гарантує захист даних користувача та конфіденційність. З цією метою програма використовує наскрізне шифрування і двофакторну автентифікацію, щоб запобігти несанкціонованому доступу і гарантувати цілісність з'єднання. Нарешті, TeamViewer пропонує кілька варіантів кастомізації, щоб користувачі могли пристосувати функціональність до своїх потреб та вподобань [2].

1.2.2 Віддалений робочий стіл Chrome

Chrome Remote Desktop надає доступ до пристроїв на основі передових веб-технологій (рисунок 1.2). Можна підключатися до робочого або персонального комп'ютера на ходу, переглядати файли, що зберігаються на комп'ютері користувача, і віддалено ділитися екраном з друзями та колегами.

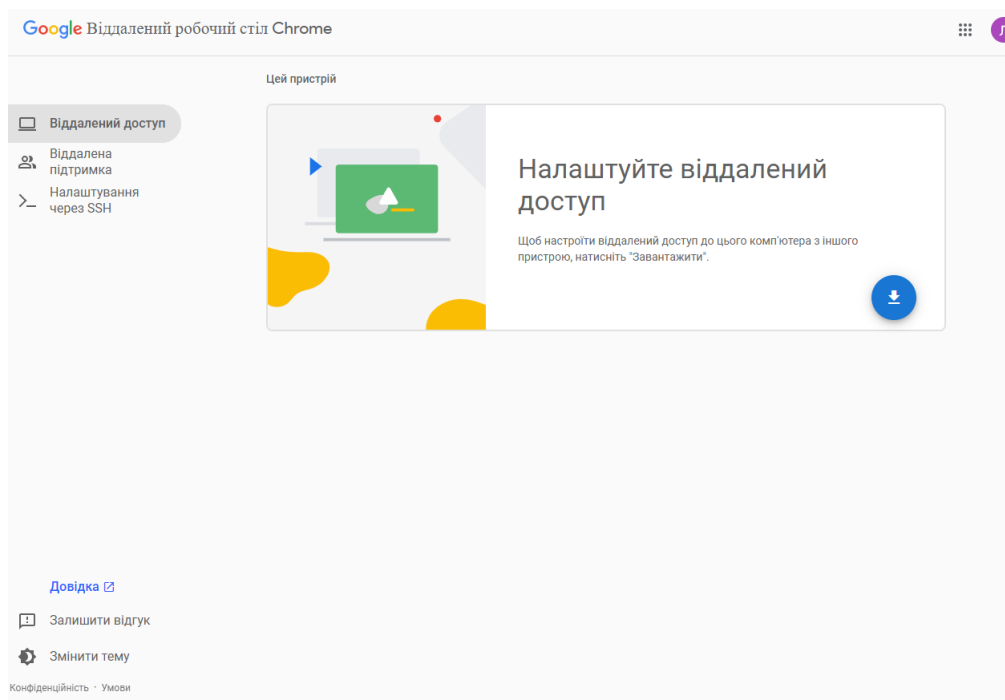


Рисунок 1.2 – Візуалізація віддаленого робочого столу Chrome

Віддалений доступ до комп'ютерів дозволяє користувачам запускати програми та переглядати файли в будь-який час, де б вони не знаходилися.

З'єднання використовує новітні веб-технології з відкритим вихідним кодом, такі як WebRTC, а безпека забезпечується інфраструктурою Google.

Підключення може бути здійснене з мобільних телефонів і планшетів на базі Android та iOS, а також комп'ютерів. У будь-якому випадку, до пристрою та файлів, що зберігаються на ньому, можна легко отримати віддалений доступ [3].

1.2.3 Платформа віддаленого доступу AnyDesk

AnyDesk - це потужне програмне рішення, призначене для надання альтернативних засобів віддаленого доступу до комп'ютерів та маніпулювання різними файлами і документами (рисунок 1.3).

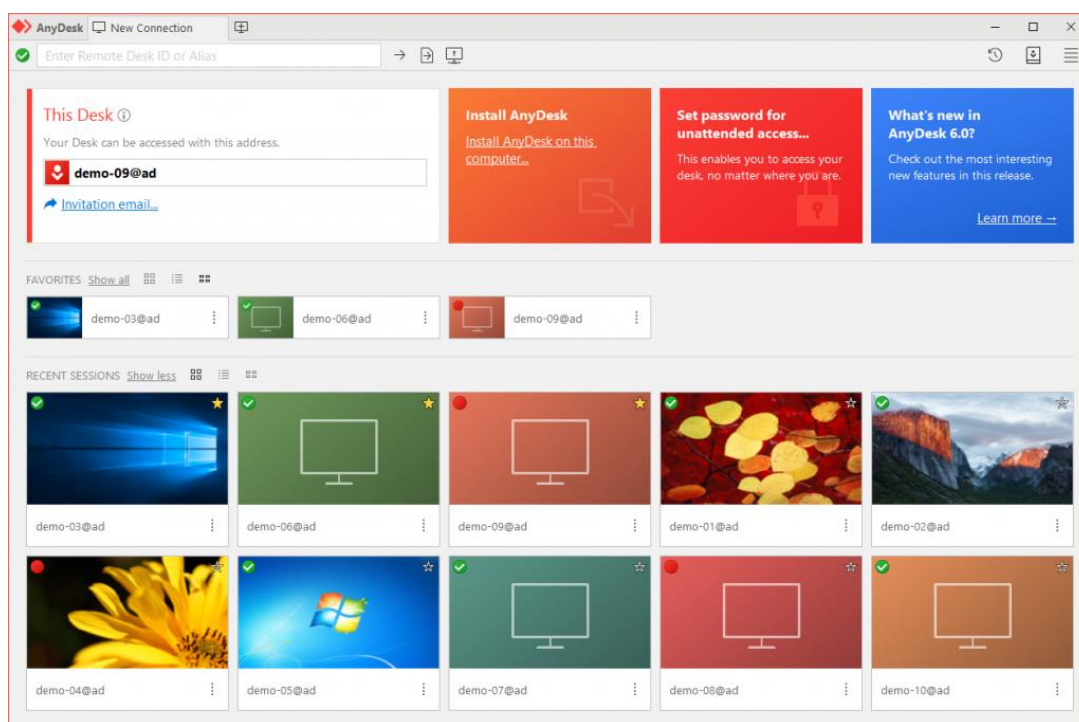


Рисунок 1.3 – Візуалізація програми AnyDesk

Зручна та ефективна програма дозволяє віддалено керувати одним або декількома комп'ютерами з мінімальними зусиллями, незалежно від їхнього розташування.

Портативний інсталятор дозволяє працювати без прав адміністратора, а AnyDesk є безкоштовним для особистого користування.

Утиліта має дуже практичний і зрозумілий інтерфейс, що робить її доступною для користувачів з будь-яким рівнем підготовки.

У головному вікні просто необхідно ввести відповідну адресу в поле «Інші робочі станції», щоб підключитися до інших комп'ютерів, на яких встановлені копії AnyDesk.

Можна швидко підключатися до одного або декількох віддалених комп'ютерів.

AnyDesk створений для того, щоб користувачі могли легко отримати доступ до потрібних документів у будь-який час і в будь-якому місці, з домашнього комп'ютера на робочий і навпаки. З міркувань безпеки AnyDesk не містить функцій обміну файлами або їх передачі.

Однак утиліта дозволяє обмінюватися вмістом буфера обміну між двома комп'ютерами, що полегшує копіювання і вставку інформації. Аналогічно, зображення можна захопити з екранів підключених комп'ютерів і зберегти на робочому столі у форматі PNG.

AnyDesk підтримує неконтрольований доступ, який дозволяє підключатися до цільового комп'ютера, не вимагаючи підтвердження на іншому кінці. У цьому випадку встановлюється пароль, щоб запобігти несанкціонованому підключенню.

Для кожного з'єднання користувач може ввімкнути або вимкнути набір дозволів, таких як прослуховування вихідного аудіо, керування та блокування клавіатури і миші, а також доступ до буфера обміну. Також можна встановити різні параметри відображення.

Завдяки TLS 1.2 AnyDesk пропонує ту саму технологію безпеки, що використовується в онлайн-банкінгу. Зашифрована автентифікація відбувається між обома сторонами з'єднання; ідентифікатор AnyDesk ID захищений від підробки. Сам виконуваний файл захищений від несанкціонованого доступу.

Загалом, AnyDesk є зручним і надійним інструментом для успішної організації віддалених підключень до інших комп'ютерів, а також для доступу до цільових комп'ютерів і контролю над ними з мінімальними зусиллями [4].

1.2.4 Результати аналізу програмного забезпечення

У кожної програми є свої переваги та недоліки, які наведені у таблиці 1.1, тому більш детально були розглянуті ці аспекти перед вибором пріоритету програм.

Таблиця 1.1 – Переваги та недоліки програм

Назва	Переваги	Недоліки
TeamViewer	-висока надійність та стабільність; -зручний інтерфейс користувача; -підтримка для різних операційних систем.	-потреба у платі за деякі функції; -обмеження на кількість одночасних підключень у безкоштовній версії.
Віддалений робочий стіл Chrome	-безкоштовний та простий у використанні; -не потребує встановлення додаткового програмного забезпечення.	-може мати обмежену функціональність.
AnyDesk	-висока швидкість і низька латентність. -підтримка для різних операційних систем.	-обмеження у безкоштовній версії.

Після ретельного аналізу найпопулярніших платформ для віддаленого доступу, таких як TeamViewer, віддалений робочий стіл Chrome і Anydesk, можна зробити висновок, що кожна з цих платформ має свої переваги та недоліки. Важко виокремити одну конкретну платформу як єдиний ідеальний вибір, оскільки кожна програма має свій власний набір особливостей та переваг, а також різний рівень відомостей та комфорту користувачів з ними.

Вибір платформи для віртуальної лабораторії повинен залежати від індивідуальних потреб та знайомства кожного користувача з цими програмами. Важливо враховувати, що кожна програма може бути налаштована під конкретні потреби користувача, і немає обмежень у виборі оптимальної платформи для конкретного завдання.

У цьому проєкті, для створення віртуальної лабораторії було обрано TeamViewer. Вибір здійснено з урахуванням конкретних потреб та зручності користувачів, які вже мали певний досвід роботи з цією програмою. Такий вибір дозволяє ефективно впровадити віртуальну лабораторію та спростити процес вивчення програмування для студентів та викладачів.

1.3 Аналіз вимог до віртуальної лабораторії

1.3.1 Урахування потреб користувачів

При виборі платформи для побудови віртуальної лабораторії на кафедрі комп'ютерних систем та мереж дуже важливо враховувати потреби та очікування користувачів цієї лабораторії. Особливу увагу слід звернути на наступні аспекти:

- важливо мати інтуїтивний та доступний інтерфейс для користувачів усіх рівнів підготовки. Інтерфейс повинен бути зрозумілим та дружнім до користувача, щоб навіть початківці могли з легкістю використовувати віртуальну лабораторію;
- платформа повинна надавати зручні інструменти та функції для вивчення дисциплін напряму 123 Комп'ютерна інженерія та практичного застосування опанованих знань. Вона повинна сприяти активному навчанню і допомагати користувачам розвивати свої навички;
- оскільки користувачі можуть використовувати різні операційні системи та пристрої (комп'ютери, планшети, смартфони), платформа повинна бути сумісною з різними платформами та браузерами;
- віртуальна лабораторія повинна працювати ефективно та швидко, надавати користувачам миттєвий доступ до необхідних ресурсів та інструментів;
- платформа повинна надавати викладачам можливість створювати та керувати навчальними матеріалами, оцінювати роботу студентів та надавати зворотний зв'язок;

– забезпечення конфіденційності та безпеки даних користувачів є критично важливим аспектом. Платформа повинна забезпечувати високий рівень захисту особистих даних та інформації студентів.

1.3.2 Вимоги доступу до віртуальної лабораторії

Для забезпечення доступу до віртуальної лабораторії важливо враховувати сучасне становище країни в умовах війни. На жаль, не всі студенти мають можливість використовувати комп'ютери для лабораторних робіт. В більшості випадків, вони покладаються на підручні гаджети, такі як ноутбуки, планшети, телефони. Оскільки студенти використовують різні пристрої та платформи для доступу до віртуальної лабораторії, важливо, щоб система була зручною, інтуїтивною та підтримувала всі ці пристрої.

Крім того, слід враховувати, що є студенти, які знаходяться на окупованих територіях. Для них особливо важливе, щоб підключення до віртуальної лабораторії було захищеним, оскільки їхні дані можуть бути в небезпеці. Також існує проблема з доступом до інтернету на окупованих територіях, тому важливо, щоб передача трафіку була максимально ефективною і не заважала студентам навчатися.

Отже, для забезпечення доступу до віртуальної лабораторії з різних пристроїв і платформ в умовах війни, необхідно розглянути можливість використання різних гаджетів, забезпечити захист даних для студентів на окупованих територіях та знайти рішення для проблем з інтернетом на цих територіях.

2 РОЗРОБКА ЗАВДАНЬ ДЛЯ ВИКОНАННЯ У ВІРТУАЛЬНІЙ ЛАБОРАТОРІЇ

Віртуальну лабораторію доцільно використовувати з точки зору двох типів завдань:

- завдання, які виконуються з використанням програмного забезпечення, яке вимагає потужних апаратних комп'ютерних ресурсів, які не завжди є у студентів;
- завдання, які виконуються із застосуванням специфічного обладнання.

В рамках цієї роботи розробимо завдання кожного типу для користувачів віртуальної лабораторії кафедри комп'ютерних систем та мереж.

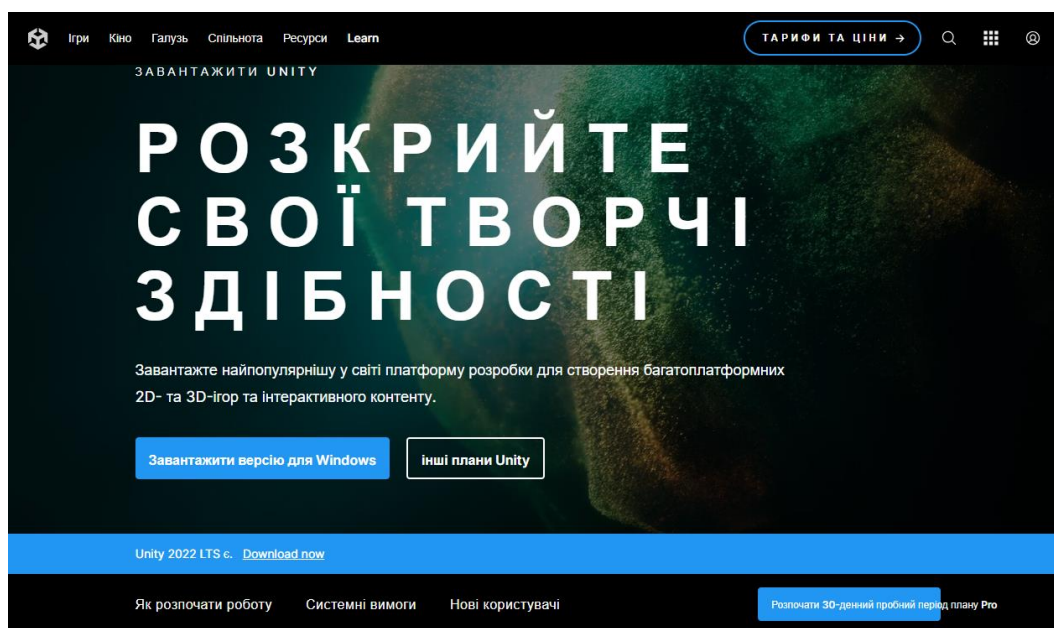
2.1 Завдання у програмному середовищі віртуальної лабораторії

Є дуже багато різноманітних практичних робіт для студентів і кожна робота вимагає встановленню нового програмного забезпечення, котрі мають певні вимоги від машини користувача, і не в кожного студента є ресурс у ПК до того чи іншого ПЗ. Це може бути як програмування мобільних додатків, так і створення доповненої реальності. Доповнена реальність, або якщо коротко AR (Augmented Reality) – це технологія, що додає в реальний, фізичний світ цифрові об'єкти.

За допомогою технології AR віртуальні об'єкти можна інтегрувати в матеріальний світ: камера знімає реальний світ і доповнює його віртуальними об'єктами. Такі додатки працюють із тривимірними моделями, текстами, зображеннями, відео та анімацією [5].

Робота з доповненою реальністю досить цікава для студентів, тому що студент може самотужки долучитись до створення цієї реальності, отримати базові навички в цьому напрямку та у майбутньому реалізувати себе у цій сфері, хоча вона не нова але дуже багато не зайнятих шпаринок, де кмітливість і креативність допоможе у подальшому розвитку.

Найбільш популярним програмним середовищем для створення доповненої реальності є Unity Hub. Необхідно завантажити програмне забезпечення з офіційного сайту (рисунок 2.1) та встановити його на ПК або використовувати віртуальну лабораторію [6].



Три етапи розробки на Unity

Рисунок 2.1 – Офіційний для завантаження Unity Hub

Після встановлення програмного забезпечення необхідно увійти до облікового запису та активувати ліцензію на використання, як представлено на рисунок 2.2. Після успішної активації необхідно перейти до розділу «installs» та обрати властивості установки (рисунок 2.3), а потім підтвердити умови використання ПЗ.

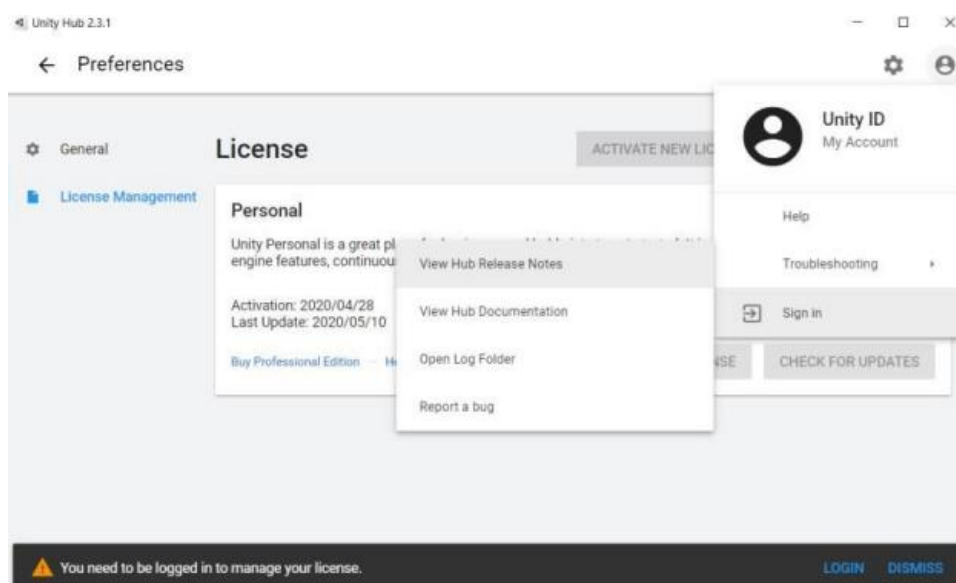


Рисунок 2.2 – Вхід до облікового запису



Рисунок 2.3 – Вибір властивостей установки

Також до Unity необхідно завантажити додаткові пакети, у даному випадку Vuforia.

Vuforia - це платформа доповненої реальності та інструментарій для розробки програмного забезпечення для мобільних пристроїв, розроблений Qualcomm. Vuforia використовує технологію комп'ютерного зору для відстеження плоских зображень і простих об'єктів тривимірної реальності в режимі реального часу.

Ця можливість рендерингу зображень дозволяє розробникам позиціонувати та переорієнтовувати віртуальні об'єкти, такі як 3D-моделі та медіа-контент, що відображаються на мобільних пристроях, у поєднанні з реальною сценою. Віртуальні об'єкти розміщуються в реальному зображенні так, щоб свідомість глядача ставилася до них так само, створюючи відчуття, що віртуальні об'єкти є частиною реального світу.

Vuforia підтримує різні типи мішеней, включаючи мішені-зображення без маркерів, тривимірні багатоцільові мішені та опорні маркери, які виділяють об'єкти в сцені для розпізнавання. Додаткові можливості включають використання так

званих "віртуальних кнопок" для уникнення ефекту маскування (оклюзії) об'єктів, виділення об'єктів і можливість їх програмного створення та реконфігурації в межах коду, що самоналаштовується [7].

Після запуску Unity необхідно відкрити раніше завантажені пакети UnityUIExtensions-2-0 і add-vuforia-package-9-0-12. Після відкриття кожного пакета необхідно натиснути кнопку «Import» [8].

Далі для повноцінного використання Vuforia необхідно отримати ліцензійний ключ на офіційному сайті, увійшовши до облікового запису, і потім отриманий ключ вставити в Unity. Після чого відкривається простір Unity, рисунок 2.4.

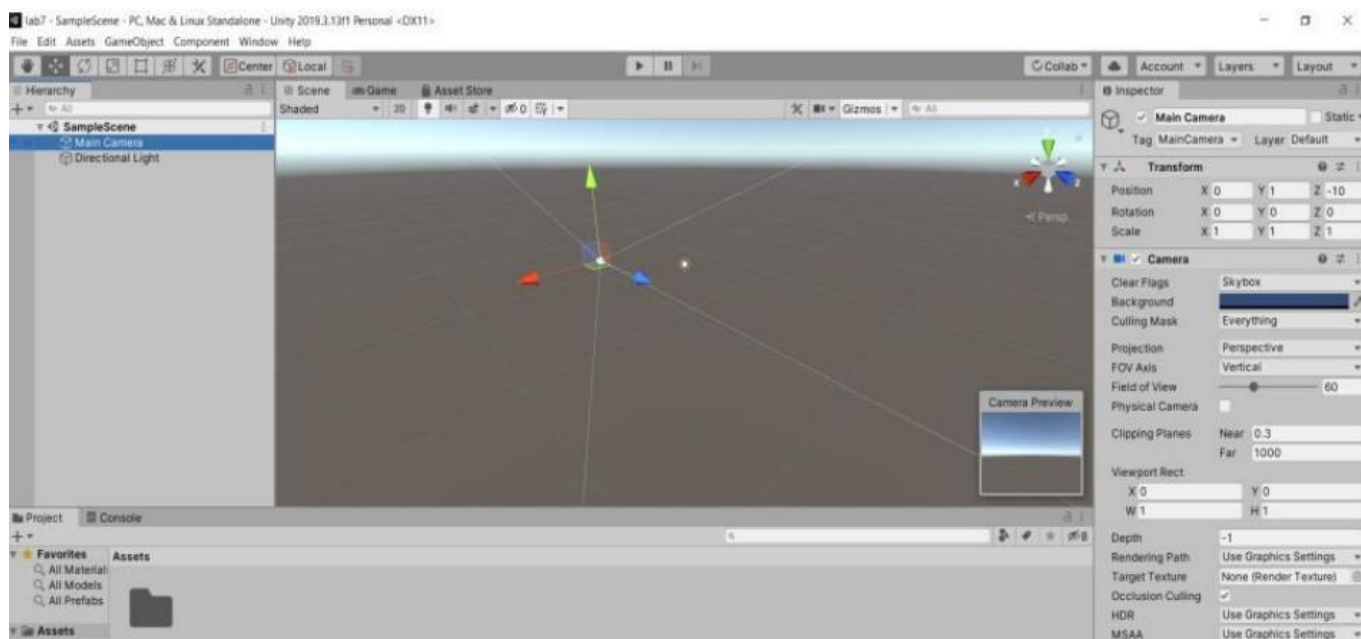


Рисунок 2.4 – Робочий простір Unity

Наступним кроком необхідно видалити головну камеру та додати AR-camera, це зображено на рисунку 2.5.

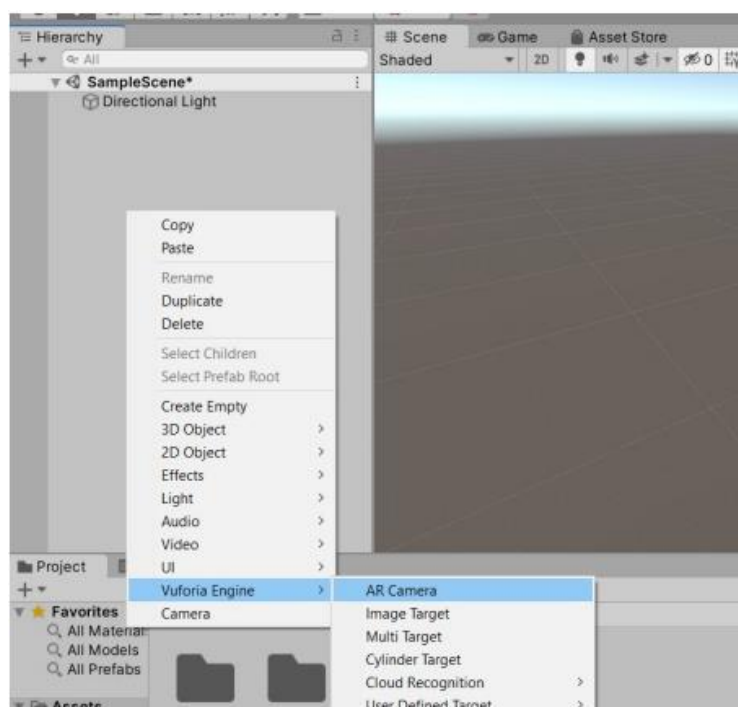


Рисунок 2.5 – Додавання AR-camera

Далі переходимо в налаштування проєкту, налаштовуємо параметри у Player, натискаємо Add open scenes, натискаємо Switch platform та обираємо платформу Android і на фіналі треба натиснути Build (рисунок 2.6).

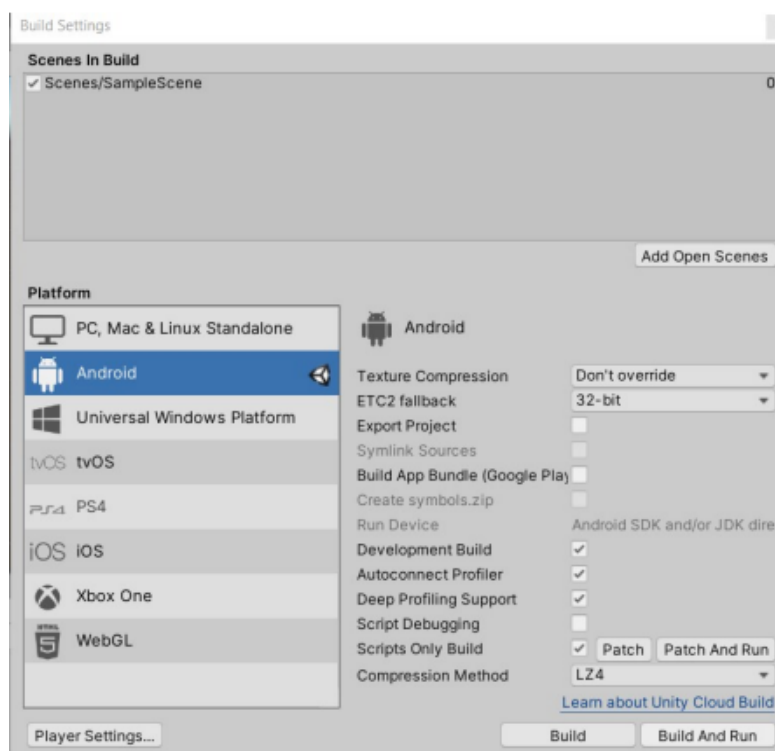


Рисунок 2.6 – Процес налаштування необхідних налаштування

Провівши всі налаштування починається практична частина використання Unity3d. Для початку необхідно створити маркер. Для цього потрібно створити малюнок незвичайний з чіткими межами чорно-білий і зберегти його з будь-яким іменем. Щоб використовувати цей маркер, його необхідно завантажити на сайт <https://developer.vuforia.com/vui/develop/databases> та натиснути Add Database (рисунок 2.7, 2.8).

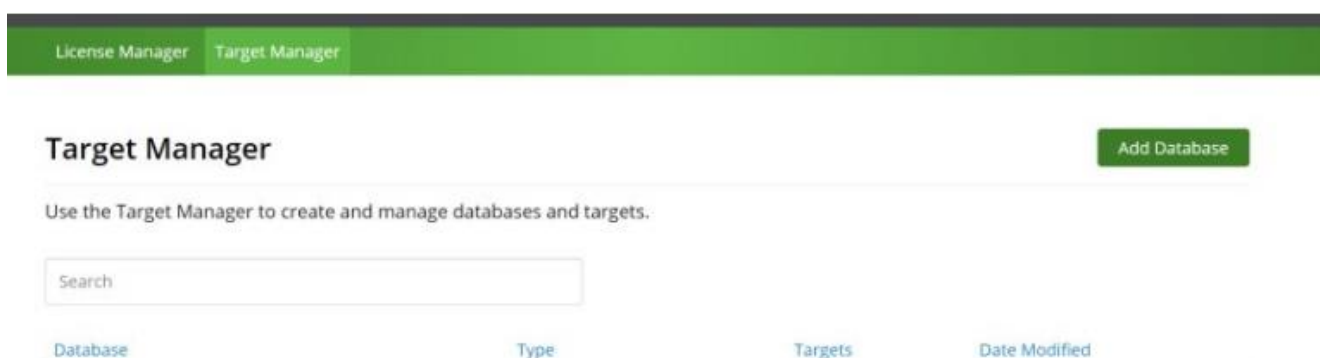
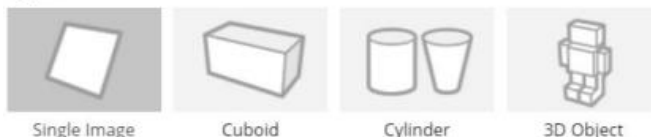


Рисунок 2.7 – Створення об'єкту для завантаження маркера

Add Target

Type:



File:

1111.png

.jpg or .png (max file 2mb)

Width:

1

Enter the width of your target in scene units. The size of the target should be on the same scale as your augmented virtual content. Vuforia uses meters as the default unit scale. The target's height will be calculated when you upload your image.

Name:

1111

Name must be unique to a database. When a target is detected in your application, this will be reported in the API.

Рисунок 2.8 – Завантаження зображення

Потім необхідно зайти до Unity Hub та імпортувати пакет Vuforia. Далі потрібно імпортувати необхідну базу маркерів, рисунок 2.9. І наступним кроком треба додати зображення до проєкту Vuforia Engine – Image target, рисунок 2.10.

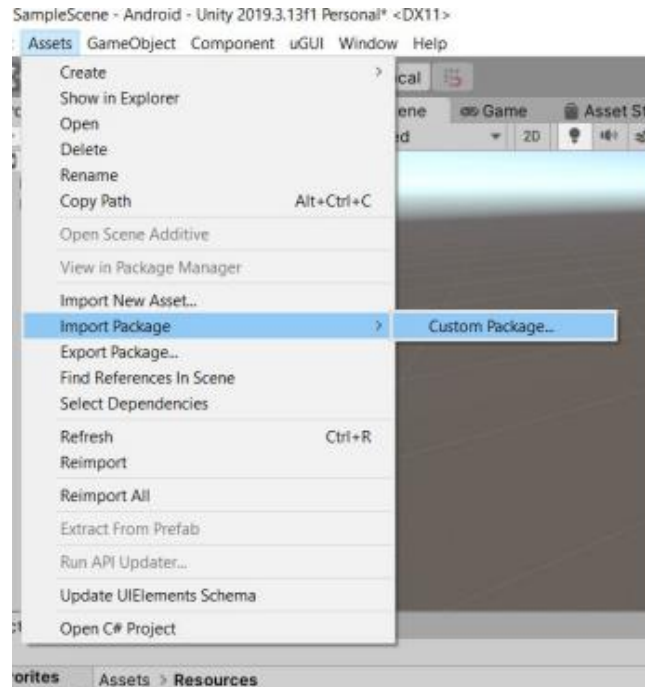


Рисунок 2.9 – Імпорт бази маркерів

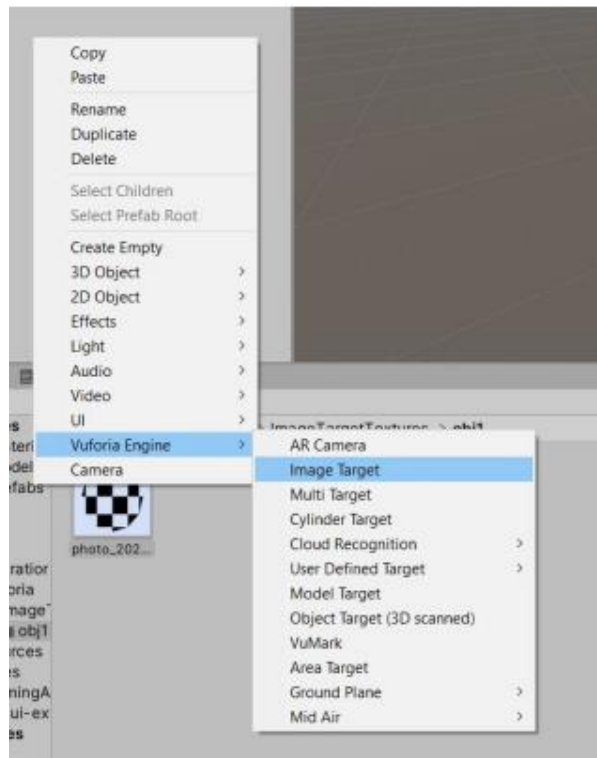


Рисунок 2.10 – Додавання об'єкту

На рисунку 2.11 зображений 3D-об'єкт у робочому просторі Unity Hub.

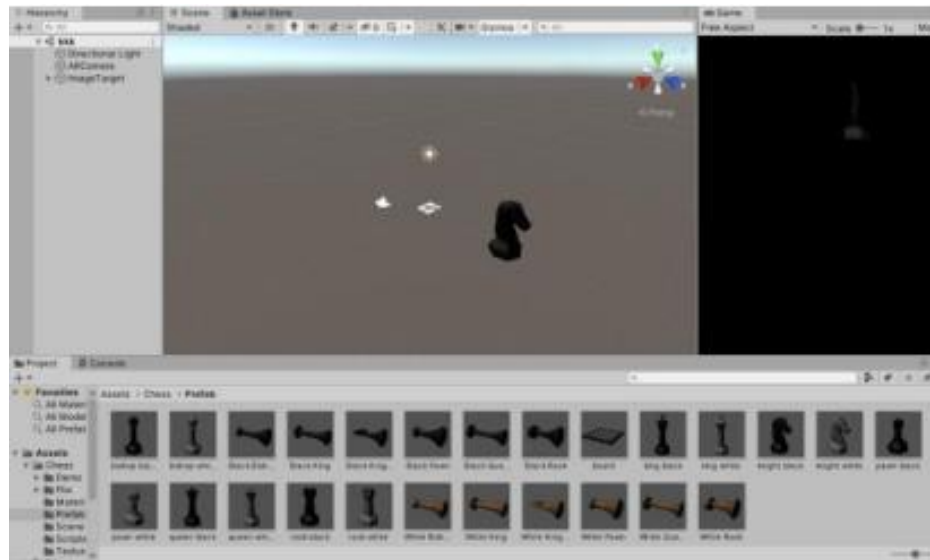


Рисунок 2.11 – 3D-об'єкт

Необхідно запустити проєкт, увімкнути камеру і на телефоні (або роздруковану) відкрити те зображення, що завантажили, рис 2.12.

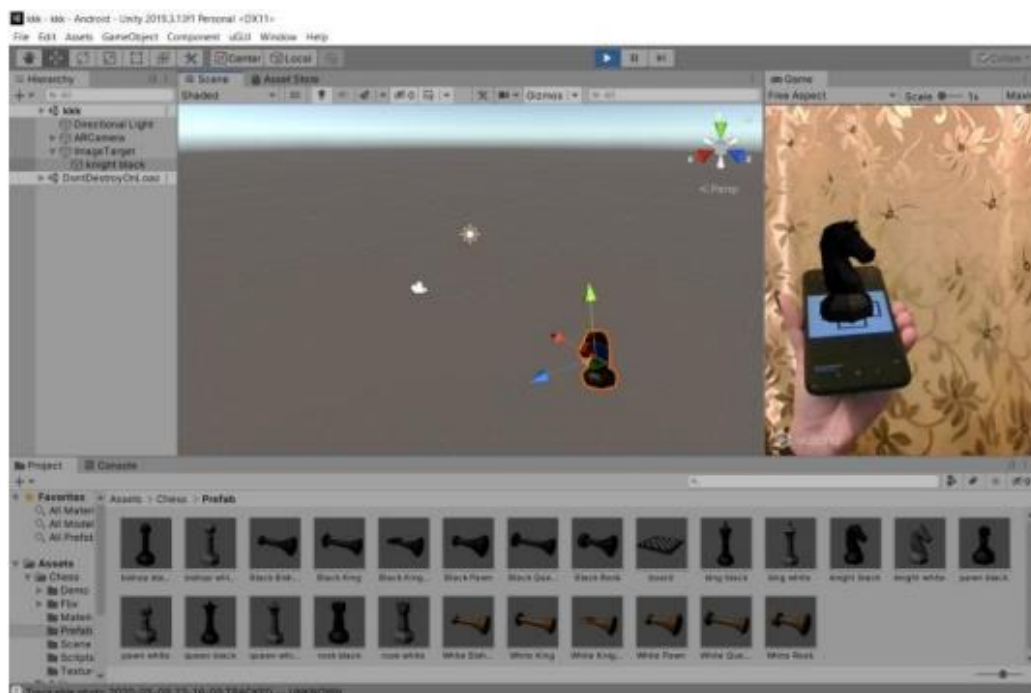


Рисунок 2.12 – Результат доповненої реальності

2.2 Використання специфічного обладнання віртуальної лабораторії

Одним із важливих аспектів, коли розглядається можливість використання віртуальної лабораторії, є потреба у вирішенні завдань, що потребують спостереження і керування певним спеціалізованим обладнанням. Розглянемо це на прикладі робота-пилососа, який доступний в кабінеті № 52 навчальної лабораторії «Системної інтеграції комп'ютерних технологій» і може використовуватися у межах завдань для дисциплін «Кіберфізичні системи» та «Інтегровані комп'ютерні системи» спеціальності 123 Комп'ютерна інженерія.

Запуск робота-пилососа, у даному випадку моделі iRobot Create 2 Roomba 600, вимагає розробки специфічних завдань для виконання студентами. Основними кроками є:

- визначення завдань;
- апаратурна підготовка;
- завдання для студентів;
- спостереження і аналіз;
- оцінка та результати.

Перш за все, потрібно розробити завдання, які студенти будуть виконувати з роботом-пилососом. Ці завдання повинні включати в себе конкретні вимоги та цілі, які студенти повинні досягти.

Важливо підготувати апаратуру та робота-пилососа для виконання завдань. Це включає в себе перевірку та підготовку робота до роботи, а також налагодження зв'язку з віртуальною лабораторією.

Студентам необхідно надати інструкції та завдання, які дозволять їм керувати роботом-пилососом через віртуальний інтерфейс. Це може включати в себе програмування рухів робота, збір даних, аналіз результатів тощо [9].

Під час виконання завдань, важливо мати можливість спостерігати за роботою робота-пилососа та аналізувати його робочі параметри та результати в

реальному часі. Для цього може використовуватися відеоспостереження та засоби збору даних.

На завершальному етапі, студенти повинні представити результати своєї роботи та оцінити виконання завдань. Це може включати в себе аналіз даних, створення звітів та обговорення результатів з викладачем.

Отже, розглянемо більш детально необхідні програмні засоби та апаратну складову для реалізації завдання на базі робота-пилососа.

2.2.1 Апаратне забезпечення

Перед розглядом самого робота-пилососа і його застосування в рамках віртуальної лабораторії, розглянемо контекст, в якому відбувається цей проект.

Roomba - розумний і ефективний робот-пилосос. Усі роботи-пилососи Roomba (рисунок 2.13) оснащені унікальною розробкою iRobot - системою штучного інтелекту iAdapt. iAdapt використовує десятки датчиків для моніторингу оточення Roomba, надсилаючи до 67 імпульсів у секунду, яка контролює ефективну безпеку та розумне прибирання.



Рисунок 2.13 – Робот-пилосос iRobot Create 2 Roomba 600

Робот має наступні анатомічні характеристики які зображені на рисунка 2.14 і 2.15, де всі розміри вказані у міліметрах. Також вага робота становить приблизно 3,5 кілограми.

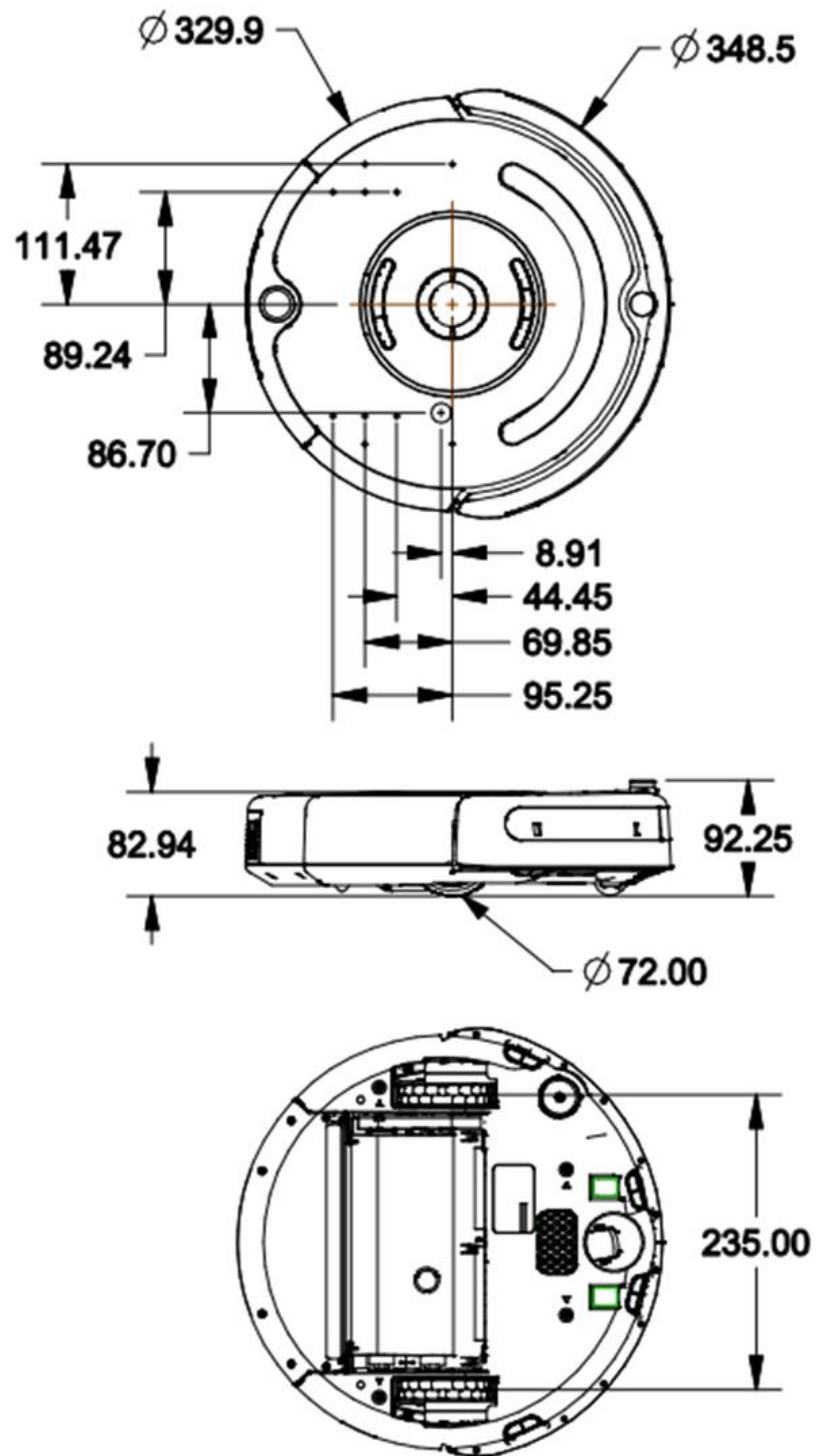


Рисунок 2.14 – Анатомічні характеристики iRobot Create 2

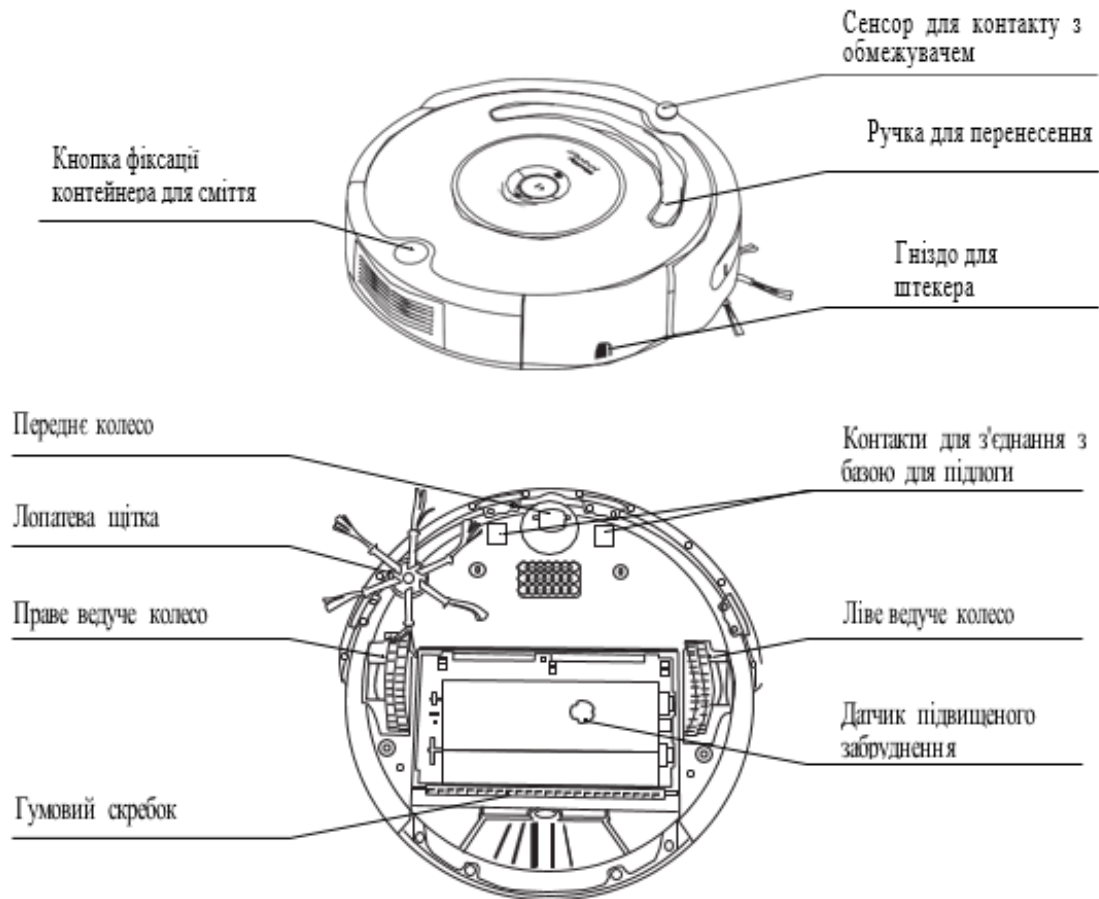


Рисунок 2.15 – Схеми пилососа з поясненнями

На рисунку 2.16 зображено панель приладів, тобто кнопки та індикатори пилососа.

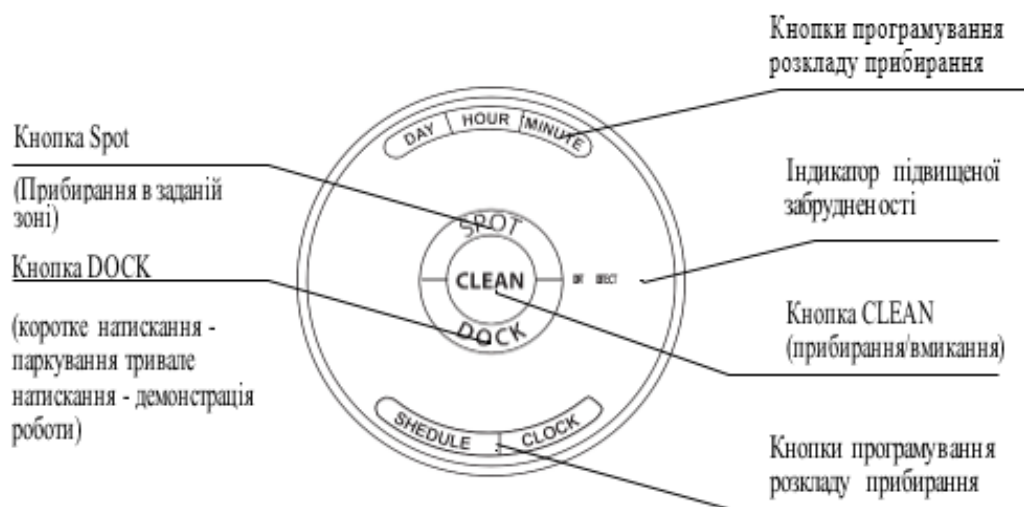


Рисунок 2.16 – Кнопки та індикатори

Увімкнення та вимкнення Roomba має такі особливості:

- робот Roomba не має окремого вимикача;
- щоб увімкнути Roomba, треба натиснути кнопку CLEAN. Пролунає сигнал, і кнопка CLEAN засвітиться;
- щоб розпочати роботу, знову треба натиснути кнопку CLEAN;
- щоб зупинити роботу робота, потрібно натиснути кнопку CLEAN під час збирання;
- якщо потрібно продовжити роботу, треба натиснути кнопку CLEAN ще раз;
- щоб вимкнути Roomba, треба натиснути кнопку CLEAN і утримувати її, доки світлові індикатори робота не згаснуть.

Розглянемо час прибирання на повній зарядці, у даному випадку, час роботи. Повний цикл роботи на повній зарядці складає близько 2-х годин. Перед кожним прибиранням акумулятор Roomba має бути повністю зарядженим. Повністю заряджений акумулятор Roomba забезпечить повний цикл прибирання. Тобто, враховуючи те, що пара йде 1 година 20 хвилин, повної зарядки буде вистачати для проведення практичних занять. А так як, після виконання всіх завдань, робот-пилосос повертається на зарядну базу – заряду вистачить на повний робочий день експлуатації.

Тривалість штатної зарядки триває 2-2,5 години. По закінченні заряджання кнопка Clean загориться зеленим. Це означає, що Roomba знову готовий до роботи.

Енергозбереження робота-пилососа. У пасивному режимі Roomba переходить у режим сну після 5 хвилин бездіяльності, щоб зберегти заряд акумулятора. У безпечному та повному режимах Roomba ніколи не спатиме, і якщо його залишити в такому стані протягом тривалого періоду часу, він глибоко розрядить акумулятор, навіть якщо його підключити до зарядного пристрою. Зарядний пристрій живить Roomba в усіх режимах, але не заряджатиме акумулятор у безпечному або повному режимах. Важливо, щоб після завершення роботи з Roomba потрібно ввести команду Passive або Stop, щоб захистити акумулятор.

Коли закінчується цикл прибирання або заряд акумулятора, Roomba повертається на зарядну базу Home Base для підзарядки.

Щоб робот міг повернутись, він повинен знайти інфрачервоний сигнал бази. Тому зарядна база завжди повинна бути підключена до живлення.

Коли Roomba знаходиться на зарядній базі, на ній повинен горіти зелений індикатор, підтверджуючи успішний процес заряджання.

Зарядну базу потрібно розмістити на твердій (це може бути і жорсткий килим) і рівної поверхні так, щоб робот Roomba міг вільно дістатися до неї.

Зарядну базу необхідно розмістити тильною стороною до стіни або іншого стійкого нерухомого об'єкта, у даному випадку до внутрішньої стіни столу, який розглянуто у розділі 3.1. Якщо Roomba не потрапляє на зарядну базу з першого разу, він повторює спробу.

Щоб вручну відправити Roomba на зарядну базу, необхідно помістити робота на відстані 1-1,5 метра від зарядної бази та натиснути кнопку DOCK (парковка), це зображено на рисунку 2.17.

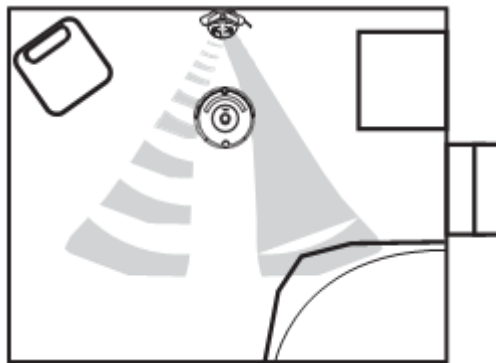


Рисунок 2.17 – Відправка Roomba на зарядну базу

У цьому проєкті робот-пилосос буде знаходитись на спеціально обладнаному столі з бортиками для унеможливлення пошкодження у разі падіння. Також можна буде з будь-якого місця відправити його на зарядну станцію.

Далі було розглянуто, що відкритий інтерфейс Roomba (OI) - це програмний інтерфейс для контролю та маніпулювання поведінкою Roomba. Інтерфейс програмного забезпечення дозволяє керувати поведінкою Roomba та зчитувати його датчики за допомогою серії команд, включаючи команди режиму, команди приводу, команди пісень, команди очищення та команди датчиків, які можна

надсилати на послідовний порт Roomba за допомогою ПК або мікроконтролера, який підключається до роз'єму Mini-DIN.

Коли мова йдеться про фізичні зв'язки, щоб використовувати ОІ, до зовнішнього роз'єму Mini-DIN на Roomba потрібно під'єднати процесор, здатний генерувати послідовні команди, наприклад ПК або мікроконтролер. Цей роз'єм забезпечує двосторонній послідовний зв'язок на рівнях TTL (0–5 В). Роз'єм також забезпечує нерегульоване пряме з'єднання з акумулятором Roomba, який можна використовувати для живлення програм ОІ. Роз'єм Mini-DIN розташований у задній правій частині Roomba, під знімною пластиковою кришкою.

Розпиновка роз'єму Mini-DIN зовнішнього послідовного порту Roomba.

На цій діаграмі (рисунок 2.18) показано розпиновку верхнього вигляду гнізда в Roomba. Слід зауважити на те, що штифти 6 і 7 знаходяться на зовнішньому оточуванні Roomba. Кожен контакт розпиновки описано у таблиці 2.1.

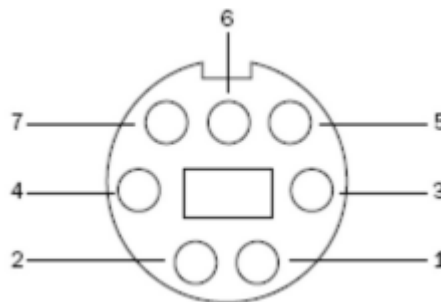


Рисунок 2.18 – Роз'єм Mini-DIN

Таблиця 2.1 – Таблиця розпіновки

Pin	Name	Description
1	Vpwr	Roomba battery + (unregulated)
2	Vpwr	Roomba battery + (unregulated)
3	RXD	0 – 5V Serial input to Roomba
4	TXD	0 – 5V Serial output from Roomba
5	BRC	Baud Rate Change
6	GND	Roomba battery ground
7	GND	Roomba battery ground

Оскільки контакти RXD і TXD використовують логічну напругу 0–5 В, а послідовні порти ПК використовують різні напруги (рівні RS 232), необхідно змінити рівні напруги. Знадобиться використовувати кабель зміни рівня, наприклад USB-кабель iRobot Create.

Контакти 1 і 2 (Vpwr) під'єднані до батареї Roomba через перезапобіжник 200 мА РТС. Безперервне споживання цих двох контактів не повинно перевищувати 200 мА. Не знімайте піковий струм більше 500 мА з цих контактів, інакше запобіжник перезапуститься.

Параметри послідовного порту мають наступні властивості:

- швидкість передачі: 115200 або 19200;
- біти даних: 8;
- парність: немає;
- стоп-біт: 1;
- контроль потоку: немає [10 - 12].

2.2.3 Програмне забезпечення

iRobot Create 2 Roomba 600 - є програмованим роботом, який може бути використаний для різних завдань і досліджень [13]. У робота є підтримка кількох мов для програмування та керування ним, зокрема:

- можна програмувати iRobot Create 2 за допомогою мови програмування C або C++;
- також робот підтримує Python, і для нього існують бібліотеки, які допомагають контролювати робота через Python;
- ROS (Robot Operating System): ROS - це платформа для робототехнічних досліджень, яку також можна використовувати для керування iRobot Create 2.

За допомогою цих мов програмування можна створювати власні програми для керування роботом, створювати маршрути, збирати дані з датчиків і багато іншого. Вибір мови залежить від навичок студентів у програмуванні та завдань, які необхідно виконати з iRobot Create 2.

Розглядаючи можливі мови програмування для робота-пилососа, можна відзначити наступне. Мови C/C++ та Python - це мови, що є популярними і досить поширеними серед розробників та вивчаються у межах освітніх програм кафедри комп'ютерних систем та мереж спеціальності 123 Комп'ютерна інженерія. Використання C/C++ та Python спрощує створення програмного коду для керування роботом і збору даних з його датчиків. Що стосується ROS (Robot Operating System), вона є потужною та розвинутою платформою для робототехнічних досліджень та розробки роботів.

Також ROS переслідує додаткові цілі, а саме:

- ROS повинен бути якомога тоншим - щоб код, написаний для ROS міг бути використаний і в інших програмних системах. Наслідком є те, що ROS легко інтегрується з іншими системами програмного забезпечення роботів: ROS вже інтегрований з OpenRAVE, Orocos і Player;

- кращою моделлю розвитку є написання бібліотек із чистим функціональним інтерфейсом;

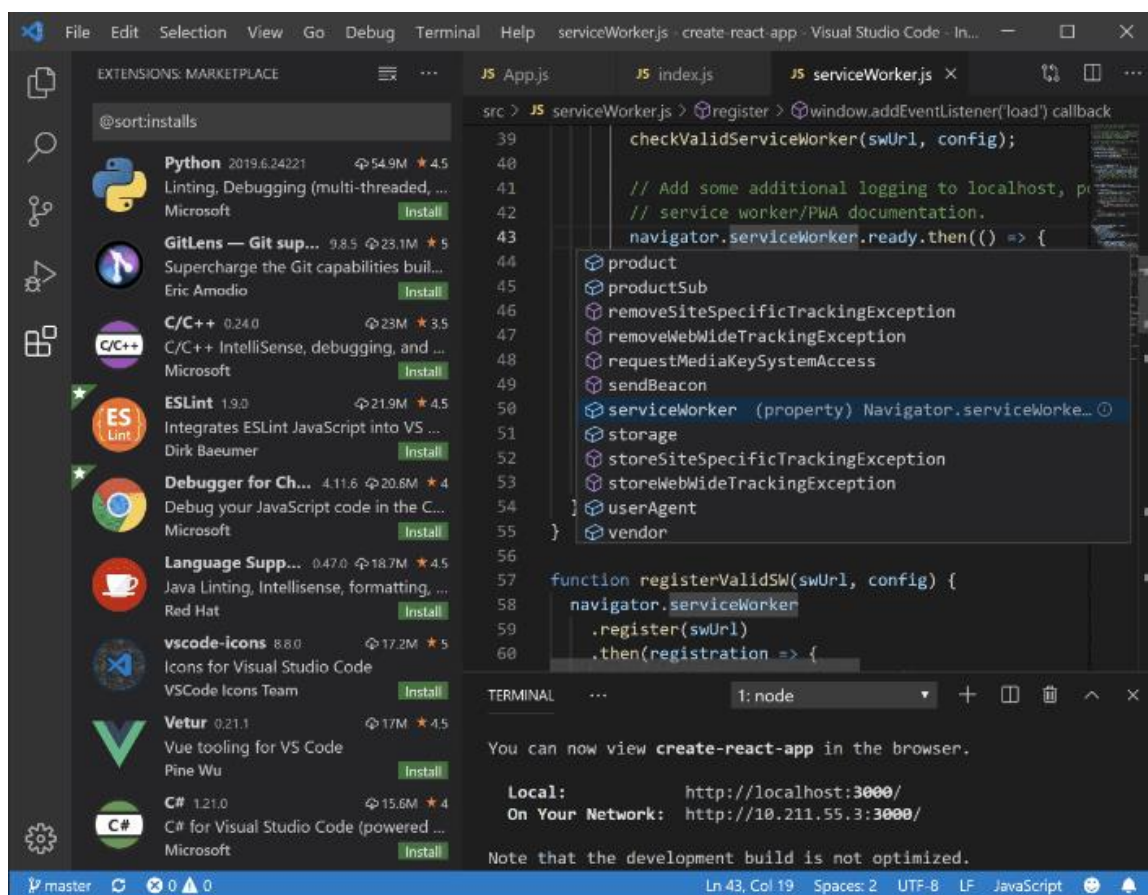
- структуру ROS легко реалізувати будь-якою сучасною мовою програмування. ROS вже реалізований на: Python, C++, Lisp та експериментальні бібліотеки на Java та Lua;

- ROS має вбудований фреймворк для тестування – rostest, що дозволяє легко тестувати програми;

- ROS підходить для великих систем виконання та для великого процесу розробки [14].

Ця система може бути новою для студентів, і вимагає певного зусилля для її налагодження на комп'ютері. Встановлення ROS може включати в себе багато кроків і вимагати встановлення додаткових пакетів.

Серед багатьох програмних середовищ було обрано Visual Studio Code (рисуюнок 2.19) саме для цього проєкту.



Рисуюнок 2.19 – Візуалізація програми VS Code

Visual Studio Code - це редактор вихідного коду для різних мов програмування, включаючи C, C#, C++, Fortran, Go, Java, JavaScript, Node.js, Python і Rust, заснований на фреймворку Electron. Visual Studio Code використовується для розробки веб-додатків на Node.js, що працюють на движку верстки Blink, і використовує той самий компонент редактора (під кодовою назвою "Monaco"), що і в Azure DevOps.

Visual Studio Code має базову підтримку більшості популярних мов програмування. Ця базова підтримка включає підсвічування синтаксису, узгодження дужок, обгортання коду, настроювані фрагменти та ін. Visual Studio Code також включає IntelliSense для JavaScript, TypeScript, JSON, CSS і HTML. До

нього також входить підтримка налагодження Node.js. Підтримка додаткових мов забезпечується безкоштовними розширеннями в VS Code Marketplace.

Замість системи проєктів користувачі можуть відкрити один або кілька каталогів, які потім можна зберегти в робочій області для подальшого використання. Це робить його редактором коду з мовною діагностикою для будь-якої мови. Він підтримує багато мов програмування і різні набори функцій для кожної мови. Налаштування можна використовувати для виключення небажаних файлів і папок з дерева проєкту; багато функцій Visual Studio Code недоступні через меню або користувацький інтерфейс, але доступні з палітри команд.

Код Visual Studio можна розширювати за допомогою розширень, доступних у центральному сховищі. Сюди входять доповнення до редакторів та мовна підтримка. Серед визначних можливостей - можливість створювати розширення, які додають підтримку нових мов, тем, налагоджувачів і налагоджувачів подорожей у часі, виконують статичний аналіз коду і додають кодові вставки за допомогою протоколу мовного сервера.

Source Control є вбудованою функцією Visual Studio Code. У рядку меню є спеціальна вкладка, яка дозволяє отримати доступ до налаштувань контролю версій і переглянути зміни, зроблені в поточному проєкті. Щоб використовувати цю функцію, Visual Studio Code має бути пов'язаний з підтримуваною системою контролю версій (наприклад, Git, Apache Subversion, Perforce). Це дозволяє користувачам створювати репозиторії та надсилати запити безпосередньо з Visual Studio Code.

Visual Studio Code включає декілька розширень FTP і може використовуватися як безкоштовне альтернативне програмне забезпечення для веб-розробки. Код можна синхронізувати між редактором і сервером без необхідності завантажувати додаткове програмне забезпечення.

Visual Studio Code дозволяє встановлювати сторінку коду, де зберігається активний документ, символ розриву рядка і мову програмування активного документа. Це дозволяє використовувати будь-яку платформу, будь-яку локальну мережу і будь-яку мову програмування [15, 16].

Після завантаження середовища постає питання з вибором мови програмування для робота-пилососа.

Мови програмування C/C++ та Python мають свої властивості та особливості, і вибір між ними залежить від конкретних вимог і завдань. Розберемо основні характеристики обох мов, а також порівняння, яке може обґрунтувати вибір:

Мова C/C++:

- C/C++ відомі своєю високою продуктивністю та ефективністю, що робить їх ідеальними для розробки висконавантажених додатків і систем;
- у C/C++ розробник повинен самостійно керувати пам'яттю, що може призвести до помилок, таких як витоки пам'яті або сегментаційні помилки;
- C/C++ мають складний синтаксис, і програми на цих мовах можуть бути більш об'ємними і вимагати більше коду для досягнення тих самих результатів.

Мова Python:

- Python відомий своєю простою та легко читаємою синтаксисом. Це робить його ідеальним для початківців та прискорює процес розробки;
- Python володіє автоматичним управлінням пам'яттю, що спрощує життя розробників та допомагає уникнути багатьох типових помилок;
- Python має велику кількість сторонніх бібліотек і модулів, які полегшують розробку та розширюють функціональність. Для робототехнічних завдань також доступні бібліотеки, такі як «pySerial» для взаємодії з пристроями через порти.

Роблячи висновок, вищерозглянутих факторів, було зроблено порівняння і вибір певної мови:

Якщо потрібна висока продуктивність або треба розробляти додаток, який працює з великими обсягами даних, то C/C++ може бути кращим вибором.

Однак для початкових користувачів, а також для задач, пов'язаних з робототехнікою, рекомендується вибирати Python. Ця мова забезпечує швидку розробку, менше витрат часу на розробку коду і більше можливостей для використання готових бібліотек [17, 18].

У контексті робототехнічних завдань, де зручність та швидкість розробки мають значення, Python стає на перший план завдяки своїй простоті та багатій екосистемі.

Тому з офіційного сайту було завантажено та встановлено, останню версію Python (рисунок 2.20), а саме 3.12.0 з усіма необхідними параметрами.

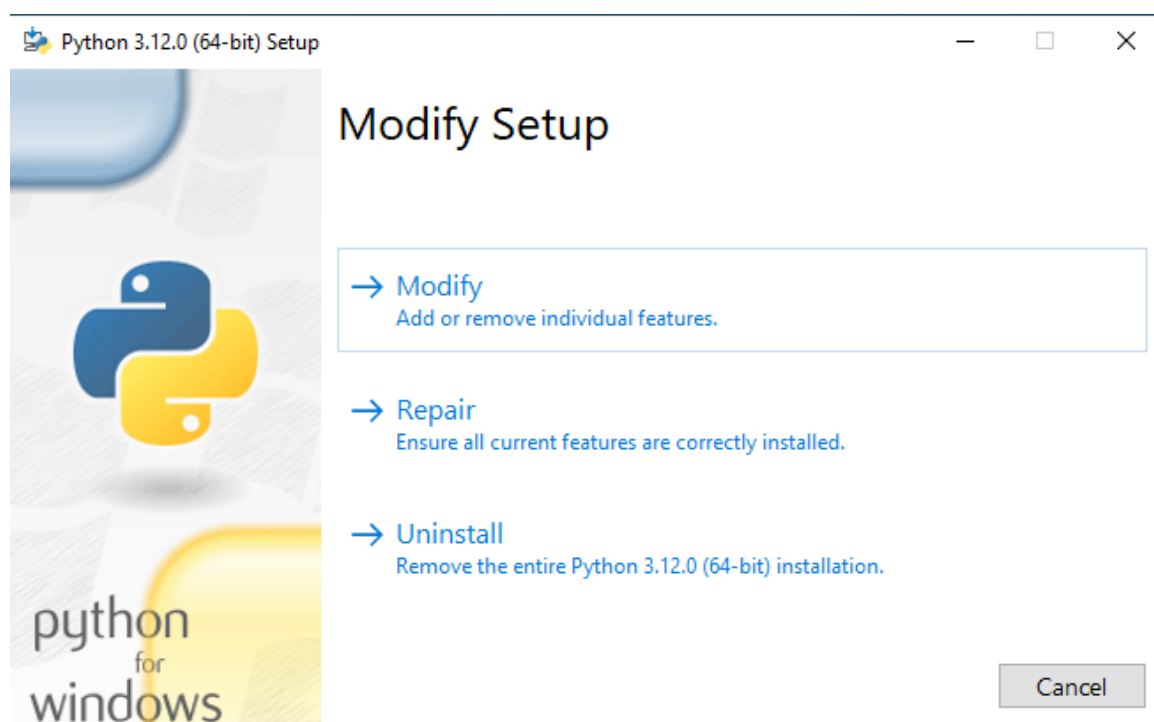


Рисунок 2.20 – Візуалізація налаштування Python

2.2.4 Рекомендації з виконання завдань

У даному розділі описано розроблені завдання для студентів магістратури спеціальності 123 Комп'ютерна інженерія, які враховують використання програмного забезпечення, яке було розглянуто у попередньому пункті (п. 2.2.3) та з використанням специфічного обладнання, яке було описано у п. 2.2, а саме робота-пилососа iRobot Create 2.

1 завдання полягає у наступному. Необхідно запрограмувати робот таким чином, щоб виконувались наступні дії:

- роботу необхідно здати назад на 10 см, тобто виїхати зі станції базування;
- розвернутися на 180 градусів;

- необхідно проїхати вперед на 30 см;
- повернути праворуч на 90 градусів;
- проїхати вперед 30 см;
- повернути праворуч на 90 градусів;
- проїхати вперед 30 см;
- повернути праворуч на 90 градусів;
- проїхати вперед 30 см, тим самим утворити фігуру «квадрат»;
- повернути ліворуч на 90 градусів та заїхати на станцію, проїхавши 10 см вперед.

Для вирішення цього завдання першочергово необхідно встановити для Python бібліотеку для керування iRobot Create 2, а саме «pycreate2». Далі у проєкті вказуємо назву порту для підключення до роботу.

Ось невеликий перелік команд, який допоможе створити програмний код для керування роботом:

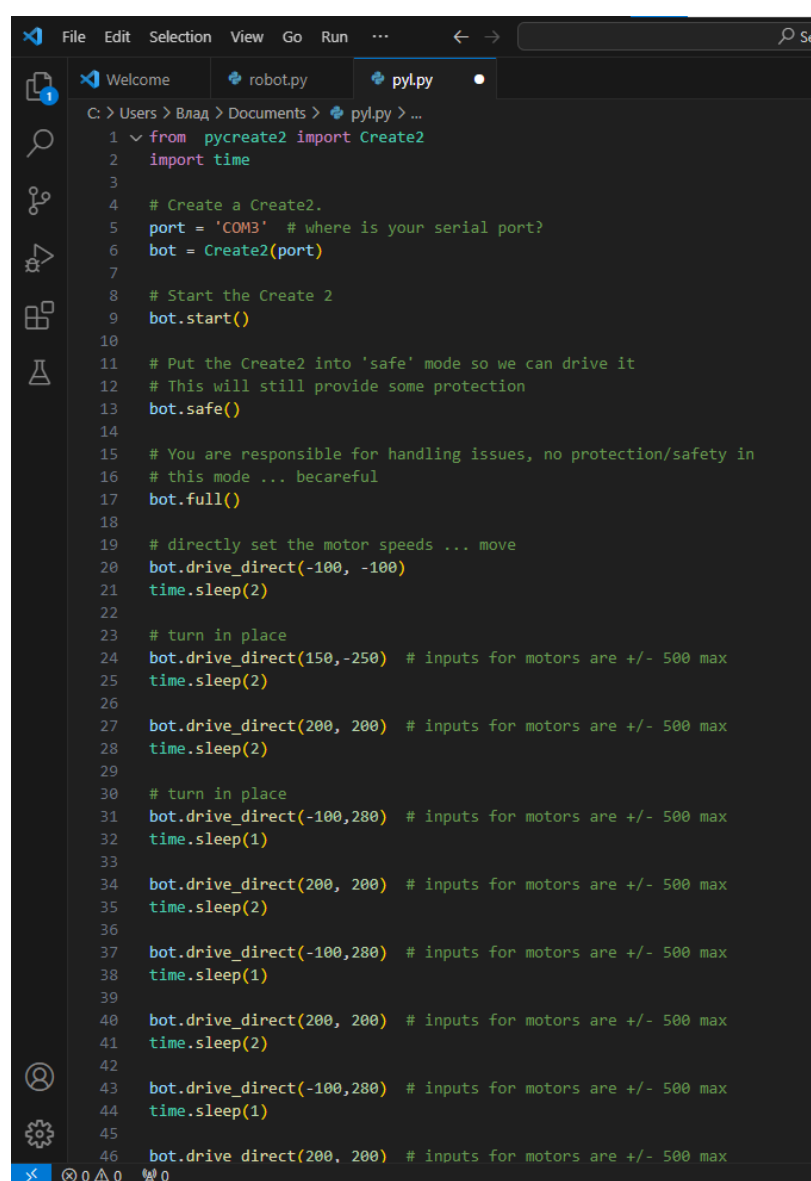
- `bot.start()` – старт роботи;
- `bot.safe()` – перехід Create2 у «безпечний» режим, щоб ним можна було керувати;
- `right_wheel_speed = до 500` – встановлення бажаної швидкості для оберту правого колеса;
- `left_wheel_speed = до 500` – встановлення бажаної швидкості для оберту лівого колеса;
- `distance_mm = 100` – завдання дистанції, яку необхідно проїхати у міліметрах;
- `distance_mm_per_rotation = 260` – зразкова відстань в міліметрах, яку робот проходить за один оберт колеса;
- `bot.drive_direct(right_wheel_speed, left_wheel_speed)` або команда може виглядати наступним чином `bot.drive_direct(100,100)` – відправлення команди на рух при заданих умовних швидкостях;

– `time.sleep(2)` – час у секундах, який дається на виконання певної команди, після вичерпання часу робот зупиняється і виконує наступну команду;

– `bot.drive_stop()` – зупинити робота;

– `bot.close()` – закриття з'єднання.

Приклад коду виконання завдання зображено на рисунку 2.21, а повний лістинг коду наведено у Додатку А. Також код можна модернізувати та скоротити його використовуючи цикл, бо певні умови повторюються. Модернізований код до цього завдання зображено на рисунку 2.22



```

1  from pycreate2 import Create2
2  import time
3
4  # Create a Create2.
5  port = 'COM3' # where is your serial port?
6  bot = Create2(port)
7
8  # Start the Create 2
9  bot.start()
10
11 # Put the Create2 into 'safe' mode so we can drive it
12 # This will still provide some protection
13 bot.safe()
14
15 # You are responsible for handling issues, no protection/safety in
16 # this mode ... becareful
17 bot.full()
18
19 # directly set the motor speeds ... move
20 bot.drive_direct(-100, -100)
21 time.sleep(2)
22
23 # turn in place
24 bot.drive_direct(150, -250) # inputs for motors are +/- 500 max
25 time.sleep(2)
26
27 bot.drive_direct(200, 200) # inputs for motors are +/- 500 max
28 time.sleep(2)
29
30 # turn in place
31 bot.drive_direct(-100, 280) # inputs for motors are +/- 500 max
32 time.sleep(1)
33
34 bot.drive_direct(200, 200) # inputs for motors are +/- 500 max
35 time.sleep(2)
36
37 bot.drive_direct(-100, 280) # inputs for motors are +/- 500 max
38 time.sleep(1)
39
40 bot.drive_direct(200, 200) # inputs for motors are +/- 500 max
41 time.sleep(2)
42
43 bot.drive_direct(-100, 280) # inputs for motors are +/- 500 max
44 time.sleep(1)
45
46 bot.drive_direct(200, 200) # inputs for motors are +/- 500 max

```

Рисунок 2.21 – Приклад коду програми для виконання завдання 1

```

1  from pycrate2 import Create2
2  import time
3
4  # Create a Create2.
5  port = 'COM3' # replace with your COM port
6  bot = Create2(port)
7
8  # Start the Create 2
9  bot.start()
10
11 # Put the Create2 into 'safe' mode so we can drive it
12 bot.safe()
13
14 # We calculate the travel time to travel 30 cm (estimated speed 100 mm/s)
15 travel_time = 3 # time in seconds
16
17 # directly set the motor speeds ... move forward
18 bot.drive_direct(100, 100)
19 time.sleep(travel_time)
20
21 # turn in place (90 degrees right)
22 bot.drive_direct(200, -200)
23 time.sleep(1)
24
25 # Repeat the same steps for each side of the square
26 for _ in range(3):
27     # move forward
28     bot.drive_direct(100, 100)
29     time.sleep(travel_time)
30
31     # turn in place (90 degrees right)
32     bot.drive_direct(200, -200)
33     time.sleep(1)
34
35 # Stop the bot
36 bot.drive_stop()
37
38 # Close the connection
39 bot.close()

```

Рисунок 2.22 – Приклад коду програми для виконання завдання 1, модернізований варіант

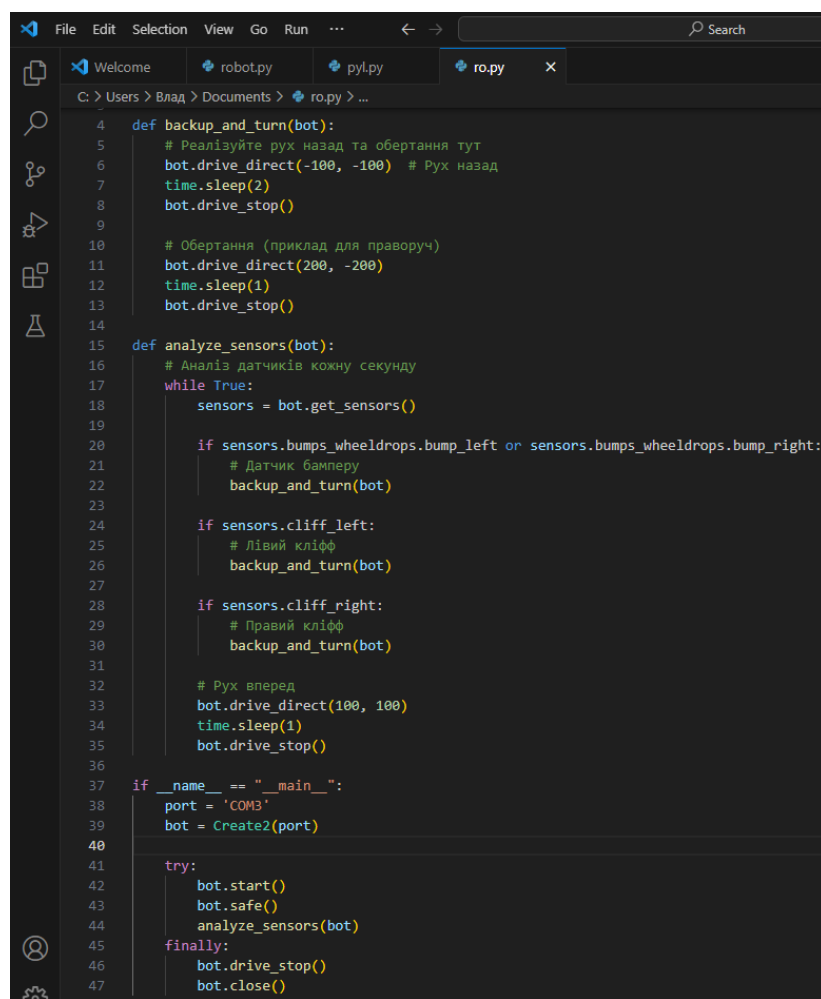
2 завдання полягає у наступному. Необхідно запрограмувати робот-пилосос таким чином, щоб виконувались такі дії:

- роботу необхідно покинути станцію базування, здати назад та розвернутися на 180 градусів;
- щоб кожні пройдені, наприклад, 20 см робот робив зупинку, аналізував датчики виявлення перешкоди, кліфф-сенсор, удар бампером;
- обмін даних з датчиками повинен відбуватися щосекунди;
- коли спрацює будь-який датчик, який реагує на перешкоду робот повинен зупинитися;
- від'їхати 10 см;
- зробити розворот на 30 градусів і спробувати ще раз подолати перешкоду;
- виконання завдання завершити після проходження усіх перешкод.

Розв'язати це завдання можна наступним чином. Для цього наведемо невеликий перелік команд, який допоможе створити програмний код для керування роботом:

- sensors.bumps_wheeldrops.bump_left / sensors.bumps_wheeldrops.bump_right – датчик удару лівого та правого бамперу;
- sensors.cliff_left or sensors.cliff_right – кліфф-сенсори допомагають роботу уникнути падіння з високих платформ або сходин.

Приклад коду виконання завдання зображено на рисунку 2.23, а повний лістинг коду наведено у Додатку Б. Також код можна модернізувати так, щоб Roomba під час руху не зупиняючись аналізував датчики та від'їжджав від перешкоди на 5 см назад, робив кут десь 15-20 градусів, та робив спробу ще раз об'їхати перешкоду.



```

4 def backup_and_turn(bot):
5     # Реалізуйте рух назад та обертання тут
6     bot.drive_direct(-100, -100) # Рух назад
7     time.sleep(2)
8     bot.drive_stop()
9
10    # Обертання (приклад для праворуч)
11    bot.drive_direct(200, -200)
12    time.sleep(1)
13    bot.drive_stop()
14
15 def analyze_sensors(bot):
16     # Аналіз датчиків кожну секунду
17     while True:
18         sensors = bot.get_sensors()
19
20         if sensors.bumps_wheeldrops.bump_left or sensors.bumps_wheeldrops.bump_right:
21             # Датчик бамперу
22             backup_and_turn(bot)
23
24         if sensors.cliff_left:
25             # Лівий кліфф
26             backup_and_turn(bot)
27
28         if sensors.cliff_right:
29             # Правий кліфф
30             backup_and_turn(bot)
31
32         # Рух вперед
33         bot.drive_direct(100, 100)
34         time.sleep(1)
35         bot.drive_stop()
36
37 if __name__ == "__main__":
38     port = 'COM3'
39     bot = Create2(port)
40
41     try:
42         bot.start()
43         bot.safe()
44         analyze_sensors(bot)
45     finally:
46         bot.drive_stop()
47         bot.close()

```

Рисунок 2.23 – Приклад коду програми для виконання завдання 2

3 завдання полягає у наступному. Необхідно запрограмувати робот-пилосос таким чином, щоб пилосос відігравав мелодію за нотами.

Розв'язати це завдання можна наступним чином. При створенні пісні, яку буде виконувати Roomba, слід враховувати те, що «спів» робота базується на нотах MIDI.

Послідовність команд для визначення пісні має наступні властивості:

– [140] [Номер пісні] [Довжина пісні] [Нота ноти 1] [Тривалість ноти 1]...[Номер ноти N] [Тривалість ноти N], де:

– 140 – повідомляє Roomba, що користувач збирається визначити пісню. (це збірка нотаток);

– [Номер пісні] – моделі роботів Roomba 4XX, Create і Dirt можуть зберігати до 16 пісень; отже, це число може бути лише в діапазоні 0-15;

– інші моделі Roomba (5XX і 7XX) містять лише чотири попередньо визначені пісні (номер пісні може бути 0-3). Якщо пісня вже була визначена під цим номером, її буде замінено новою піснею. Визначення пісень зберігаються, доки Roomba не буде вимкнено;

– [Довжина пісні] – кількість нот, які будуть надіслані. Максимальна кількість нот в одній пісні – 16; отже, діапазон становить від 0 до 15;

– [Номер ноти 1] – команда Byte, яка представляє ноту відповідно до таблиці нижче; Діапазон командного байта становить від 31 до 127. На рисунку 2.24 наведені ці діапазони;

– значення від 60 до 83 є найбільш рекомендованими нотами з точки зору «приємних звуків»;

– [Тривалість ноти] – тривалість ноти, указана з кроком 1/64 секунди [19].

Number	Note	Frequency	Number	Note	Frequency	Number	Note	Frequency
31	G	49.0	58	A#	233.1	85	C#	1108.8
32	G#	51.9	59	B	246.9	86	D	1174.7
33	A	55.0	60	C	261.6	87	D#	1244.5
34	A#	58.3	61	C#	277.2	88	E	1318.5
35	B	61.7	62	D	293.7	89	F	1396.9
36	C	65.4	63	D#	311.1	90	F#	1480.0
37	C#	69.3	64	E	329.6	91	G	1568.0
38	D	73.4	65	F	349.2	92	G#	1661.3
39	D#	77.8	66	F#	370.0	93	A	1760.0
40	E	82.4	67	G	392.0	94	A#	1864.7
41	F	87.3	68	G#	415.3	95	B	1975.6
42	F#	92.5	69	A	440.0	96	C	2093.1
43	G	98.0	70	A#	466.2	97	C#	2217.5
44	G#	103.8	71	B	493.9	98	D	2349.4
45	A	110.0	72	C	523.3	99	D#	2489.1
46	A#	116.5	73	C#	554.4	100	E	2637.1
47	B	123.5	74	D	587.3	101	F	2793.9
48	C	130.8	75	D#	622.3	102	F#	2960.0
49	C#	138.6	76	E	659.3	103	G	3136.0
50	D	146.8	77	F	698.5	104	G#	3322.5
51	D#	155.6	78	F#	740.0	105	A	3520.1
52	E	164.8	79	G	784.0	106	A#	3729.4
53	F	174.6	80	G#	830.6	107	B	3951.2
54	F#	185.0	81	A	880.0			
55	G	196.0	82	A#	932.4			
56	G#	207.7	83	B	987.8			
57	A	220.0	84	C	1046.5			

Рисунок 2.24 – Діапазон командного байту (номер ноти)

Як за приклад, розглянемо, як запрограмувати Roomba на відтворення пісні Jingle Bells. Для цього необхідно надіслати на виконання наступні команди:

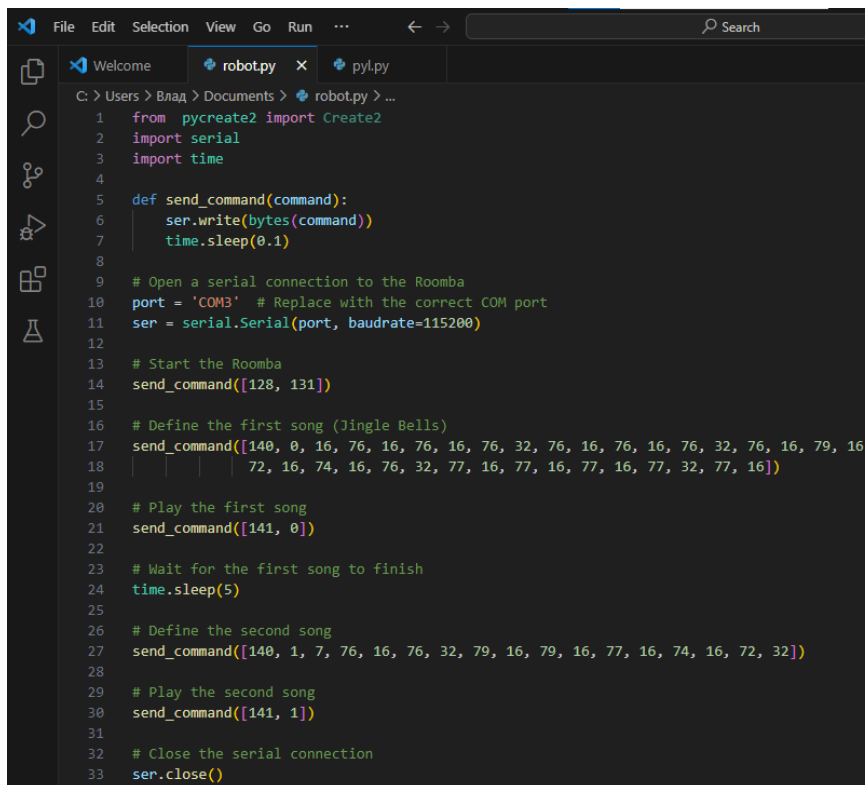
- 128 131
- 140 0 16
- 76 16 76 16 76 32 76 16 76 16 76 32 76 16 79 16
- 72 16 74 16 76 32 77 16 77 16 77 16 77 32 77 16
- 140 1 7
- 76 16 76 32 79 16 79 16 77 16 74 16 72 32
- 141 0
- 141 1

У цьому переліку кожна команда має наступне пояснення:

- [128 131] (запустити потік команд і змінити режим Roomba на «Безпечний» режим);
- [140 0] (почати визначення пісні та визначити першу пісню, номер пісні0);

- [16] (для пісні під номером 0 буде надіслано 16 нот);
- [76 16] (відтворення звуку Мі протягом 1/4 секунди);
- [76 16] (відтворення звуку Мі протягом 1/4 секунди);
- [76 32] (відтворення звуку Мі протягом 1/2 секунди);
- [76 16] (відтворення звуку Мі протягом 1/4 секунди);
- [76 16] (відтворення звуку Мі протягом 1/4 секунди);
- [76 32] (відтворення звуку Мі протягом 1/2 секунди);
- [76 16] (відтворення звуку Мі протягом 1/4 секунди);
- [79 16] (відтворення звуку Sol протягом 1/4 секунди);
- [72 16] (відтворення звуку протягом 1/4 секунди);
- [74 16] (повторне відтворення звуку протягом 1/4 секунди);
- [76 32] (відтворення звуку Мі протягом 1/2 секунди);
- [77 16] (відтворення звуку Фа протягом 1/4 секунди);
- [77 16] (відтворення звуку Фа протягом 1/4 секунди);
- [77 16] (відтворення звуку Фа протягом 1/4 секунди);
- [77 32] (відтворення звуку фа протягом 1/2 секунди);
- [77 16] (відтворення звуку Фа протягом 1/4 секунди);
- [140 1] (Почати визначення пісні та визначити другу пісню, пісня номер1);
- [7] (для пісні № 1 буде надіслано 7 нот);
- [76 16] (відтворення звуку Мі протягом 1/4 секунди);
- [76 32] (відтворення звуку Мі протягом 1/2 секунди);
- [79 16] (відтворення звуку Sol протягом 1/4 секунди);
- [79 16] (відтворення звуку Sol протягом 1/4 секунди);
- [77 16] (відтворення звуку Фа протягом 1/4 секунди);
- [74 16] (повторне відтворення звуку протягом 1/4 секунди);
- [72 32] (відтворення звуку протягом 1/2 секунди);
- [141 0] (відтворити пісню номер 0);
- [141 1] (відтворити пісню № 1).

Після більш детального пояснення, необхідно ці команди зібрати у повноцінний код для виконання Roomba, цей код зображено на рисунку 2.25, а також повний лістинг коду наведено у Додатку В.



```

1  from pycreate2 import Create2
2  import serial
3  import time
4
5  def send_command(command):
6      ser.write(bytes(command))
7      time.sleep(0.1)
8
9  # Open a serial connection to the Roomba
10 port = 'COM3' # Replace with the correct COM port
11 ser = serial.Serial(port, baudrate=115200)
12
13 # Start the Roomba
14 send_command([128, 131])
15
16 # Define the first song (Jingle Bells)
17 send_command([140, 0, 16, 76, 16, 76, 16, 76, 32, 76, 16, 76, 16, 76, 32, 76, 16, 79, 16,
18 | | | 72, 16, 74, 16, 76, 32, 77, 16, 77, 16, 77, 16, 77, 32, 77, 16])
19
20 # Play the first song
21 send_command([141, 0])
22
23 # Wait for the first song to finish
24 time.sleep(5)
25
26 # Define the second song
27 send_command([140, 1, 7, 76, 16, 76, 32, 79, 16, 79, 16, 77, 16, 74, 16, 72, 32])
28
29 # Play the second song
30 send_command([141, 1])
31
32 # Close the serial connection
33 ser.close()

```

Рисунок 2.25 – Приклад коду програми для виконання завдання 3

3 РОЗРОБКА ВІРТУАЛЬНОЇ ЛАБОРАТОРІЇ

3.1 Структурна схема лабораторії

При ретельному аналізі схеми розміщення персональних комп'ютерів у аудиторії № 52, що зображена на рисунку 3.1, можна визначити оптимальний спосіб підключення робота-пилососа до комп'ютера з метою ефективних лабораторних досліджень та програмування. Основний критерій вибору місця для розташування робота-пилососа полягає у забезпеченні зручності підключення та оптимальних умов для його функціонування.

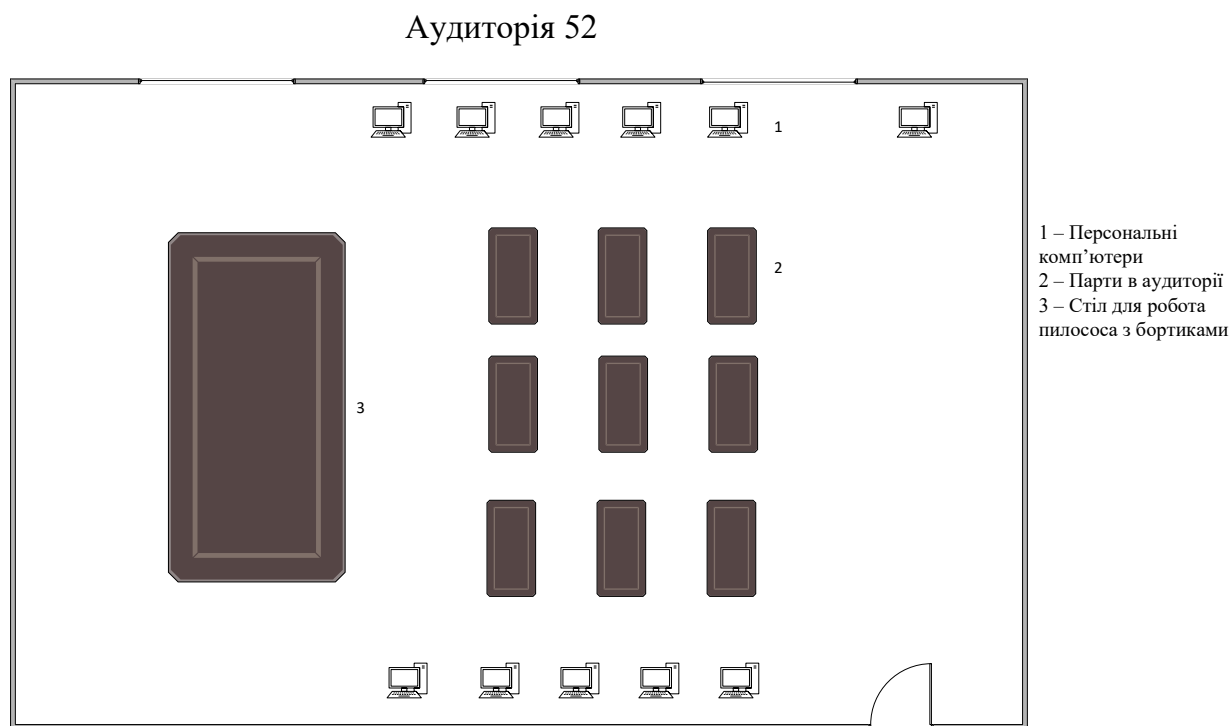


Рисунок 3.1 – Планограма аудиторії № 52

Один із найбільш оптимальних варіантів полягає в використанні останнього персонального комп'ютера, розташованого біля останнього вікна аудиторії. Це обґрунтовується кількома факторами:

- останній комп'ютер знаходиться найближче до столу, що використовується для експериментів з роботом-пилососом. Це забезпечує зручний доступ до робота і столу, що є важливим для навчальних експериментів;

- завдяки цій локації можна винести останній комп'ютер ближче до робочого столу для пилососа, що забезпечить додатковий простір для рухів робота під час виконання програми або алгоритму. Це сприяє уникненню перешкод та забезпечує плавний рух робота під час виконання завдань;

- такий варіант розміщення дозволяє забезпечити необхідний простір для підключення кабелю від комп'ютера до робота-пилососа. Це важливо для передачі програм та даних на робота та від нього до комп'ютера.

З урахуванням цього розроблено структурну схему віртуальної лабораторії, яка зображена на рисунку 3.2. Структурна схема має більш чітке пояснення виду

комунікації з віртуальною лабораторією. Від користувача, а саме, від студента або викладача необхідно мати мінімальний набір гаджетів для виходу в Інтернет, це – смартфон або планшет або ноутбук (нетбук), або комп’ютер. Враховуючи фактор війни в країні, необхідний будь-який пристрій з можливістю виходу до Інтернету для віддаленого доступу до лабораторії. В цей час, до комп’ютера в університеті має бути обов’язково під’єднана вебкамера, щоб спостерігати за виконанням практичних завдань, пов’язаних із спеціалізованим обладнанням, у реальному часі, і безпосередньо це спеціалізоване обладнання. У випадку розроблених завдань у даному дипломному проєкті, таким спеціалізованим обладнанням є робот-пилосос.

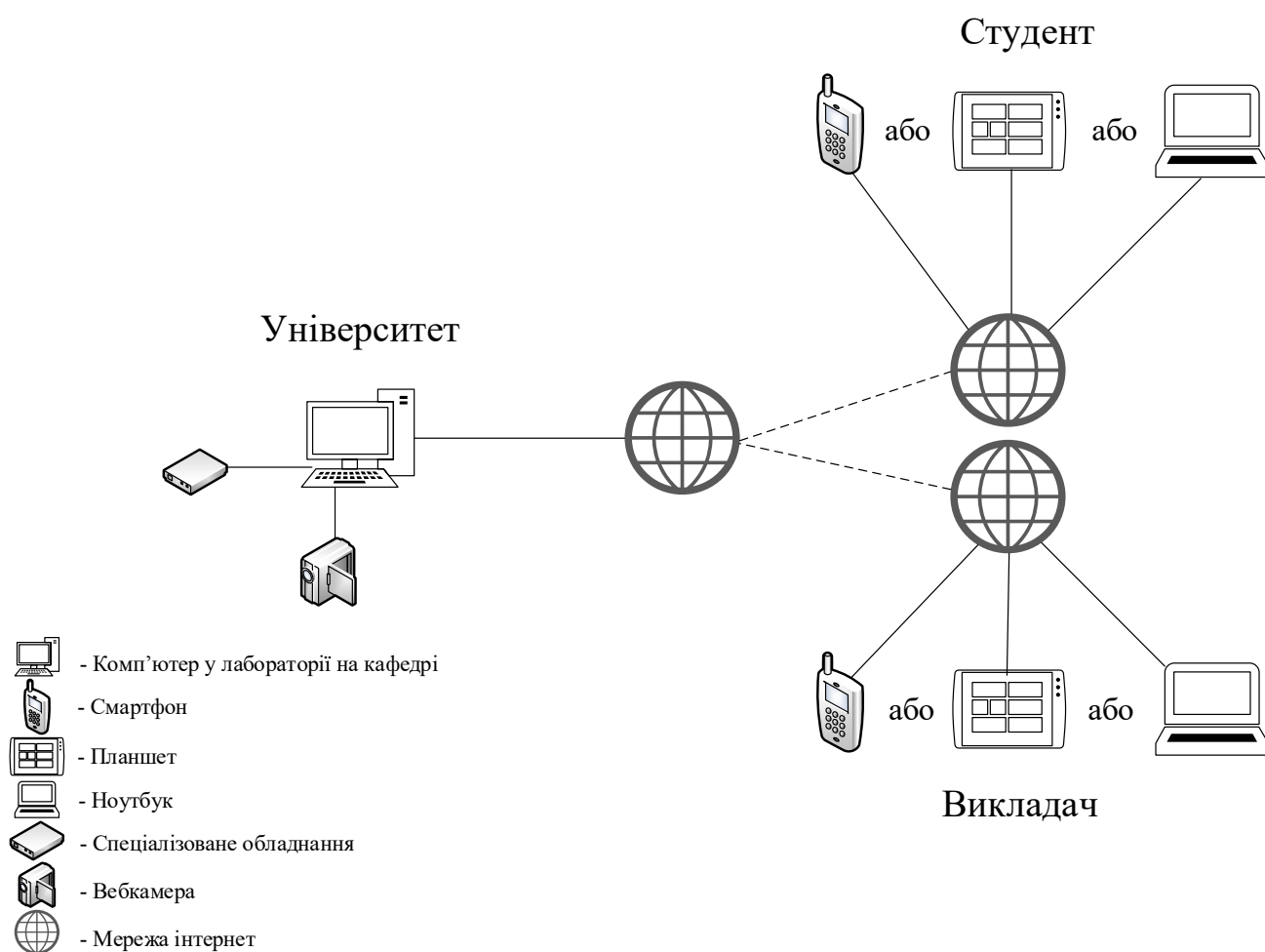


Рисунок 3.2 – Структурна схема віртуальної лабораторії

Вигляд аудиторії №52 навчальної лабораторії «Системної інтеграції комп’ютерних технологій» зображено на рисунках 3.3 та 3.4.



Рисунок 3.3 – Фото аудиторії №52



Рисунок 3.4 – Фотознімок аудиторії №52

Враховуючи всі ці фактори, вибір місця для розміщення робота-пилососа підтримує ефективне проведення лабораторних досліджень та навчальних занять з програмування на практиці.

3.2 Функціонування лабораторії

Управління віртуальною лабораторією віддалено передбачає врахування різних аспектів та потреб, включаючи як технічні, так і людські фактори. Більш детально розглянемо ті аспекти, які потрібно враховувати:

- запуск та моніторинг ПК;
- ймовірність перебоїв з електропостачанням;
- підключення пристроїв;
- робота асистента;
- запуск лабораторних завдань.

Враховуючи віддалене управління, маємо можливість включення і вимкнення комп'ютерів з віддаленої позиції. Також, є система моніторингу для відстеження працездатності комп'ютерів та вчасного виявлення можливих проблем.

В умовах війни і перебоїв в електропостачанні можуть виникати проблеми з живленням комп'ютерів. Враховуючи це, маємо можливість віддалено перезавантажувати комп'ютери або включати їх, якщо постачання електроенергії відновлюється. Це запобігає втраті даних та забезпечує неперервну роботу лабораторії.

В деяких віртуальних лабораторних роботах є можливість фізично підключити пристрої до комп'ютерів. Наприклад, у наступному розділі буде описано робот-пилосос, для підключення якого використовується адаптер. Також може бути необхідно встановити пилосос на його станцію живлення. Для цього асистент повинен виконати необхідні дії в аудиторії.

Потрібно враховувати, що наявність асистента на початковому етапі надзвичайно важлива. Асистент відповідає за запуск комп'ютерів та забезпечення їхньої працездатності. Він також відповідає за підключення різних пристроїв до ПК, що може бути складним завданням, наприклад, у випадку підключення робота-пилососа. Асистент вирішує технічні проблеми та координує роботу в лабораторії.

Асистент може бути відповідальним за запуск лабораторних завдань на комп'ютерах та надавати інструкції студентам щодо проведення експериментів.

Загалом, віддалене управління лабораторією враховує як технічні, так і організаційні аспекти, щоб забезпечити ефективну та безпечну роботу віртуальної лабораторії у всіх умовах, включаючи війну та перебої в енергопостачанні.

3.3 Вибір програмного та апаратного забезпечення системи

3.3.1 Апаратна складова в межах університету

Під час аналізу апаратної складової для створення віртуальної лабораторії, необхідно враховувати наявність вже наявного обладнання та інфраструктури в університеті. Ретельний огляд показав, що в університетській споруді існують всі необхідні компоненти для успішної реалізації цього проєкту.

Головні аспекти для цього, це:

- першою важливою складовою є наявність стабільного живлення будівлі, що гарантує постійне живлення для всіх пристроїв та обладнання, необхідних для віртуальної лабораторії;
- університет також має надійне та швидке підключення до мережі Інтернету. Це дозволить студентам і викладачам безперервно користуватися віртуальною лабораторією без затримок чи перебоїв у мережі;
- університет оновив обладнання, і в даний час в наявності є персональні комп'ютери, які відповідають більш ніж мінімальним вимогам до програмного забезпечення, необхідного для створення віртуальної лабораторії.

Персональні комп'ютера в аудиторії № 52 мають наступні апаратні характеристики:

- процесор - AMD Ryzen 5 PRO 4650G;
- материнська плата - ASUS TUF GAMING A-520M-PLUS;
- оперативна пам'ять - Kingston HYPERX, дві планки по 16gb. Загальний обсяг 32gb;
- пам'ять, SSD - Samsung 970EVO Plus 250gb;
- пам'ять HDD - Toshiba PC P300 1tb;
- блок живлення - GameMax GM-500w;
- монітор - Asus BE249QLBH;
- системний корпус - Qube hardware solution.

Персональні комп'ютера в аудиторії № 52 мають наступні програмні характеристики:

- на ПК встановлено операційна система Windows 10 з усіма її компонентами;
- встановлений пакет Microsoft Office;
- браузер Google Chrome;
- програма для віддаленого доступу TeamViewer останньої версії;
- мова програмування Python версії 3.12.0 з усіма модулями;
- програмне середовище для програмування Visual Studio Code.

Це означає, що наявне обладнання може бути використане для реалізації проекту, наприклад, програмування робота-пилососа або інших подібних проєктів, які вимагають великих ресурсів від користувача.

Враховуючи ці фактори, маємо можливість створити віртуальну лабораторію на основі наявних складових, що сприятиме ефективному навчанню та дослідженням в навчальному закладі.

3.3.2 Додаткове устаткування

Наступним етапом в розгляді питань стосовно підключення робота-пилососа до комп'ютера є вирішення того, як спостерігати за його рухами під час

завантаження програмного коду. Для досягнення цієї мети може бути використана звичайна вебкамера, яка може бути легко встановлена на системний блок комп'ютера.

Камера повинна мати роздільну здатність 1920x1080. Це зручний формат відео, який легко обробить звичайна мережа в 100Mb/s, і навіть LTE. У такої роздільної здатності досить чітке зображення, при цьому ресурси комп'ютера не перевантажені. Також пристрій повинен мати штатив або кріплення до корпусу ПК, щоб зафіксувати його у потрібному положенні.

Вебкамера має постійно тримати об'єкт зйомки в кадрі, тому під час спостереження за експериментами важливий її кут огляду. Цей параметр варіюється від 45 до 170°. Для даного проєкту потрібна камера з великим кутом огляду, від 110°.

Застосування вебкамери дозволить отримати візуальний контроль за рухами робота-пилососа під час виконання програми або завантаження коду. Вона може бути розміщена зверху на системному блоці, надаючи можливість спостерігати за пилососом з висоти.

Ця вебкамера допомагає вирішити важливу задачу спостереження та аналізу рухів робота-пилососа під час виконання завдань, а також може бути використана для відстеження ефективності програми або алгоритму. Такий підхід сприяє покращенню якості лабораторних досліджень та навчання програмування, надаючи можливість візуально контролювати роботу пилососа під час виконання різноманітних завдань.

3.3.3 Апаратна складова для користувача (студент/викладач)

Враховуючи складну ситуацію в Україні через війну, а також той факт, що багато студентів і викладачів змушені тимчасово залишати свої постійні місця проживання і переїжджати вглиб країни або за кордон, практично неможливо придбати і взяти з собою велику кількість обладнання та витратних матеріалів. За таких обставин важливо враховувати наявність базових гаджетів, такі як смартфони, і використовувати їх для продовження навчання та спілкування.

Особливу увагу слід звернути на мінімальну кількість гаджетів, зокрема смартфонів, при розгляді питання доступу до віртуальних лабораторій. Смартфони є важливими інструментами для навчання, оскільки це пристрої, які є у більшості людей.

У попередньому розділі було розглянуто три найпопулярніші платформи для управління віддаленим доступом: TeamViewer, Chrome Remote Desktop та Anydesk. Важливо зазначити, що кожна з цих програм має сумісну мобільну версію і може бути безкоштовно завантажена з магазинів додатків Google Play або AppStore для смартфонів. Це важливий аспект надання студентам і викладачам доступу до віртуальної лабораторії, навіть у віддаленому та обмеженому середовищі. (Див. рис. 3.5).

Більшість програмного забезпечення для віддаленого доступу підтримує мобільні додатки, які максимально спрощують процес підключення до віртуальної лабораторії. Важливо пам'ятати, що немає сенсу обмежувати використання лише одного додатка, оскільки користувачі можуть мати різний досвід та вподобання.

Аналіз розміру програмних компонентів, що встановлюються на комп'ютери в аудиторії, показує, що вони не є великими за розміром, тому можна встановити всі необхідні програми на одній машині. Студенти та викладачі можуть бути проінформовані про доступ до цього комп'ютера шляхом надання ідентифікатора доступу та пароля для підключення до сеансу. Це спрощує процес доступу до віртуальної лабораторії та робить його більш зручним для всіх користувачів.

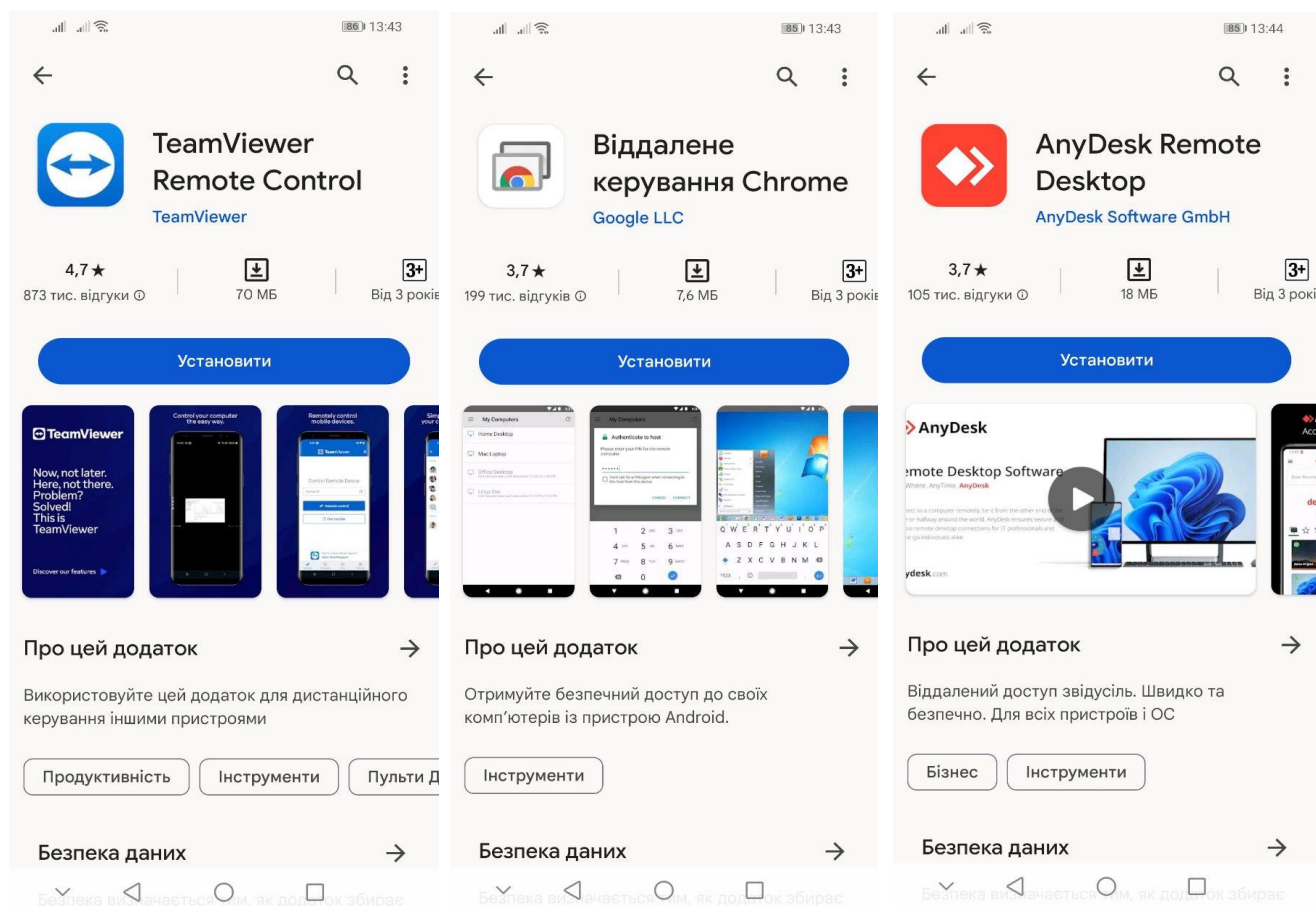


Рисунок 3.5 – Скриншоти додатків з Google Play

3.4 Збереження цілісності даних

Збереження цілісності даних завжди було і залишається актуальною проблемою. Зберігання даних на різних носіях, таких як жорсткі диски, флешки і інші, не завжди гарантує повну безпеку і надійність інформації. Існують численні фактори, які можуть спричинити втрату даних, включаючи відключення електроенергії, перенапругу в мережі, природні катастрофи, а також людський фактор, такий як випадкова втрата носія інформації.

У зв'язку з цим одним із найбільш надійних та ефективних способів збереження інформації є використання хмарних середовищ. Хмарні технології дозволяють зберігати дані на віддалених серверах, які підтримують високу доступність та резервне копіювання. Це забезпечує захист даних від втрати у

випадку непередбачених обставин, і забезпечує їх цілісність та доступність для користувачів у будь-який момент.

На сьогоднішній день існує багато різних хмарних сервісів, які надають різні функціональні можливості для зберігання та обробки даних в хмарі. Деякі з найбільш розповсюджених та відомих хмарних сервісів включають:

- Google Drive;
- Microsoft OneDrive;
- Dropbox.

Google Drive - це сервіс зберігання, редагування та синхронізації файлів, розроблений Google.

Надаючи доступ до файлів, Disk використовує шифрування для забезпечення високого рівня безпеки, а оскільки Drive - це хмарний сервіс, немає необхідності зберігати файли локально, що знижує ризик для пристрою.

Диск інтегровано з хмарними додатками, такими як Документи, Таблиці та Презентації, для ефективної співпраці в режимі реального часу.

Drive сумісний з технологіями, які використовує ваша команда, і доповнює їх; немає необхідності конвертувати файли Microsoft Office в інші формати для спільної роботи; ви можете редагувати і зберігати понад 100 типів файлів, включаючи PDF, CAD тощо.

Спеціальні інструменти, реалізовані в сервісі Диск, забезпечують виняткову швидкість, ефективність і надійність пошуку Google. Крім того, готові пошукові фільтри дозволяють командам швидко знаходити потрібні файли [20].

OneDrive - хмарний сервіс від Microsoft, який забезпечує доступ до всіх ваших файлів. Файли можна зберігати та захищати, ділитися ними з іншими користувачами та отримувати до них доступ з будь-якого місця на всіх пристроях [21].

Dropbox - сервіс файлового хостингу від компанії Dropbox Inc., що включає персональне хмарне сховище, синхронізацію файлів і клієнтські програми. Штаб-квартира компанії знаходиться в Сан-Франциско.

Dropbox дозволяє користувачам створювати спеціальні папки на своїх комп'ютерах, які Dropbox синхронізує таким чином, щоб один і той самий вміст можна було переглядати на будь-якому пристрої. Файли в цій папці також доступні через веб-сайт Dropbox і мобільні додатки. Dropbox працює за моделлю freemium, де користувачі мають можливість створити безкоштовний обліковий запис з певним обсягом, але для збільшення обсягу свого облікового запису необхідна платна підписка [22].

Порівнявши різні хмарні сервіси, було вирішено зупинити вибір на Google Диску. Хоча кожне хмарне середовище має свої сильні та слабкі сторони, Google Диск виділяється своєю загальною популярністю та високим авторитетом серед користувачів. Ця популярність спрощує процес адаптації для нових користувачів і дозволяє їм швидше почати використовувати сервіс для зберігання та обміну даними.

Однією з головних переваг Google Диску є його легка інтеграція з іншими сервісами Google, такими як Gmail та Google Docs. Це полегшує обмін файлами та спільну роботу над документами для користувачів, які вже користуються іншими продуктами Google.

Крім того, Google Диск пропонує безкоштовне сховище, що робить його доступним для багатьох користувачів. За додаткову плату обсяг сховища можна розширити за потреби.

Слід зазначити, що вибір хмарного сервісу може залежати від конкретних потреб і цілей користувача. Однак Google Диск є привабливим варіантом для широкого кола користувачів завдяки своїй простоті та популярності.

3.5 Налаштування роботи системи

Для ефективного використання Google Диску у віртуальній лабораторії необхідно виконати кілька важливих кроків. Перш за все, потрібно налаштувати

доступ студентів та викладачів до відповідних розділів та папок.

Спочатку потрібно створити поштові скриньки для кафедри, де будуть зберігатися файли віртуальної лабораторії. Далі створюються папки для кожної дисципліни на кафедрі, що допомагає організувати матеріали.

Кожна папка дисципліни повинна містити підпапки для груп студентів, а в кожній групі - папки для окремих студентів. Це допомагає зберігати та організовувати файли відповідно до структури курсу.

Для того, щоб студенти та викладачі мали доступ до відповідних папок, використовується опція «Надати доступ», далі «Поділитись» на Google Диску. Для кожної папки визначається, кому надається доступ. Тут, на рисунку 3.6, можна обрати два основні способи:

– доступ за посиланням: цей метод надає можливість створити посилання на папку та надіслати його студентам. Однак цей метод може бути менш безпечним, оскільки посилання може бути розповсюджене між студентами і надавати доступ до папки іншим користувачам;

– доступ за електронною адресою: цей метод передбачає введення електронних адрес студентів та викладачів. Після надсилення запрошення на пошту, користувачі можуть приймати доступ до папок, і їхні індивідуальні файли залишаються приватними.

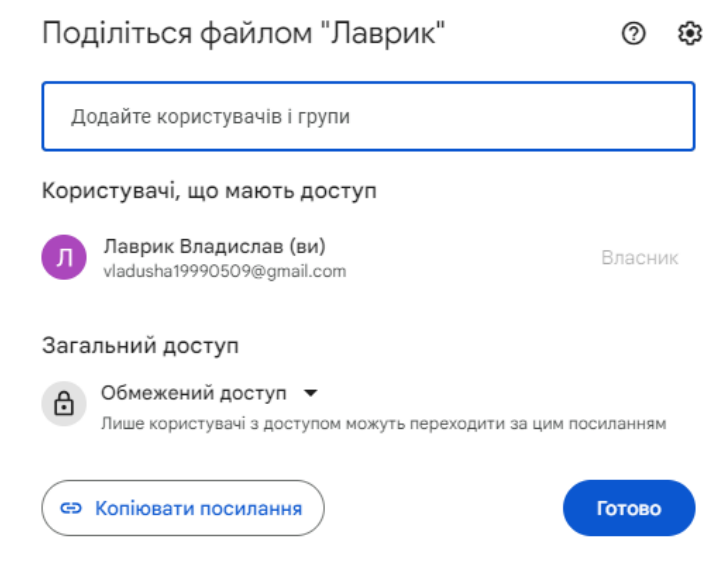


Рисунок 3.6 – Демонстрація налаштувань доступу

Студенти отримують доступ до власних папок та групових папок, в яких можна зберігати матеріали для лабораторних робіт. Викладачі отримують доступ до загальних папок з дисциплінами, де можуть переглядати результати студентів та надавати оцінки.

Після завершення лабораторних завдань у віртуальній лабораторії, студенти можуть завантажувати свої готові проєкти або робочі файли до власних папок. Вони обирають відповідну дисципліну, групу і папку зі своїм іменем для збереження матеріалів.

Цей підхід допомагає організувати і керувати доступом до матеріалів віртуальної лабораторії та забезпечує безпеку даних, дозволяючи студентам та викладачам працювати зі своїми файлами відповідно до власних потреб та завдань.

4 ТЕСТУВАННЯ РОБОТИ ЛАБОРАТОРІЇ

Для перевірки працездатності розробленої системи необхідно провести її тестування у реальних умовах. Тестування відбувалося наступним чином. Спочатку ініціюємо з'єднання з віртуальною лабораторією за допомогою програми TeamViewer. Вводимо ідентифікатор доступу та пароль для підключення до віддаленої машини, яка знаходиться в університеті. В цей час, а у майбутньому – на початку практичної роботи, асистент лабораторії вмикає живлення для комп'ютера та входить в систему. Програма TeamViewer працює у фоновому режимі на комп'ютері, тому більше ніяких маніпуляцій не треба робити з боку асистента. Від користувача потрібно ввести в поле ID машини, а далі натиснути «З'єднатись» і ввести пароль, як це зображено на рисунку 4.1.

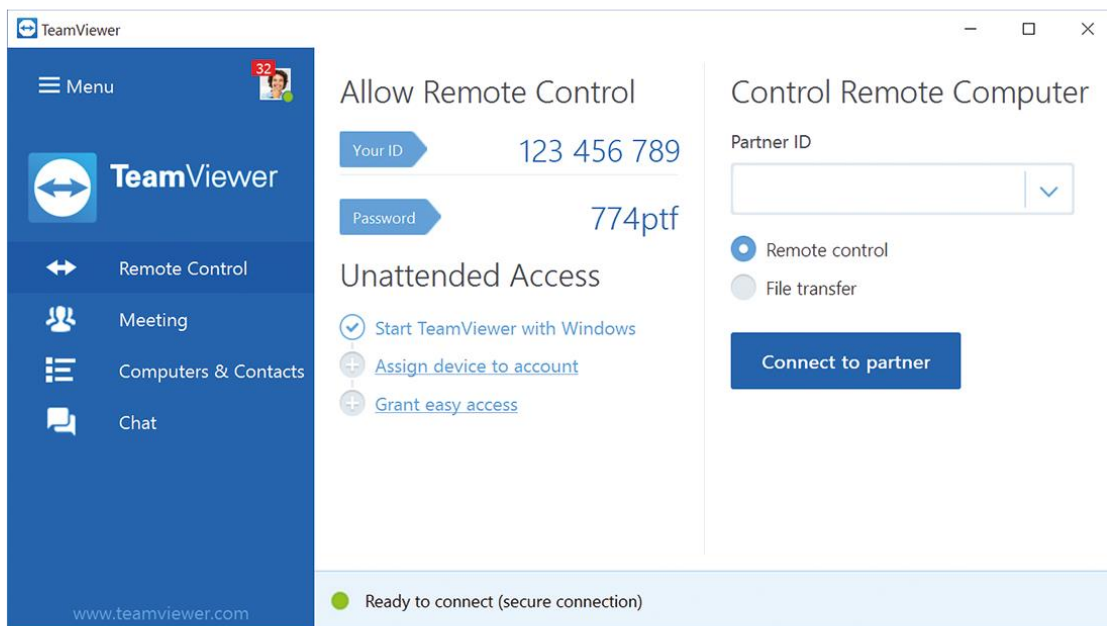


Рисунок 4.1 – Підключення через TeamViewer до віртуальної лабораторії

Після успішного з'єднання відкривається робочий стіл машини, яка знаходиться в університеті, тож з цієї миті можна керувати віртуальною лабораторією. На даному етапі для студента відкривається 2 можливості:

- використовувати віртуальну лабораторію у якості ПК для програмування або для виконання лабораторних робіт, де потрібні високі вимоги до програмного забезпечення.

- це безпосереднє програмування роботизованих пристроїв. На цьому етапі потрібно буде задіяти асистента, для підключення фізичних пристроїв до машини.

У даному проєкті під час підключення iRobot Create 2 до операційної системи Windows 10 самостійно підшукує відповідний драйвер до робота-пилососа та встановлює його автоматично.

iRobot Create 2 під'єднується до ПК за допомогою кабелю адаптеру. Кабель адаптер зображений на рисунку 4.2, який з одного боку підключення до робота, має 7-ми піновий роз'єм, а на іншому кінці кабелю, підключення до ПК, плата-адаптер з виходом на роз'єм USB.



Рисунок 4.2 – 7-піновий адаптер Mini-DIN

Після підключення робота через адаптер та встановлення усіх необхідних драйверів у розділі: Пуск → Панель керування → Обладнання та звук → Пристрої та принтери, можна побачити, якщо все правильно під'єднано і завантажено, пристрій робот-пилосос, це зображено на рисунку 4.3.

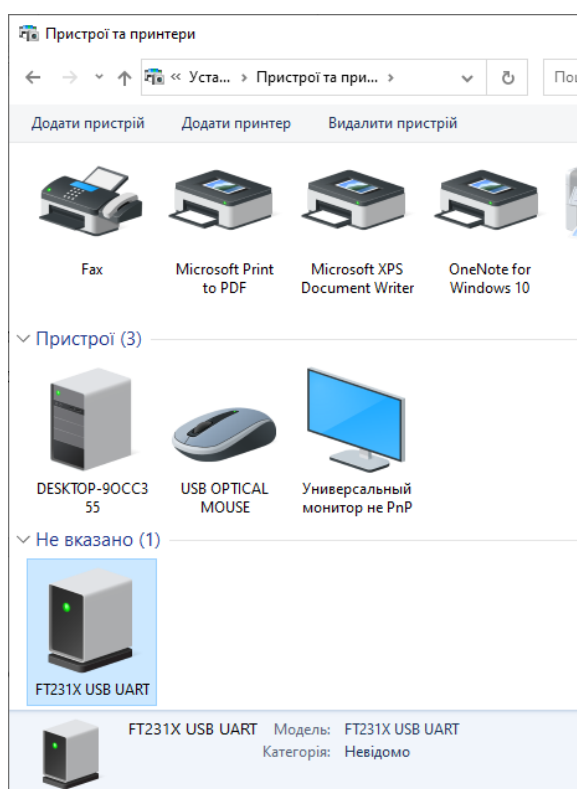


Рисунок 4.3 – Скриншот розділу «Пристрої та принтери»

На даному зображенні рис.4.3, можна спостерігати назву пристрою, а саме: FT231X USB UART.

Після перевірки успішного з'єднання з пристроєм можна починати програмувати iRobot Create 2 за допомогою програмного середовища Visual Studio Code, рис 4.4, яке було обрано у попередньому розділі.

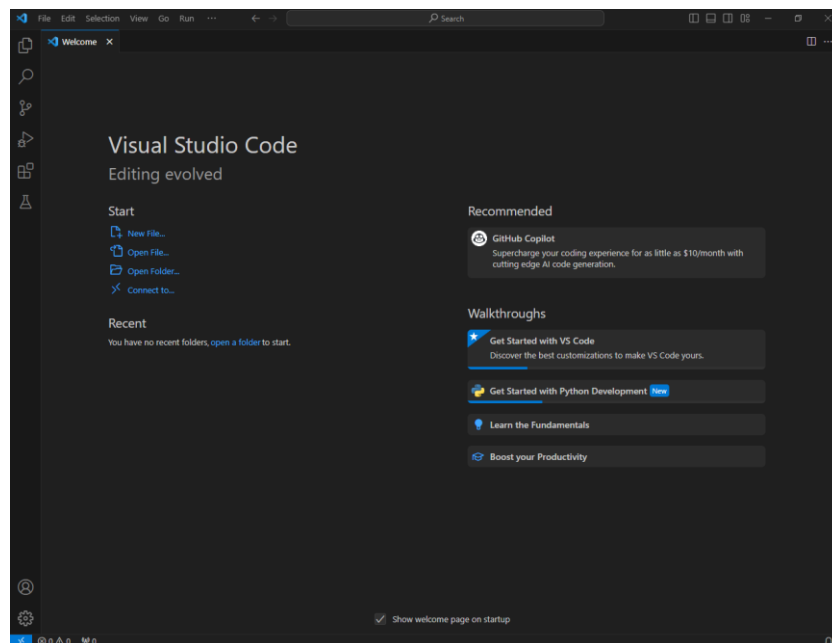


Рисунок 4.4 – Візуалізація програмного середовища Visual Studio Code

Далі необхідно створити файл проєкту натиснувши «New File», рисунок 4.5, та обрати «Text file», натиснувши на обране лівою кнопкою миші або «Enter».

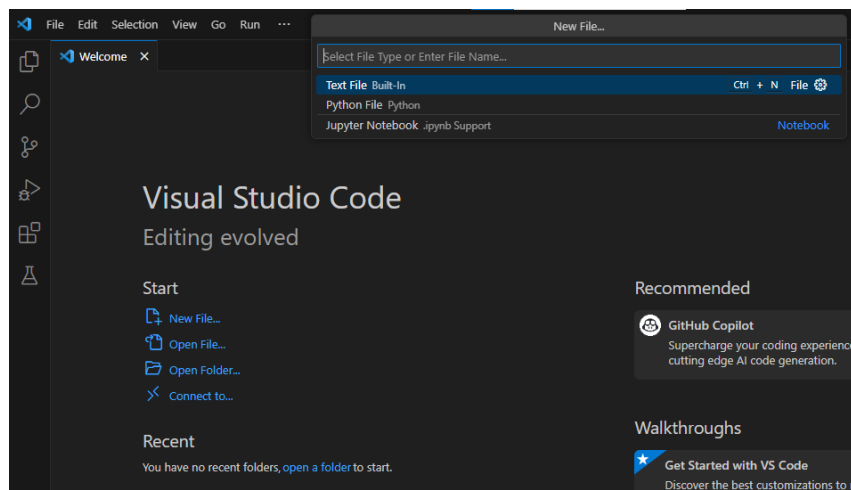


Рисунок 4.5 – Створення проєкту у Visual Studio Code

Після створення нового текстового файлу на рисунку 4.6 потрібно обрати мову проєкту. Натиснувши на текст, який підсвічується синім, а саме «Select a language» випадає список мов, які можна обрати для проєкту, рисунок 4.7. Так як у попередньому розділі було розглянуто декілька мов програмування і було зроблено вибір саме на Python, тому треба проскролити мишкою нижче та обрати саме цю мову, див. рис. 4.7.

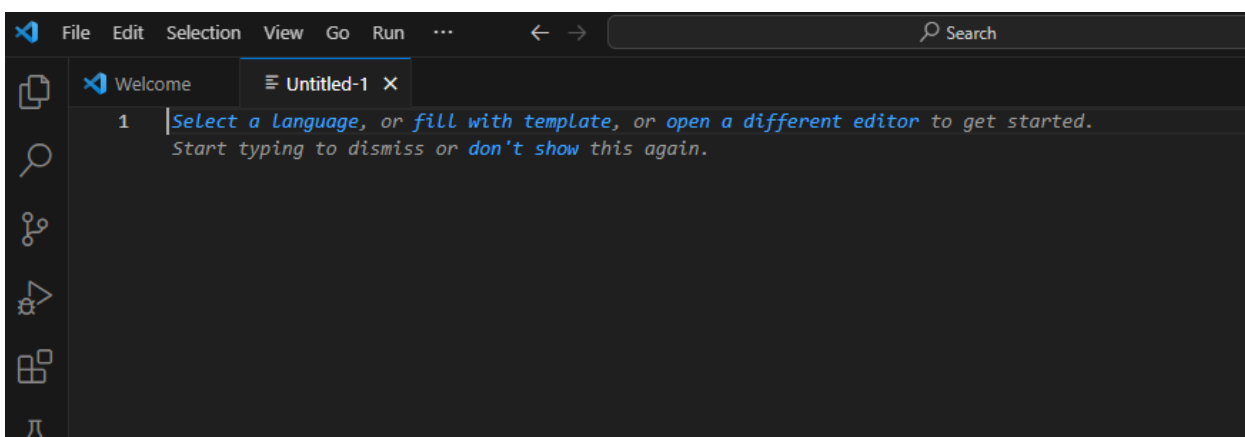


Рисунок 4.6 – Створення проєкту

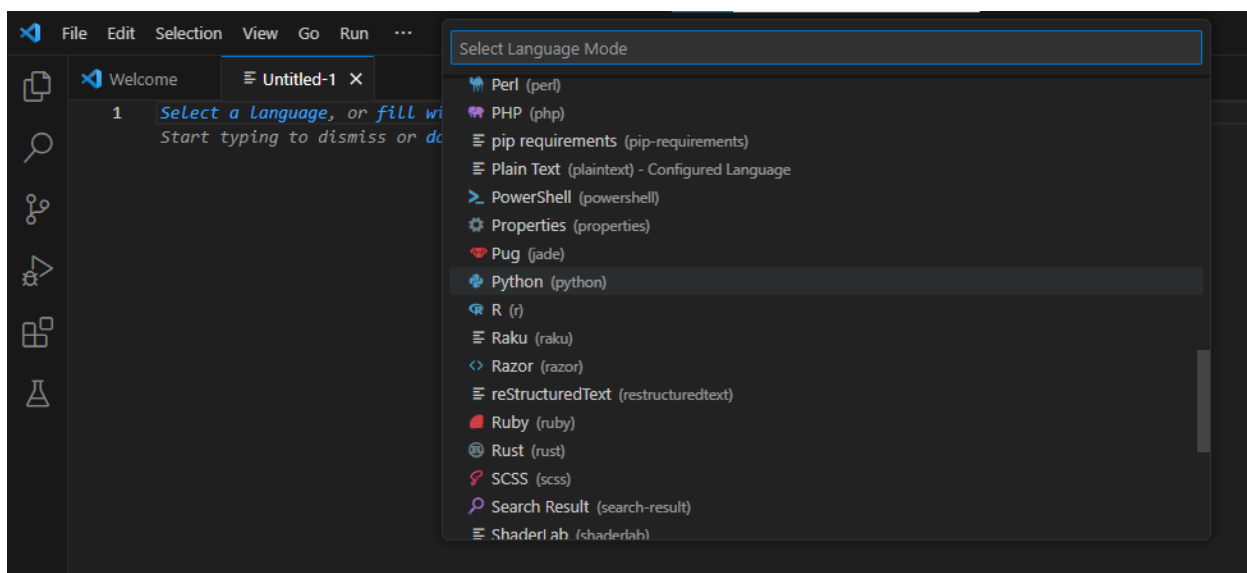


Рисунок 4.7 – Вибір мови програмування

Наступним кроком після створення проєкту його необхідно зберегти. Це можна зробити декількома варіантами:

– File → Save As... → обрати місце на комп'ютері;

– сполучення клавіш «Ctrl+S» → обрати місце на комп’ютері.

У віртуальній лабораторії студент може зберігати його проєкт у власній папці, на прикладі зображено збереження файлу у папку Documents рисунок 4.8.

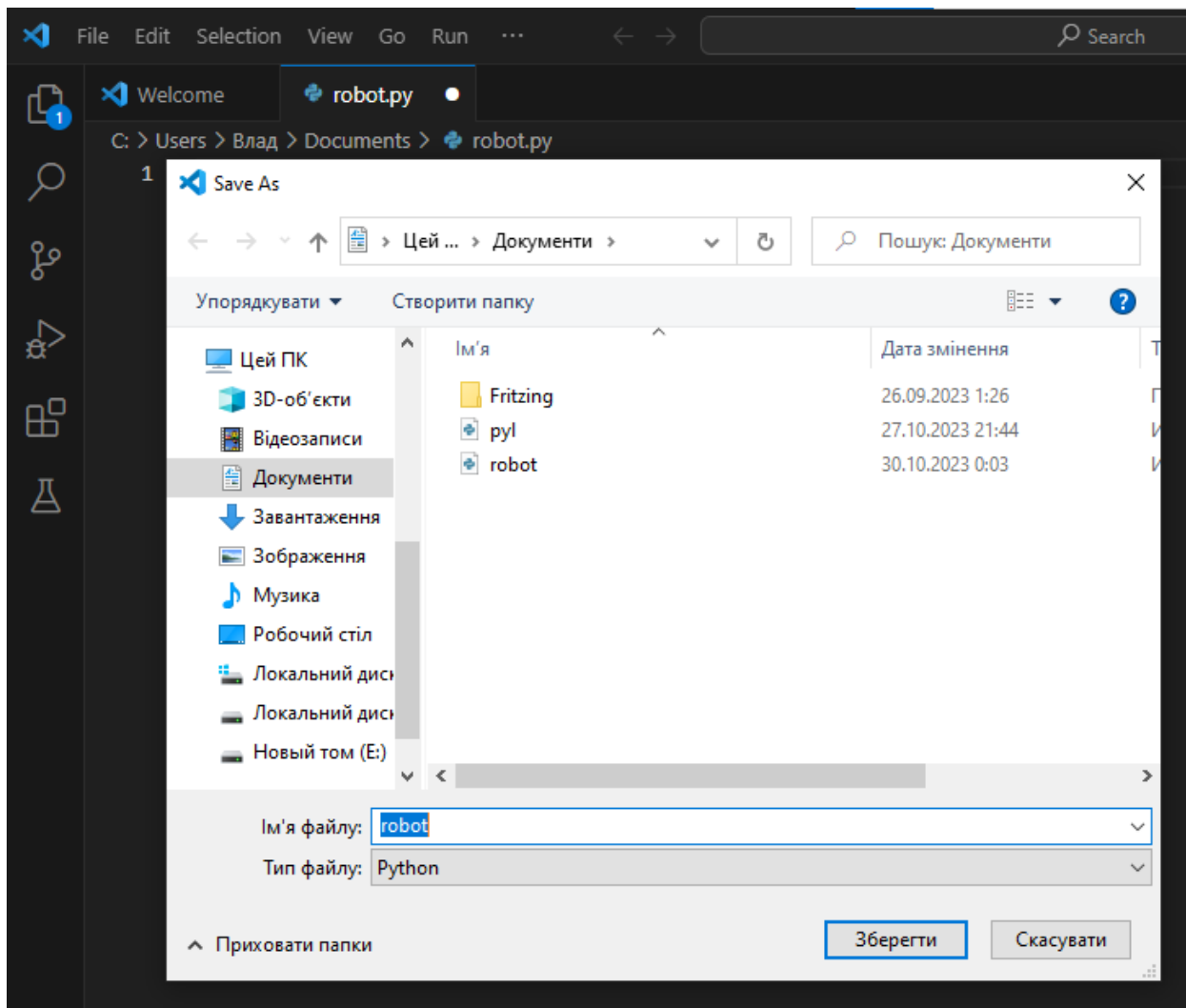
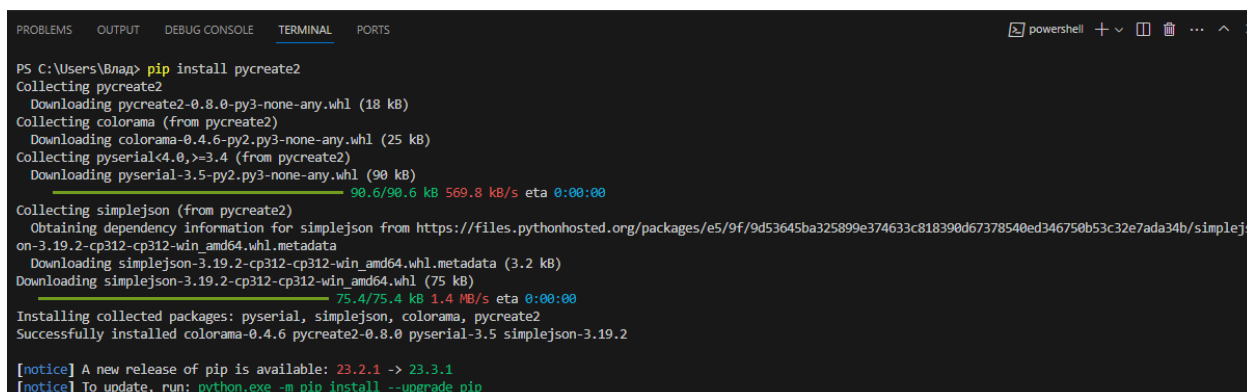


Рисунок 4.8 – Збереження проєкту

Наступним етапом після зберігання файлу треба завантажити бібліотеку для керування iRobot Create 2 за допомогою python. Для того, щоб завантажити цю бібліотеку, необхідно у середовищі проєкту у Visual Studio Code натиснути у верхньому лівому куті на вкладку «View», далі треба натиснути на «Terminal». У нижній частині проєкту відкриється вікно терміналу, де треба вказати команду для завантаження бібліотеки, а саме: `pip install pycreate2`.

Після завантаження та встановлення бібліотеки, рисунок 4.9, можна писати програмний код згідно до завдання.



```

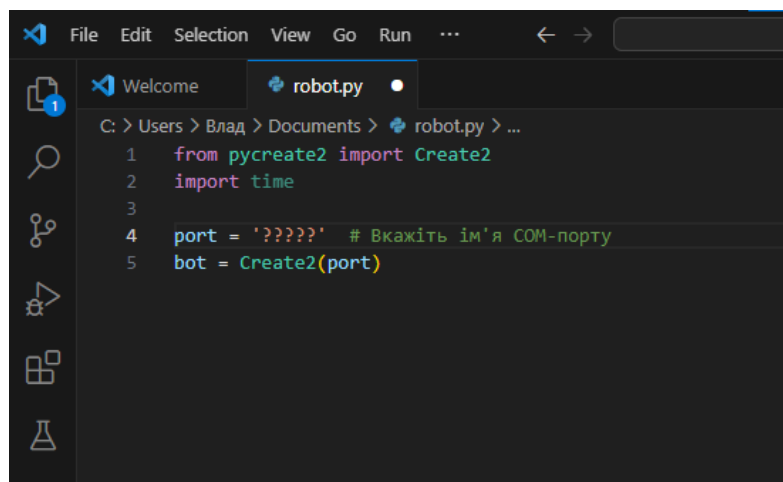
PS C:\Users\Bnag> pip install pycreate2
Collecting pycreate2
  Downloading pycreate2-0.8.0-py3-none-any.whl (18 kB)
Collecting colorama (from pycreate2)
  Downloading colorama-0.4.6-py2.py3-none-any.whl (25 kB)
Collecting pyserial<4.0,>=3.4 (from pycreate2)
  Downloading pyserial-3.5-py2.py3-none-any.whl (90 kB)
    90.6/90.6 kB 569.8 kB/s eta 0:00:00
Collecting simplejson (from pycreate2)
  Obtaining dependency information for simplejson from https://files.pythonhosted.org/packages/e5/9f/9d53645ba325899e374633c818390d67378540ed346750b53c32e7ada34b/simplejs
on-3.19.2-cp312-cp312-win_amd64.whl.metadata
  Downloading simplejson-3.19.2-cp312-cp312-win_amd64.whl.metadata (3.2 kB)
  Downloading simplejson-3.19.2-cp312-cp312-win_amd64.whl (75 kB)
    75.4/75.4 kB 1.4 MB/s eta 0:00:00
Installing collected packages: pyserial, simplejson, colorama, pycreate2
Successfully installed colorama-0.4.6 pycreate2-0.8.0 pyserial-3.5 simplejson-3.19.2

[notice] A new release of pip is available: 23.2.1 -> 23.3.1
[notice] To update, run: python.exe -m pip install --upgrade pip

```

Рисунок 4.9 – Завантаження бібліотеки для керування iRobot Create 2

У процесі програмування необхідно написати порт комп'ютера, до якого під'єднаний робот-пилосос, і на цьому кроці постає питання, де його знайти, рисунок 4.10.



```

1 from pycreate2 import Create2
2 import time
3
4 port = '?????' # Вкажіть ім'я COM-порту
5 bot = Create2(port)

```

Рисунок 4.10 – Код програми

Для того, щоб дізнатися який порт підключення має робот-пилосос, потрібно звернутися до розділу «Диспетчер пристроїв», див. рис. 4.11, це можна зробити декількома варіантами, а саме:

- Пуск → Параметри → Про систему → Диспетчер пристроїв;
- або сполучення клавіш «Win + X» → Диспетчер пристроїв.

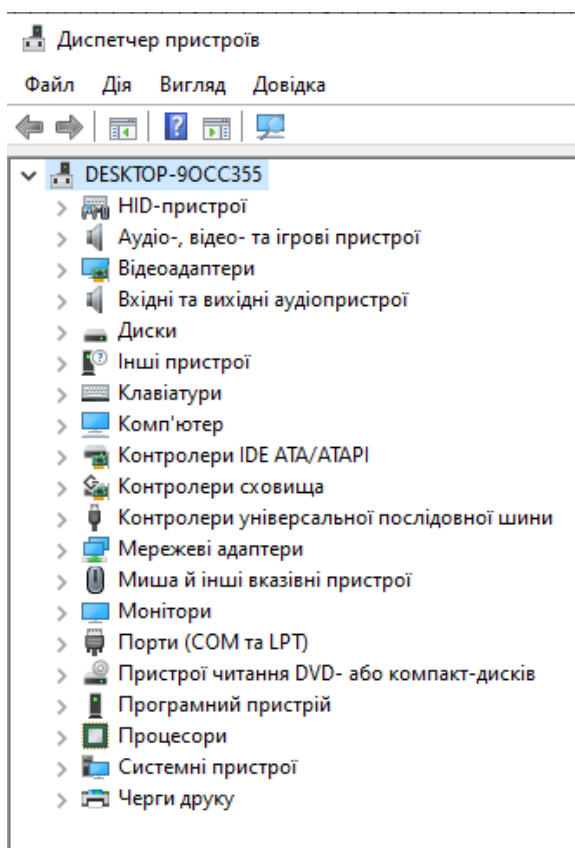


Рисунок 4.11 – Диспетчер пристроїв

У розділі «Диспетчер пристроїв» потрібно розкрити розділ «Порти», де можна отримати інформацію щодо того, який порт комп'ютера забезпечує підключення до роботи. У даному випадку порт відповідає COM3, повна назва порту USB Serial Port (COM3), рисунок 4.12.

Таким чином, дізнавшись назву порту можна далі продовжити написання програмного коду, який зображено на рисунку 4.10.

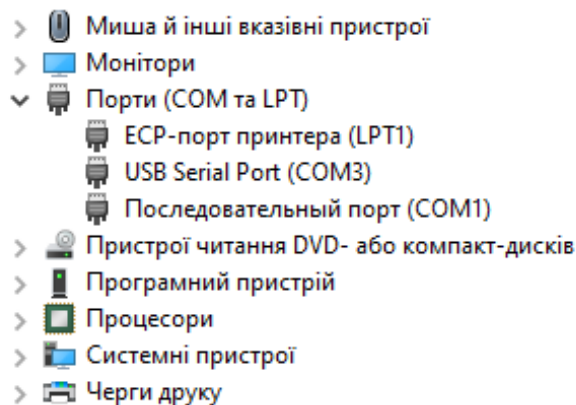


Рисунок 4.12 – Диспетчер пристроїв

Після написання програмного коду до проєкту, перед завантаженням на робот-пилосос його необхідно зберегти. Збереження можна виконати за допомогою сполучення клавіш «Ctrl+S». Далі, щоб завантажити програму на iRobot Create 2, необхідно відкрити «Командний рядок». Його можна визвати з панелі пошуку, написати ввівши «Командний рядок» або «cmd», рисунок 4.13.

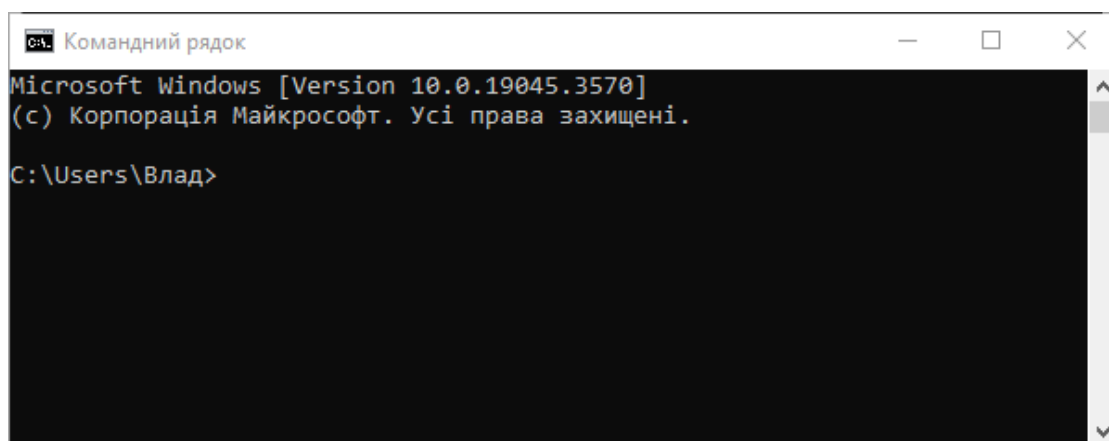


Рисунок 4.13 – Командний рядок

Так як проєкт зберігся у папці «Документи», треба перейти до неї, написавши в полі вводу команду «cd Documents». Після переходу до обраної директорії, треба ввести назву проєкту у поле вводу, проєкт був збережений за назвою: python robot.py, тому вводиться ця команда на виконання, та на початку команди пишеться ідентифікатор мови запуску. Для підтвердження дії треба натиснути «Enter», див.рис.4.14.

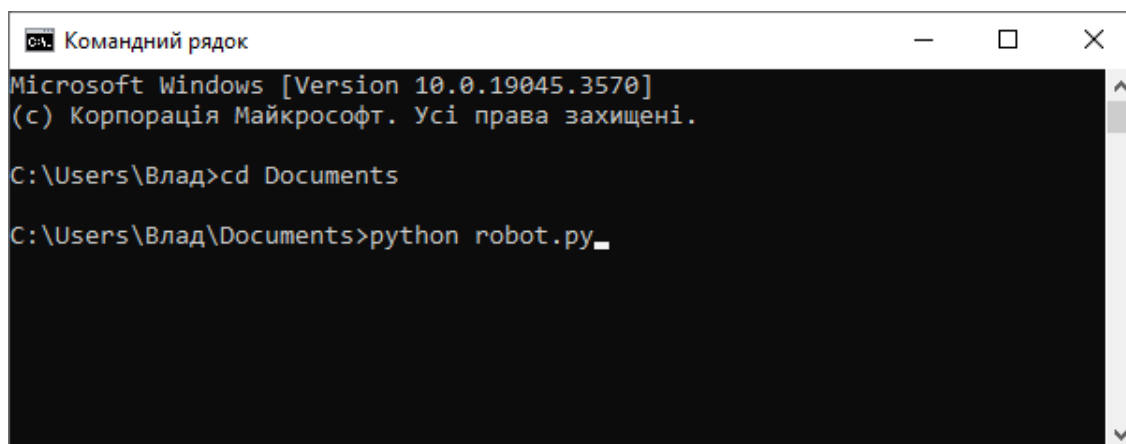


Рисунок 4.14 – Командний рядок з командами

Після виконання програми робот починає виконувати певні умови, які були запрограмовані у файлі коду. Але, щоб спостерігати за рухом робота-пилососа у реальному часі, треба під'єднати до ПК в університеті вебкамеру. Після успішного під'єднання вебкамери до ПК, під час віддаленого доступу через програму TeamViewer, необхідно розгорнути панель керування у правому нижньому куті, рисунок 4.15, та ввімкнути відео – що дає можливість спостерігати за роботом у реальному часі на будь-якій відстані.

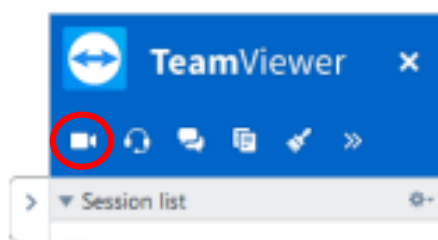


Рисунок 4.15 – Панель керування TeamViewer

Коли студент програмує робота, виконуючи певні задачі, через віддалений доступ, а саме у віртуальній лабораторії на програмному забезпеченні університету, в реальності, це виглядає наступним чином рисунки 4.16 – 4.18.



Рисунок 4.16 – Вигляд процесу програмування iRobot Create 2

Після завантаження програмного коду на робот, пристрій починає виконувати певні умови, які закладені у програмному коді. Вигляд з вебкамери під час руху пілососа зображено на рисунках 4.17 та 4.18 з різних ракурсів.



Рисунок 4.17 – Процес виконання програмного коду



Рисунок 4.18 – Процес виконання програмного коду

Після виконання робіт зі створення проєкту та тестування програмного коду на пристрої, студент може зберегти свій проєкт у папку на комп'ютері, або одразу відправити проєкт до хмарного сховища Google Disk. Це можна зробити відкривши браузер, наприклад буде використовуватися Chrome, на стартовій сторінці потрібно в правому кутку натиснути 9-ти крапки, де вже буде відкрито адміністративний обліковий запис, рисунок 4.19.

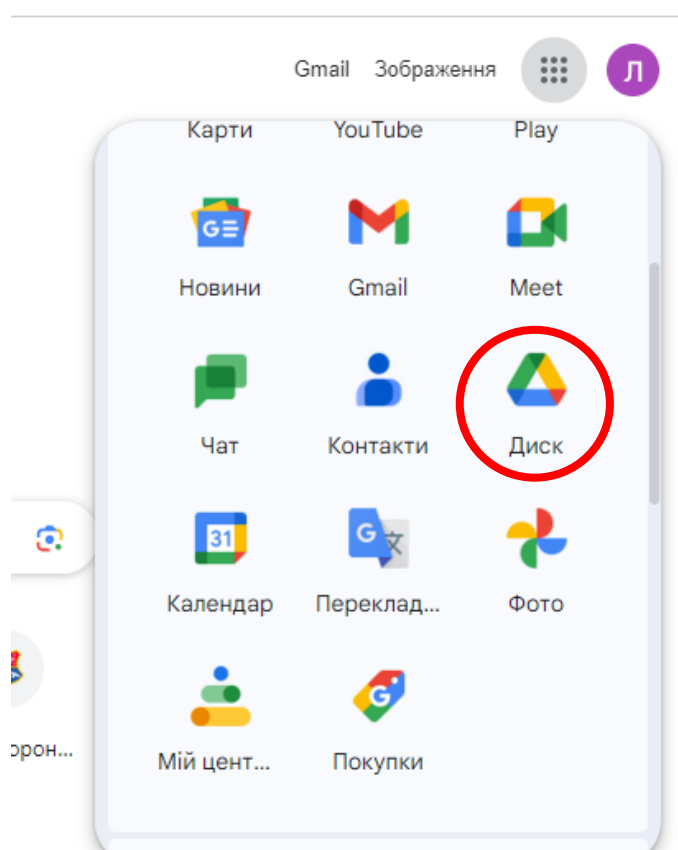


Рисунок 4.19 – Перелік додатків у браузері Chrome

Студенту необхідно перейти у додаток Google Disk, де він обирає папку з предметом дисципліни, далі папку з групою та вже папку зі своїм ім'ям, зображено на рисунку 4.20. Завантажуючи на хмару, студент може скористатися цим проєктом для модернізації в майбутньому, а викладач може оцінити навички студента, на скільки точно була виконана робота за всіма критеріями. Хмарне середовище зберігає всю інформацію та надає доступ у будь-який час при будь-яких обставинах.

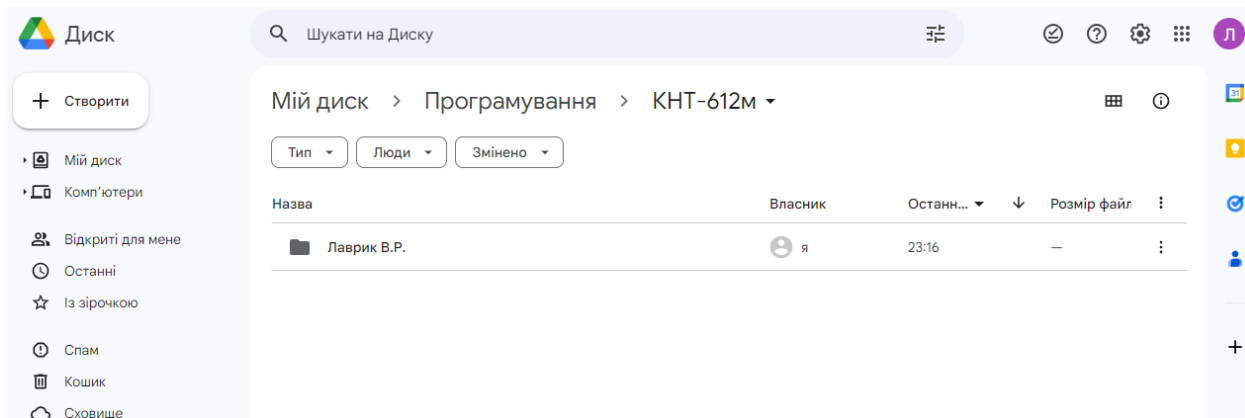


Рисунок 4.20 – Вигляд хмарного середовища Google Disk

Таким чином, проведене тестування довело, що система є працездатною і повністю виконує покладені на неї функції.

ВИСНОВКИ

Під час роботи в даному дипломному проєкті було проведено, оцінено, розроблено та продемонстровано доцільність створення віртуальної лабораторії на базі аудиторії 52 навчальної лабораторії «Системної інтеграції комп'ютерних технологій» кафедри комп'ютерних систем та мереж НУ «Запорізька політехніка».

При проведенні дослідження було визначено виклики сучасності і вимоги до розроблюваного проєкту. А саме було враховано, що є студенти, які знаходяться на окупованих територіях. Для них особливо важливе, щоб підключення до віртуальної лабораторії було захищеним, оскільки їхні дані можуть бути в небезпеці. Також оскільки існує проблема з доступом до інтернету на окупованих територіях, тому важливо, щоб передача трафіку була максимально ефективною і не заважала студентам навчатися.

Під час проведення аналізу наявної інфраструктури було визначено, що на кафедрі є всі необхідні компоненти для створення віртуальної лабораторії. Існує електричне живлення будівлі, стабільне підключення до мережі інтернет, а також є оновлені персональні комп'ютери, які відповідають мінімальним вимогам до необхідного програмного забезпечення. Також було проаналізовано програмне забезпечення, необхідне для реалізації віртуальної лабораторії, а саме організації віддаленого доступу. Із розглянутих систем віддаленого доступу, таких як: TeamViewer, віддалений робочий стіл Chrome, Anydesk, вибір було зроблено на користь TeamViewer, бо це ПЗ є більш популярним та знайомим по функціоналу більшості користувачів сьогодення, використовує технологію шифрування даних, аналогічну з шифруванням банківських операцій, що повністю забезпечує цілісність даних користувача та забезпечує безперервну комунікацію з віддаленою віртуальною лабораторією, навіть при низькій швидкості інтернету.

Наступним кроком було розроблено практичні завдання, для яких доцільно використовувати віртуальну лабораторію: завдання із використанням спеціалізованого апаратного забезпечення, якого скоріше усього немає в наявності

у студента, і завдання, які потребують значних обчислювальних ресурсів, які теж можуть бути недоступними для студента. Як спеціалізоване обладнання було обрано роботу із роботом-пилососом iRobot Create 2 Roomba 600. Другим типом завдання було створення доповненої реальності за допомогою програмного забезпечення Unity3D. Окрім розроблених завдань також надано детальний опис їх розв'язку.

Наступним кроком було спроектовано віртуальну лабораторію та налаштовано її роботу з використанням обраних програмних та апаратних засобів.

У дипломному проєкті було доведено працездатність розробленої віртуальної лабораторії шляхом перевірки виконання розроблених завдань у межах університету та з віддаленим доступом до лабораторії [23].

Таким чином, важливо зазначити, що робота над створенням віртуальної лабораторії вимагає збалансованого підходу, урахування потреб користувачів та актуальних обставин. Цей проєкт став кроком уперед в забезпеченні доступу до освіти в умовах війни і обмежень на фізичному рівні, а також у впровадженні сучасних технологій для покращення навчального процесу магістрів на кафедрі комп'ютерних систем та мереж (Додаток Г) зі спеціальності 123 Комп'ютерна інженерія.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Лаврик В.Р. Аналіз стратегій ефективного просування Instagram-магазину для успішного бізнесу / В.Р. Лаврик, М.Ю. Тягунова // Системи та технології. – № 1 (65). – 2023. – ISSN 2521-6643. – С. 46-52. <https://doi.org/10.32782/2521-6643-2023.1-65.6>
2. Програма TeamViewer. URL: <https://www.teamviewer.com>
3. Програма Віддалений робочий стіл Chrome. URL: <https://remotedesktop.google.com/access/?hl=ua>
4. Програма AnyDesk. URL: <https://anydesk.com>
5. Доповнена реальність. URL: <https://intboard.ua/pres-sluzhba/blog/ar/>
6. Unity Hub. URL: <https://unity.com/>
7. Vuforia. URL: <https://developer.vuforia.com/>
8. Тягунова М.Ю. Методичні вказівки до лабораторних робіт з дисципліни «Інтегровані комп'ютерні системи» для студентів спеціальності 123 «Комп'ютерна інженерія» за освітньою програмою «Спеціалізовані комп'ютерні системи» усіх форм навчання. Частина II. – Запоріжжя: НУ «Запорізька політехніка», 2023. – 44 с.
9. Tiahunova M., Kyrychek H., Bohatyrova T., Moshynets D. System and method of automatic collection of objects in the room. Proceedings of the 4rd Workshop for Young Scientists in Computer Science & Software Engineering, CS&SE@SW 2021, Vol-3077, Kryvyi Rih, Ukraine, December 15, 2021, P.174-186.
10. Getting Started with Create 2. URL: <https://edu.irobot.com/learning-library/getting-started-with-create-2>
11. Instructions. Robotic vacuum cleaners iRobot Roomba 600 series. 2012. c32
12. Specification based on the iRobot Roomba 600. URL: <https://www.irobot.com/>
13. Moshynets D., Tiahunova M., Kyrychek H. Research of the robotic platform iRobot Create 2. Proceedings of the Dortmund IRC 2022: Dortmund International Research Conference 2022, IRC 2022, Dortmund, Germany, July 1-2, 2022, P. 272-273.
14. ROS - Robot Operating System. URL: <https://robocraft.ua/robosoft/721>

15. Visual Studio Code. URL: <https://code.visualstudio.com/>
16. Програмне середовище Visual Studio Code. URL: <https://visualstudio.microsoft.com/>
17. Посібник з мови програмування C++. URL: <https://metanit.com/cpp/tutorial/>
18. Python. URL: <https://www.python.org/>
19. Програмування Roomba. URL: <http://www.robotappstore.com>
20. Хмарний сервіс Google Drive. URL: https://www.google.com/intl/ru_uA/drive/
21. Хмарний сервіс OneDrive. URL: <https://support.microsoft.com/>
22. Хмарний сервіс Dropbox. URL: <https://www.dropbox.com/>
23. Тягунова М.Ю. Віртуальна лабораторія як ефективне рішення під час дистанційної освіти/ М.Ю. Тягунова, В.Р. Лаврик // Системи та технології. – № 2 (66). – 2023. – ISSN 2521-6643. – у друці.

ДОДАТОК А

```

from pycreate2 import Create2
import time

# Create a Create2.
port = 'COM3' # where is your serial port?
bot = Create2(port)

# Start the Create 2
bot.start()

# Put the Create2 into 'safe' mode so we can drive it
# This will still provide some protection
bot.safe()

# You are responsible for handling issues, no protection/safety in
# this mode ... becareful
bot.full()

# directly set the motor speeds ... move
bot.drive_direct(-100, -100)
time.sleep(2)

# turn in place
bot.drive_direct(150, -250) # inputs for motors are +/- 500 max
time.sleep(2)

bot.drive_direct(200, 200) # inputs for motors are +/- 500 max
time.sleep(2)

# turn in place
bot.drive_direct(-100, 280) # inputs for motors are +/- 500 max
time.sleep(1)

bot.drive_direct(200, 200) # inputs for motors are +/- 500 max
time.sleep(2)

bot.drive_direct(-100, 280) # inputs for motors are +/- 500 max
time.sleep(1)

bot.drive_direct(200, 200) # inputs for motors are +/- 500 max
time.sleep(2)

bot.drive_direct(-100, 280) # inputs for motors are +/- 500 max
time.sleep(1)

bot.drive_direct(200, 200) # inputs for motors are +/- 500 max
time.sleep(2)

bot.drive_direct(280, -100) # inputs for motors are +/- 500 max
time.sleep(1)

bot.drive_direct(100, 100) # inputs for motors are +/- 500 max
time.sleep(2)

# Stop the bot
bot.drive_stop()

# query some sensors
sensors = bot.get_sensors() # returns all data
print(sensors.light_bumper_left)

# Close the connection
bot.close()

```

Рисунок А.1 – Скриншот лістингу коду програми до завдання 1

ДОДАТОК Б

```

from pycreate2 import Create2
import time

def backup_and_turn(bot):
    # Implement backward movement and rotation here
    bot.drive_direct(-100, -100) # Drive backwards
    time.sleep(2)
    bot.drive_stop()

    # Rotation (example for right)
    bot.drive_direct(200, -200)
    time.sleep(1)
    bot.drive_stop()

def analyze_sensors(bot):
    # Analysis of sensors every second
    while True:
        sensors = bot.get_sensors()

        if sensors.bumps_wheeldrops.bump_left or
sensors.bumps_wheeldrops.bump_right:
            # Bumper sensor
            backup_and_turn(bot)

        if sensors.cliff_left:
            # Left cliff
            backup_and_turn(bot)

        if sensors.cliff_right:
            # Right cliff
            backup_and_turn(bot)

        # Moving forward
        bot.drive_direct(100, 100)
        time.sleep(1)
        bot.drive_stop()

if __name__ == "__main__":
    port = 'COM3'
    bot = Create2(port)

    try:
        bot.start()
        bot.safe()
        analyze_sensors(bot)
    finally:
        bot.drive_stop()
bot.close()

```

Рисунок Б.1 – Скриншот лістингу коду програми до завдання 2

ДОДАТОК В

```

from pycreate2 import Create2
import serial
import time

def send_command(command):
    ser.write(bytes(command))
    time.sleep(0.1)

# Open a serial connection to the Roomba
port = 'COM3' # Replace with the correct COM port
ser = serial.Serial(port, baudrate=115200)

# Start the Roomba
send_command([128, 131])

# Define the first song (Jingle Bells)
send_command([140, 0, 16, 76, 16, 76, 16, 76, 32, 76, 16, 76, 16, 76,
32, 76, 16, 79, 16,
                72, 16, 74, 16, 76, 32, 77, 16, 77, 16, 77, 16, 77, 32,
77, 16])

# Play the first song
send_command([141, 0])

# Wait for the first song to finish
time.sleep(5)

# Define the second song
send_command([140, 1, 7, 76, 16, 76, 32, 79, 16, 79, 16, 77, 16, 74,
16, 72, 32])

# Play the second song
send_command([141, 1])

# Close the serial connection
ser.close()

```

Рисунок В.1 - Скриншот лістингу коду програми до завдання 3

ДОДАТОК Г



Вадим ШАЛОМЄЄВ
(ім'я, прізвище)

р.

АКТ

впровадження результатів НДР

1 грудня 2023 р.

м. Запоріжжя

Складено комісією у складі:

Голова Равіль КУДЕРМЕТОВ
(ім'я, прізвище)

члени комісії Марія ТЯГУНОВА, Олега ПОЛЬСЬКА,
(ім'я, прізвище)
Наталія ЦЕРБАК

Комісія провела роботу по визначенню фактичного впровадження дипломного проекту «Віртуальна комп'ютерна лабораторія кафедри комп'ютерних систем та мереж», виконаної у НУ «Запорізька політехніка» студентом ФКНТ, гр. КНТ-612м, Лавриком Владиславом, науковий керівник – доцент каф. КСМ Тягунова М.Ю., згідно тематичного плану держбюджетної теми 04811 «Дослідження, розробка та удосконалення методів, моделей та підсистем комп'ютерних систем, зокрема, інтелектуальних та кіберфізичних систем», згідно тематичного плану кафедральних держбюджетних науково-дослідних робіт на 2021-2024 навчальні роки, та установила, що результати вказаної роботи впроваджено як складову засобів навчання під час здобуття освіти в дистанційному режимі та безпосередньо в учбовий процес НУ «Запорізька політехніка» на кафедрі комп'ютерних систем та мереж.

Вид та об'єм впровадження: створено віртуальну комп'ютерну лабораторію на базі ауд. 52 кафедри комп'ютерних систем та мереж, запропоновані завдання, які розроблено в межах дипломного проєкту, впроваджено до лабораторних робіт студентів магістратури з дисциплін «Інтегровані комп'ютерні системи» та «Кіберфізичні системи» за спеціальністю 123 Комп'ютерна інженерія.

Досягнені технічні результати, їх рівень: підвищено якість підготовки фахівців з відповідної спеціальності.

Річний економічний ефект від впровадження складає (тис. грн.) _____

Складено у 2 прим.:

1-й прим. у патентно-інформаційний відділ НУ «Запорізька політехніка»,

2-й прим. виконавцю.

Голова комісії

Члени комісії

Равіль КУДЕРМЕТОВ
підпис

Равіль КУДЕРМЕТОВ
(ім'я, прізвище)

Марія ТЯГУНОВА
підпис

Марія ТЯГУНОВА
(ім'я, прізвище)

Олега ПОЛЬСЬКА
підпис

Олега ПОЛЬСЬКА
(ім'я, прізвище)

Наталія ЦЕРБАК
підпис

Наталія ЦЕРБАК
(ім'я, прізвище)