МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ПОЛІСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет інформаційних технологій,
обліку та фінансів
Кафедра комп'ютерних технологій
і моделювання систем

Кваліфікаційна робота
на правах рукопису

ПЕНЬКІВСЬКИЙ ДЕНИС МИКОЛАЙОВИЧ

УДК 004:378.147

КВАЛІФІКАЦІЙНА РОБОТА

Інформаційна система забезпечення дистанційного навчання

126 «Інформаційні системи та технології»

Подається на здобуття освітнього ступеня бакалавр

кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ Денис ПЕНЬКІВСЬКИЙ

Керівник роботи:
Лапін Андрій Валерійович,
кандидат технічних наук

Житомир - 2024

Висновок кафедри _____
за результатами попереднього захисту: _____

Протокол засідання кафедри _____
№ ____ від «_____» _____ 20____ р.

Завідувач кафедри _____

_____ _____ _____
(науковий ступінь, вчене звання) (підпис) (прізвище, ім'я, по батькові)
«_____» _____ 20____ р.

Результати захисту кваліфікаційної роботи

Здобувач вищої освіти _____ захистив (ла)
(прізвище ,ім'я, по батькові)
кваліфікаційну роботу з оцінкою:

сума балів за 100-бальною шкалою _____
за шкалою ECTS _____
за національною шкалою _____

Секретар ЕК

_____ _____ _____
(науковий ступінь, вчене звання) (підпис) (прізвище, ім'я, по батькові)

АНОТАЦІЯ

Пеньківський Д. М. Інформаційна система забезпечення дистанційного навчання. – *Кваліфікаційна робота на правах рукопису.*

Кваліфікаційна робота на здобуття освітнього ступеня бакалавра за спеціальністю 126 – Інформаційні системи та технології. – Поліський національний університет, Житомир, 2023.

Кваліфікаційна робота присвячена розробці та аналізу інформаційної системи для забезпечення дистанційного навчання. У роботі розглядаються актуальні аспекти використання інформаційних технологій у навчальному процесі, особливості дистанційного навчання та вимоги до інформаційних систем, які забезпечують цей процес.

Розроблена інформаційна система включає в себе модулі для взаємодії між викладачами та студентами. В роботі детально описані функціональні можливості системи, її архітектура та інтерфейс.

Отже, кваліфікаційна робота відображає актуальність та важливість використання інформаційних систем для організації та забезпечення дистанційного навчання у сфері інформаційних технологій.

Ключові слова: інформаційна система, дистанційне навчання, платформа, Discord.

SUMMARY

Pienkovskyi, D. M. Information System for Distance Learning Provision. – Qualification work as a manuscript.

Qualification work for the degree of Bachelor in the specialty 126 – Information Systems and Technologies. – Polissia National University, Zhytomyr, 2023.

The qualification work is dedicated to the development and analysis of an information system for distance learning provision. The paper discusses the relevant aspects of using information technologies in the educational process, the specifics of distance learning, and the requirements for information systems that support this process.

The developed information system includes modules for interaction between teachers and students. The paper provides a detailed description of the system's functional capabilities, its architecture, and interface.

Thus, the qualification work reflects the relevance and importance of using information systems for organizing and providing distance learning in the field of information technologies.

Keywords: information system, distance learning, platform, Discord.

ЗМІСТ

ВСТУП

Актуальність теми. В сучасному інформаційному суспільстві зростає значення та популярність дистанційного навчання як ефективного засобу освіти.

За останні десятиліття цей метод отримав широке визнання як серед студентів, так і серед освітніх установ. Це пояснюється розвитком інформаційних технологій та змінами в суспільних потребах. Однак, разом з ростом популярності дистанційного навчання виникають нові вимоги до систем, які його забезпечують. У цьому контексті стає актуальною проблема створення та оптимізації інформаційних систем, які забезпечують дистанційне навчання, зокрема, їх адаптація до потреб сучасного користувача та вдосконалення процесу навчання.

Метою даної роботи є розробка та впровадження нової інформаційної системи, яка відповідатиме потребам сучасного користувача та сприятиме підвищенню ефективності дистанційного навчання.

Предметом дослідження є процес створення та впровадження інформаційної системи забезпечення дистанційного навчання.

Об'єктом дослідження є самі системи дистанційного навчання, їхні функціональні можливості та характеристики.

Завданням на кваліфікаційну роботу є:

1. Аналіз існуючих систем дистанційного навчання.
2. Визначення основних вимог до нової інформаційної системи.
3. Розробка концепції нової інформаційної системи.
4. Реалізація та тестування розробленої системи.
5. Оцінка ефективності та користування новою системою дистанційного навчання.

В роботі було використано методи об’єктно-орієнтованого проектування, об’єктно-орієнтованого програмування.

Розроблений додатковий додаток може використовуватись для проведення дистанційного навчання як у школах, так і в університетах. Робота над додатковим програмним забезпеченням буде продовжуватись для додавання

нового функціоналу, зміні у роботі при необхідності та при зміні використання основного сервісу.

РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РЕАЛІЗАЦІЇ БОТА ДЛЯ ДИСТАНЦІЙНОГО НАВЧАННЯ

1.1. Аналіз інформаційних потреб і визначення предметної області

Дистанційне навчання стало надзвичайно актуальним у контексті світової пандемії COVID-19, коли школи, університети та навчальні заклади по всьому світу змушені були переглянути свої методи навчання та перейти до дистанційного формату. За цих умов дистанційне навчання стало необхідним для збереження безпеки учнів та персоналу, а також з метою забезпечення неперервності навчального процесу. Однак, варто зазначити, що навіть перед пандемією дистанційне навчання поступово набирало популярності через свою гнучкість та доступність.

Дистанційне навчання виявилося особливо корисним для тих студентів, які знаходяться в важкодоступних регіонах або мають обмежені можливості для фізичного перебування в навчальних закладах. Навчання з використанням онлайн-платформ дозволяє їм отримувати якісну освіту, не виходячи з дому.

Також в контексті глобальних технологічних змін та цифрової трансформації, дистанційне навчання виявляється не лише необхідним зараз, а й перспективним у майбутньому. Застосування сучасних технологій у навчальному процесі дозволяє зробити освіту більш доступною, ефективною та інтерактивною[2]. Learning Management Systems (LMS) - це програмне забезпечення, яке використовується для організації, управління та виконання навчальних курсів або програм. Воно дозволяє вчителям створювати онлайн-курси, завдання, відстежувати прогрес учнів, спілкуватися з ними та надавати доступ до різноманітного навчального контенту, такого як відео, тексти, тести тощо. Для студентів LMS забезпечує можливість доступу до навчальних матеріалів, виконання завдань, спілкування з викладачами та студентами, а також

відстеження їх прогресу у навчанні. Таким чином, дистанційне навчання відкриває нові можливості для розвитку освіти у цифрову епоху.

З огляду на велику кількість доступних платформ для дистанційного навчання, важливо провести аналіз технічних рішень, які вони пропонують.

При аналізі відомих платформ для дистанційного навчання варто врахувати різноманітні аспекти, такі як функціональність, доступність, легкість використання та вартість. Ось кілька ключових платформ та їх особливості:

Moodle відомий своєю доступністю та широким застосуванням серед учнів і педагогів. Це безкоштовна та відкрита система управління навчанням. Розроблений на основі педагогічних принципів, Moodle використовується для різних форм навчання, включаючи змішане навчання, дистанційну освіту та перегорнуті класи. Завдяки своїм розширеним функціям управління, які можна налаштовувати відповідно до потреб навчання. Важливою перевагою Moodle є можливість розширення та адаптації навчального середовища за допомогою плагінів, розроблених спільнотою користувачів. Він надає широкий спектр інструментів, таких як форуми, завдання, онлайн-тести, що дозволяє викладачам створювати різноманітні курси. Також Moodle має велику спільноту користувачів, яка надає активну підтримку та безліч розширень для різних освітніх потреб. Проте, для успішного використання Moodle може знадобитися певний рівень технічних знань, адже налаштування та підтримка платформи може виявитися складними для новачків. Також інтерфейс Moodle може здаватися складним та заплутаним для деяких користувачів.

Blackboard Learn - інша популярна платформа, яка використовується у багатьох навчальних закладах. Вона має розширений функціонал, включаючи інтеграцію з іншими системами та інструменти для відстеження прогресу студентів. Проте, високі витрати на ліцензії можуть стати обмеженням для деяких організацій.

Google Classroom пропонує зручну інтеграцію з іншими Google сервісами, такими як Google Drive та Google Docs. Його простий і зрозумілий інтерфейс полегшує використання. Проте, вона має обмежений функціонал порівняно з іншими платформами.

Canvas LMS відомий своєю гнучкістю, широким спектром функціональності та інтуїтивним інтерфейсом. Вона надає інструменти для створення інтерактивних курсів, відстеження прогресу студентів та співпраці. Canvas LMS також відомий своєю надійністю та швидкістю роботи. Проте вартість використання Canvas LMS може бути високою, а менша популярність порівняно з іншими платформами може обмежити доступність підтримки та ресурсів.

Зважаючи на те, що сучасні Learning Management Systems (LMS) є ключовим інструментом для організації дистанційного навчання, їхні переваги і недоліки відіграють значну роль у визначенні ефективності та задоволення потреб користувачів. Попри те, що LMS надають широкий спектр можливостей, таких як створення курсів, спілкування між учасниками, оцінювання та звітність, вони також можуть зазнавати певних обмежень.

Серед загальних недоліків можна відзначити складність використання для користувачів з різним рівнем технічної підготовки. Наприклад, користувачі з обмеженим досвідом у роботі з комп'ютерами можуть зіткнутися з труднощами при навігації або використанні різних функцій платформ. Також існує проблема з безпекою даних, оскільки збільшення обсягу інформації, що обробляється, може призвести до ризику несанкціонованого доступу чи порушень безпеки. Крім того, деякі LMS можуть вимагати додаткових налаштувань для задоволення конкретних потреб користувачів, що може призвести до збільшення часу та ресурсів, витрачених на розробку та налагодження.

Не дивлячись на ці недоліки, важливо врахувати, що правильно обрана та налаштована LMS може забезпечити значні переваги у реалізації навчальних

програм та підвищити ефективність навчання. Тому вибір підходящої платформи вимагає уважного вивчення її можливостей та обмежень, а також врахування потреб та можливостей користувачів.

1.2. Аналіз мов програмування та аналіз бібліотек

Взагалі можливо використовувати будь яку мову програмування, в якій реалізовано можливість HTTP запроси. Завдяки цьому, ми з’єднуємся до сервісу та можемо керувати ботом, як нам завгодно. Але самі поширені мови програмування для створення бота, є:

- Python
- PHP
- Node.js
- C#
- GoLand

Це основні мови програмування. Серед усіх обрано Python. Python - інтерпретована об'єктно-орієнтована мова програмування високого рівня зі строгою динамічною типізацією. Розроблена в 1990 році Гвідо ван Россумом. Структури даних високого рівня разом із динамічною семантикою та динамічним зв'язуванням роблять її привабливою для швидкої розробки програм, а також як засіб поєднування наявних компонентів. Python підтримує модулі та пакети модулів, що сприяє модульності та повторному використанню коду. Інтерпретатор Python та стандартні бібліотеки доступні як у скомпільованій, так і у вихідній формі на всіх основних платформах. В мові програмування Python підтримується кілька парадигм програмування, зокрема: об'єктно-орієнтована, процедурна, функціональна та аспектно-орієнтована. Python є самим легким та самим зручною мовою. В неї є як і переваги, так і недоліки. Сама перша перевага

– синтаксис. Він простий та зрозумілий. Також у Python є багато сторонніх бібліотек для, майже, всіх завдань які нам можуть пригодитись далі.

Після обрання мови програмування Python, далі треба вибрати бібліотеку для розробки бота. Основна же є Discord.py. Дана бібліотека є основною і вже всі інші бібліотеки, які можливо найти, є на базі цієї бібліотеки (форк бібліотеки). На мій погляд, самий успішний форк даної бібліотеки – є Disnake.

Disnake - сучасна, проста у використанні, багатофункціональна й готова до асинхронності оболонка API для Discord, написана на Python. Завдяки синтаксису та структури бібліотеки discord.py та бібліотеки dislash.py яка доповнювала бібліотеку discord.py, і з'явилась бібліотека disnake. Disnake – симбіоз цих двох бібліотек. Завдяки її регулярним оновленням, ми маємо завжди найновіший функціонал, який додає Discord.

Після додавання нового функціоналу, такого як slash command.

Slash command або слеш команда – це надзвичайно потужний спосіб забезпечити багату інтерактивність для учасників вашого сервера Discord. Все, що потрібно зробити, це ввести «/», і користувачі готові використовувати бота. Користувачі можуть легко побачити всі команди бота, перевірка введених даних та обробка помилок допомагають отримати команду правильно з першого разу. Розробник даної бібліотеки припинив оновлення своєї бібліотеки. Але нещодавно, до його команди долучились нові люди, які зацікавили його до продовження розробки бібліотеки discord.py.

Бібліотека Disnake увібрала в себе всі стабільні функції бета версії, та пропонує свої, нові. Наприклад використання slash command. Для користувача вони дуже зручні, і користувачу не потрібно самому прописувати відповідну команду для того, щоб її використовувати, а лише відкрити меню бота та він має всі команди бота, які можливо використовувати, та також вписувати перемінні. Тому, на мій погляд, бібліотека Disnake зараз має більше функціоналу та можливостей, чим звичайна бібліотека discord.py.

1.3. База даних

В даному проекті використано базу даних SQLite. SQLite - полегшена реляційна система керування базами даних. Втілена у вигляді бібліотеки, де реалізовано багато зі стандарту SQL-92. Сирцевий код SQLite поширюється як суспільне надбання, тобто може використовуватися без обмежень та безоплатно з будь-якою метою. Завдяки тому, що БД зберігається у файлі, ми маємо можливість проглянути її завдяки системами управління базами даних, та побачити на етапі тестів, коректно чи ні записуються дані до БД.

Висновки до розділу 1

Отже, для реалізації бота для проведення дистанційного навчання було проаналізовано та вирішено, що буде використовуватись:

- Мова програмування Python
- Основний сервіс Discord
- Бібліотека для мови програмування Python – Disnake
- Базу даних SQLite
- інтегроване середовище розробки Visual Studio Code;

РОЗДІЛ 2. ПРОЕКТУВАННЯ ДОПОМІЖНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ПРОВЕДЕННЯ ДИСТАНЦІЙНОГО НАВЧАННЯ

2.1 Визначення варіантів використання та структура

Допоміжне програмне забезпечення, тобто бот, створюється з метою полегшити та автоматизувати процес викладання та навчання для навчального закладу. Перед нами постає задача, зробити це максимально зручним для користувачів та максимально гнучким, щоби подалі, було можливо легко додавати функціонал програми. На даному етапі ми маємо три ролі користувачів:

- Викладачі
- Студенти / учні
- Адміністратор

Для кожного із цих користувачів, будуть мати свої функції та можливості, які вони можуть використовувати

Вимоги користувачів.

Викладачі будуть мати доступ до наступних функцій:

1. Починати пару;
2. Змінювати налаштування кімнати пари;
3. Закінчити пару;
4. Додавати групу до пари;
5. Керувати студентом/учнем під час пари.

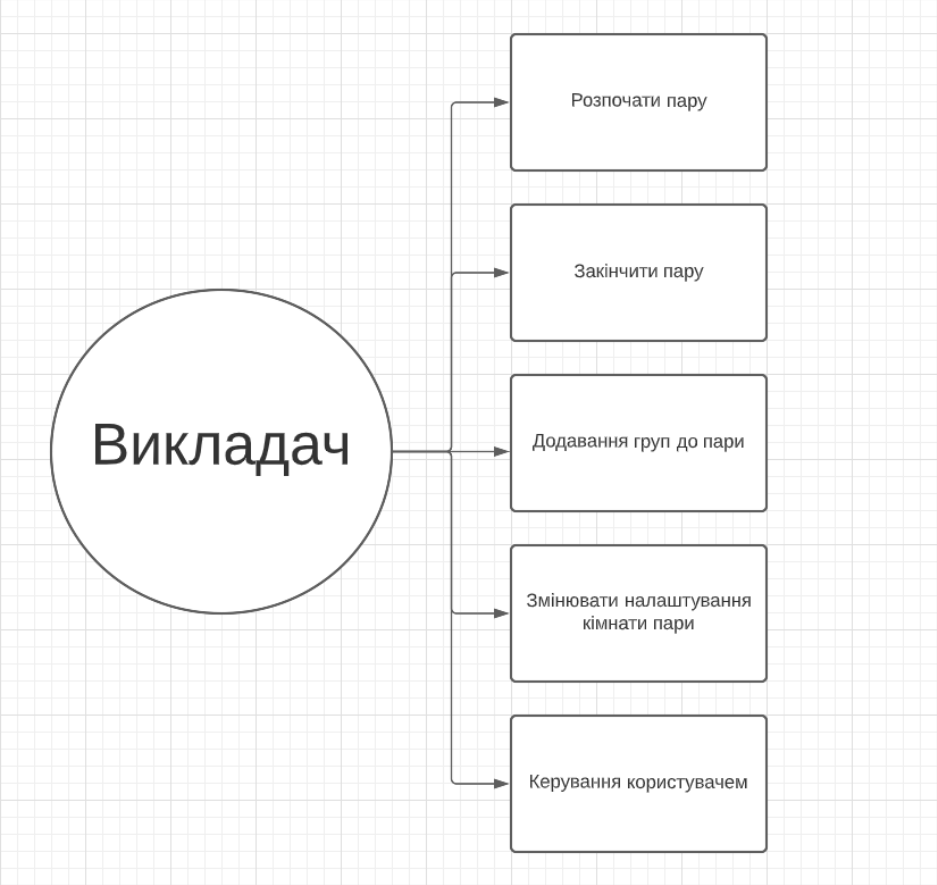


Рис. 2.1. Функціонал викладача

Студенти мають такий функціонал:

1. Підключитись до своєї групи завдяки коду доступу
2. Підключатись до заняття
3. Спілкування в чаті групи

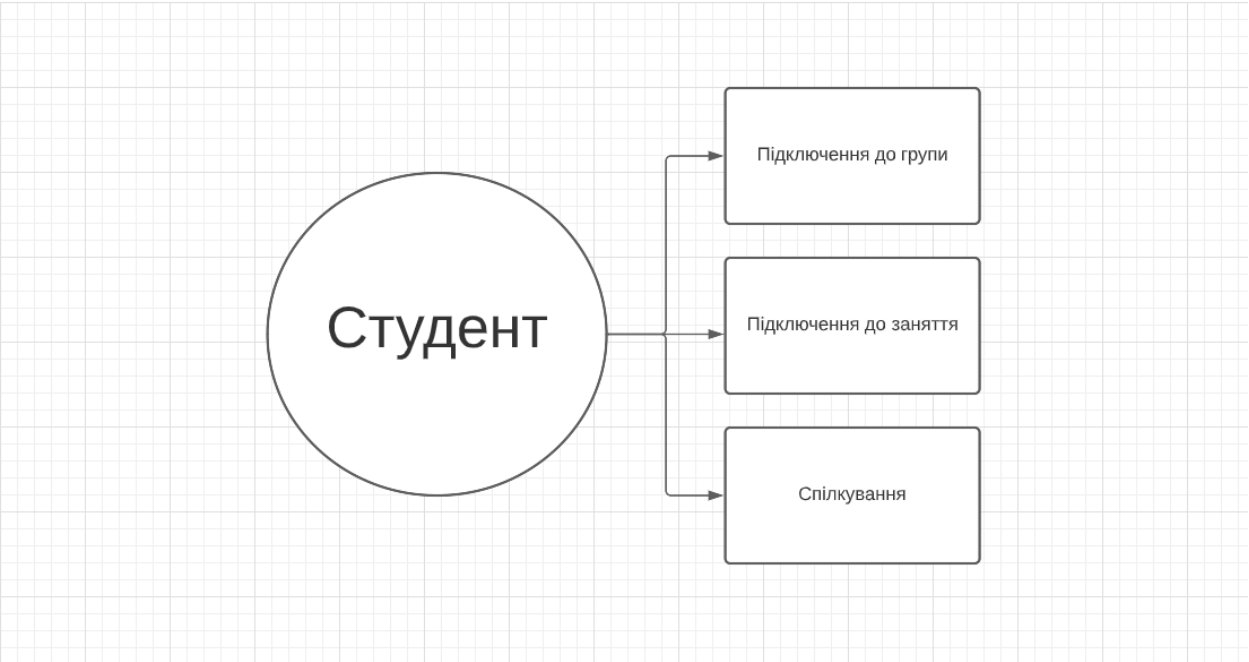


Рис.. 2.2. Фунціонал студента

Адміністратор має весь функціонал викладача, також має і індивідуальний функціонал, а саме:

1. Створювати та видаляти групи
2. Створювати код до приєднання до групи
3. Перегляд вже створених кодів

Рис. 2.3. Функціонал адміністратора

Функціональні вимоги:

1. Викладач повинен мати можливість створити пару завдяки командам та редагувати всі параметри кімнати завдяки командам боту.
2. Додавати групи у разі необхідності. Ця можливість потрібно якщо одна пара і таж пара у декількох груп разом.
3. Адміністратор має можливість створювати групи та видаляти їх при необхідності. Також створювати для студентів код доступу завдяки чому, студент може підключитись до своєї групи.

Нефункціональні вимоги:

1. Сприйняття
   - Час, потрібний для навчання с ботом для звичайних користувачів – 15 хвилин, для досвідчених – 5 хвилин;
   - Час відповіді боту та виконання звичайних дій не повинен перевищувати - 2 секунд, для складних – 3 секунди;
   - Надійність;
   - Безпека
2. Продуктивність. Бот повинен підтримувати безперебійну роботу користувача.
3. Можливість експлуатації:
   - Масштабування – після оновлення програмного забезпечення продукт повинен працювати без перебоїв. Жодних помилок не повинні з’являтись;
   - Оновлення версії – нова версія поширюється завдяки оновленню на github.
4. Технічні характеристики комп’ютера для роботи бота
   - Операційна система Windows або linux;
   - Python версією 3.8. або вища;

– Процесор архітектури x86 або x64 з тактовою частотою не менше 1 ГГц ;
– 10 ГБ вільного місця на жорсткому діску

Аналіз функціональних вимог дозволив виділити наступні сутності, що забезпечать реалізацію програмного комплексу. Діаграми використання програмної системи. (Рис. 2.4. – 2.6.)

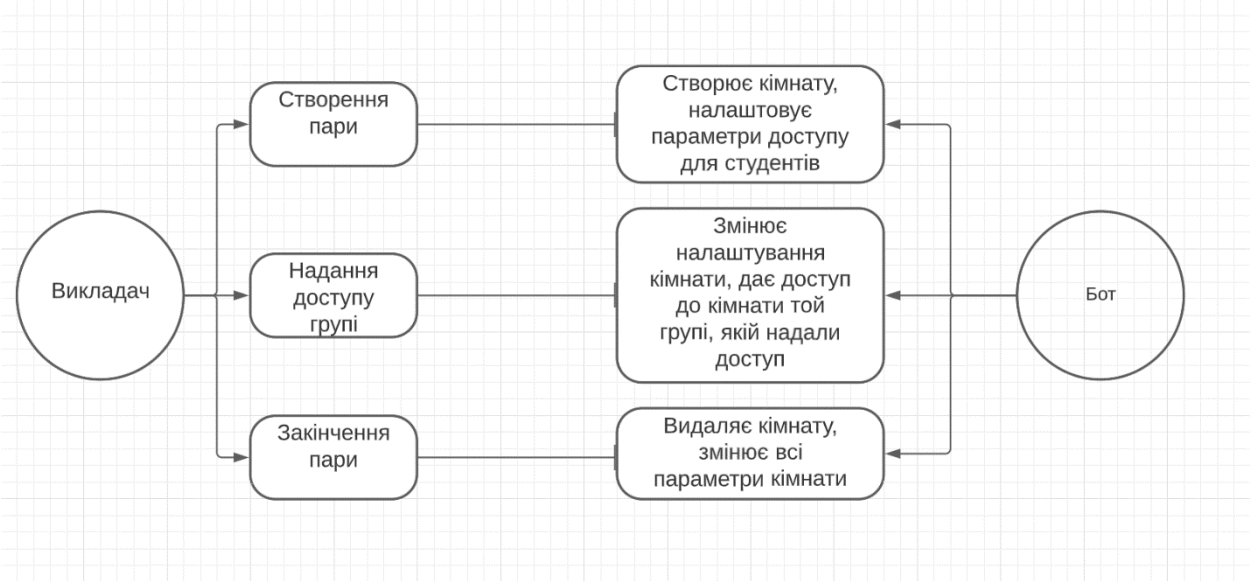


Рис. 2.4. Показ взаємодії бота та викладача

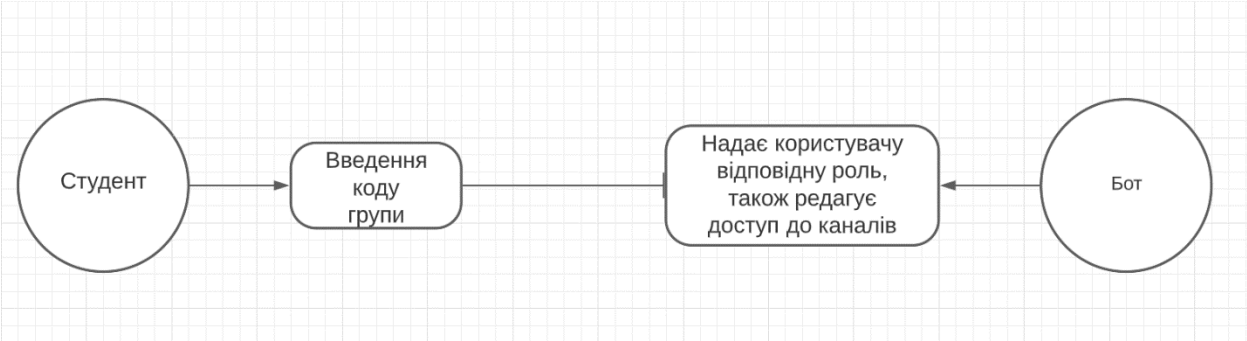


Рис. 2.5. Показ взаємодії бота та студента

Рис. 2.6. Взаємодія Адміністратора та бота

Завдяки діаграмам ми бачимо взаємодію користувачів з ботом. На рис. 2.7. ми бачимо всі класи та функції завдяки яким, працює сам бот.

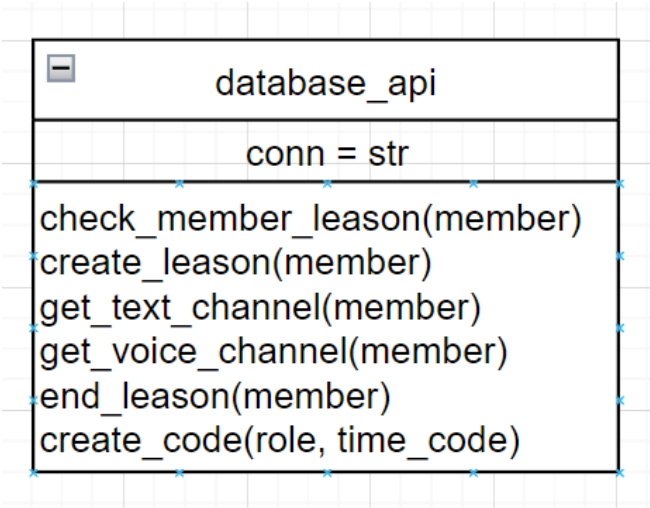


Рис. 2.7. Клас бази даних

Даний клас робить підключення до бази даних та робить записи до БД. Функція check_member_leason – робить перевірку для викладачів, чи вони розпочали вже пару. Дана функція забороняє створювати дві пари для одного викладача, Це зроблено для того, щоби у Discord не було багато розпочатих і не закінчених пар. Дана функція записує до БД час та унікальний ідентифікатор кімнати у Discord, що дає смогу подалі робити маніпуляції з кімнатою.

Функція end_leason – видаляє з бази даних запис про заняття, також після виклику даної функції, кімната автоматично видаляєтеся і адміністратору приходить повідомлення про закінчення заняття. Це зроблено для того, щоб адміністратор міг бачити. Коли викладач розпочав заняття, скільки по часу тривало заняття. Це повідомлення не видно для викладачів та студентів, а тільки для адміністратора.

Функція create_leason – створюється запис у бази даних, в якій записуються користувач, кімната та час створення заняття. Завдяки цьому і перевіряється те, чи є вже заняття у даного викладача.

Функція create_code – створює випадково створений код для студентів, щоб вони могли приєднатись до групи. Даною функцією може користуватись лише адміністратор. В БД записується сам код та тривалість його життя.

Всі ці функції не визиваються на пряму користувачем, а викликаються завдяки іншим функцій, які і викликають дані функції та передають параметри. Так програма має цілісність бази даних, тому що користувачи можуть випадково не так записати, або передати параметри. Тому завдяки додатковими функціями ми и викликаємо функції роботи з базою даних.

На рис. 2.8 буде показані функції, завдяки яким і виконуються всі запити до сервісу Discord. Завдяки цьому ми і маємо весь функціонал в сервісу і бот може коректно працювати.

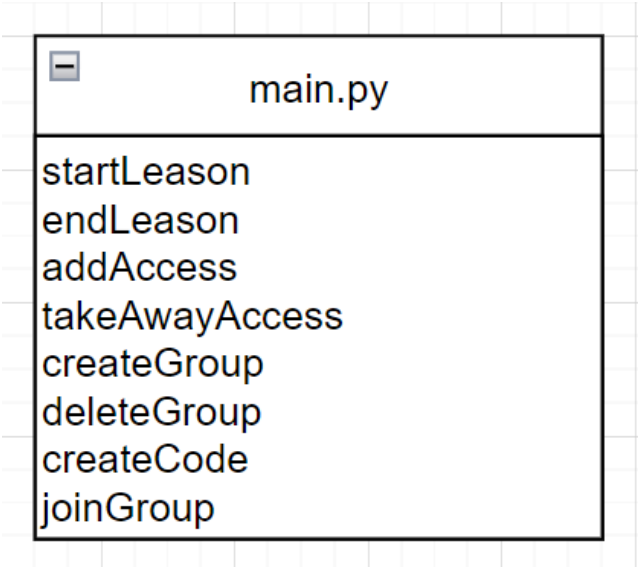


Рис. 2.8. Основні функції бота

Всі ці функції і дають смогу користувачам використовувати спеціальні команди та функціонал боту. Функція strartLeason – цю функцію мають можливість викликати лише викладачі та адміністрація. Дана функція має такі зміні:

Таблиця 1.1 Опис змінних функції startLeason

Назва зміної	**Її описання**
Inter	Автоматична переміна. Ця переміна являю собою об’єкт класу бібліотеки. В цій переміній всі значення користувача, які ми маємо можливість використовувати. Завдяки цій переміній, ми корегуємо кімнати для даного користувача
Name_group	Дана зміна має список ролей, тобто викладач може обрати групу. Для цієї групи автоматично відкривається доступ до заняття, для того, щоб приєднатися до заняття.
Leason_name	Строкова переміна, в якій викладач пише назву предмету, по якій проходить заняття. Бот не дасть змогу створити кімнату без назви, тому користувач повинен заповнювати дане поле.

На скриншотах (рис. 2.9 – 2.10) показано як саме користувач бачить та викликає функцію startLeason.

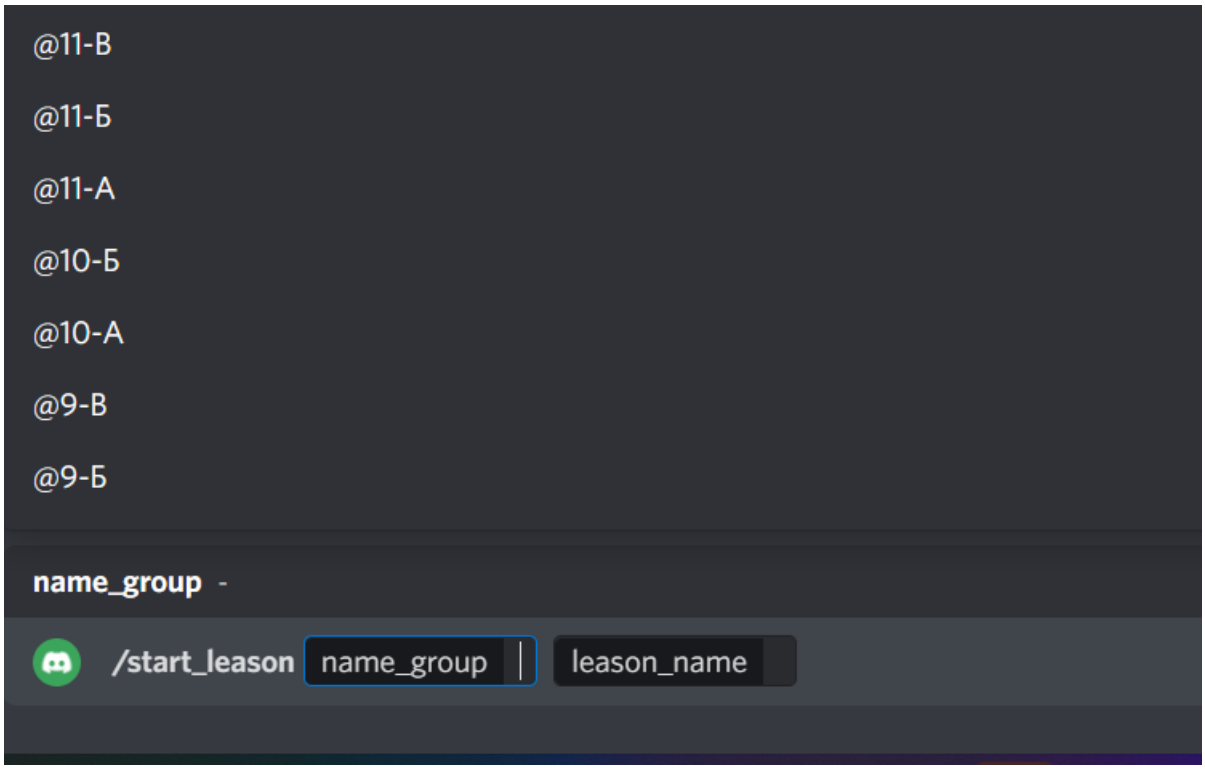


Рис. 2.9. Список змінної name_group

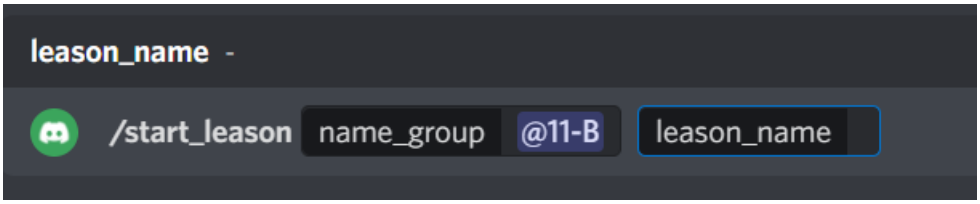


Рис. 2.10. Заповнення leason_name

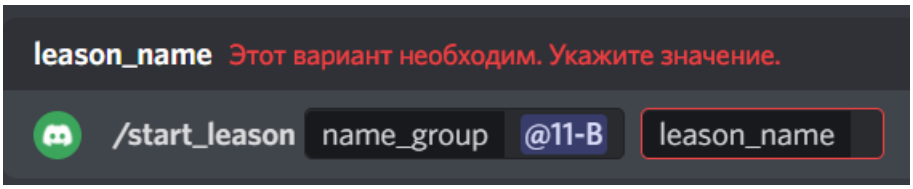


Рис. 2.11. Помилка якщо не заповнювати поле.

На рис. 2.11. вказана помилка, якщо викладач не заповнить всі поля. Завдяки цій функції, в базу даних обов’язково запишиться всі поля, які потребує БД. Наступна функція addAccess – дана функція дає смогу додати доступ до заняття додатковій групі. Ця функції дає можливість додавати до заняття необмежену кількість груп. Завдяки цьому на одному занятті може бути декілька груп одночасно. Дана функція має тільки одну зміну – name_group. Це таж сама зміна, що і у функції startLeason, має можливість вибрати групу або клас із списку.

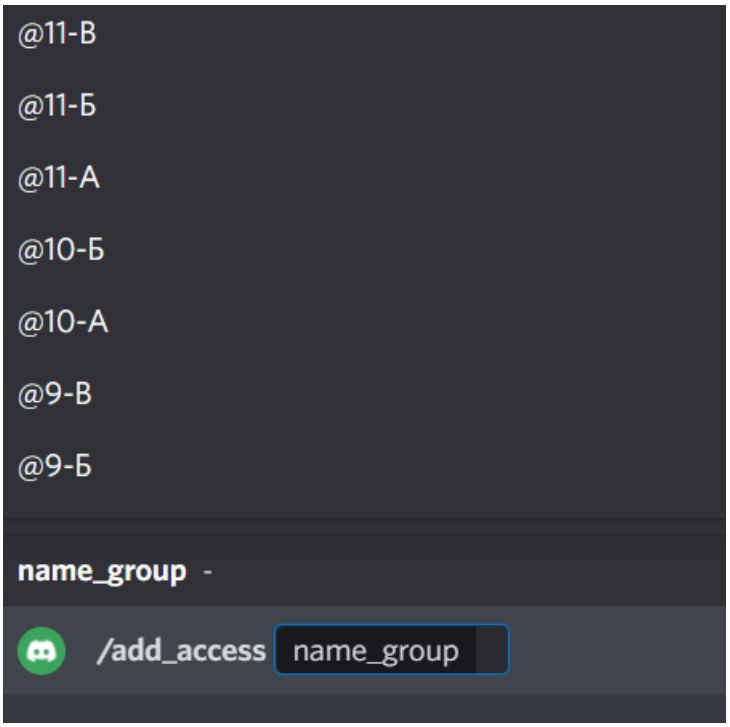


Рис. 2.12. Список вибору у функції addAccess

Групи автоматично додаються завдяки функції CreateGroup. Функція CreateGroup може використовувати лише адміністратор та також має лише одну зміну, це name_group. В даному випадку, дана зміна має не має списку, а являється стрічкою.

В даній зміні адміністратор вручну заповнює назву групи. Після виконання цієї команди, автоматично створюється група, і вона потрапляє до списку у тих функціях, де потрібно вибирати.

Функції endLeason - закінчує заняття. Вона автоматично видаляє кімнату заняття. Ця функція має лише одну зміну, це inter. Дана зміна автоматично застосовується коли викладач використовує дану функцію, тобто викладач не повинен нічого заповнювати (рис. 2.13.). Якщо викладач не розпочинав заняття та використовує дану функцію, то бот йому відправить помилку (рис. 2.14.).

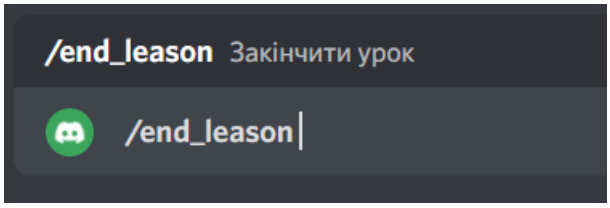


Рис. 2.13. Команда для завершення заняття

Якщо викладач не розпочинав заняття та використовує дану функцію, то бот йому відправить помилку (рис. 2.14.)

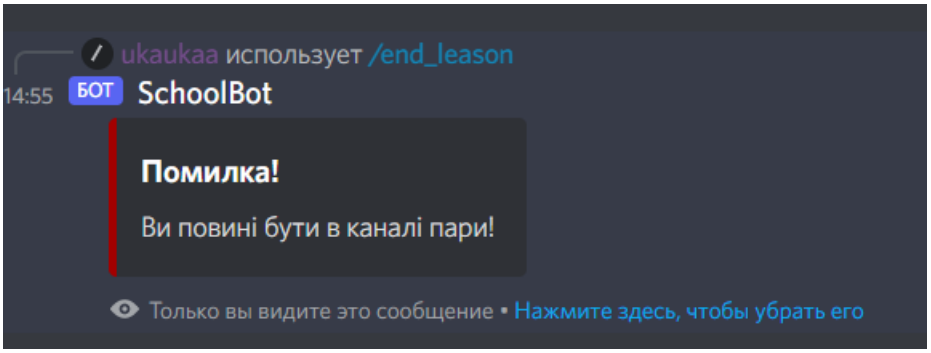


Рис. 2.14. Помилка при завершені заняття

Функція takeAwayAccess аналог функції addAccess, але працює навпаки. Дана функція забирає доступ у групи до заняття. Вона має такіж самі зміні, що і функція addAccess.

Функція joinGroup – функція, яка дає смогу студентам під’єднуватись до групи завдяки випадково генеруємого коду завдяки функції createCode. Функція joinGroup має лише одну зміну, це строкова зміна code. В неъ користувач вписує код групи щоби приєднатись до неї. Після вірно написаного коду користувачу дається роль групи.

Щоб підключитись до групи, адміністратор повинен створити код, для цього він має спеціальну функцію create_code. Дана функція має дві зміні(табл.1.2):

Таблиця 1.2. Опис змінних функції

Назва зміної	Її опис
Name_group	Вибір із списку групи, які створені
day	Кількість днів, протягом яких буде активний код. Після того як певна кількість днів пройде, код автоматично знищиться.

Після заповнення всіх змінних, адміністратору буде написано код у відповідь (рис. 2.20.), і з того часу він починає діяти. Коли пройде та кількість днів, які вказав адміністратор, то код автоматично знищується і стирається з бази даних.

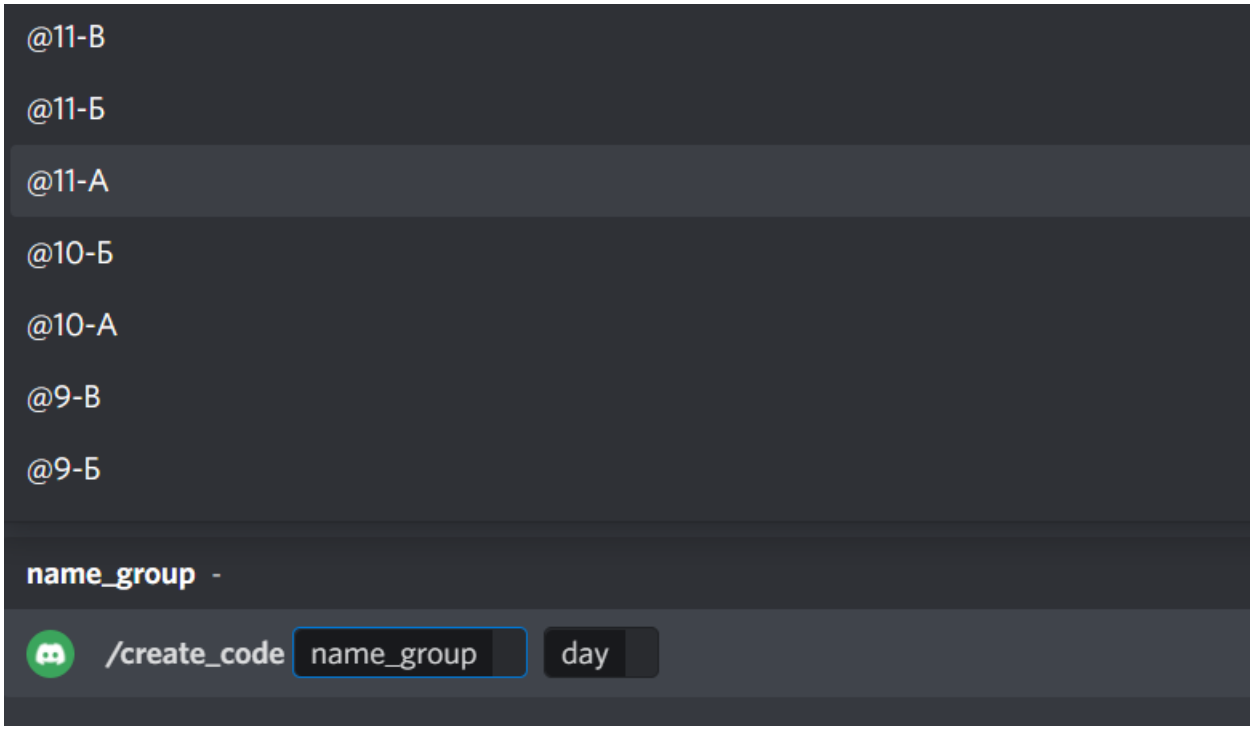


Рис. 2.20. Список вибору групи

2.2. Структура файлів системи

В даному програмному забезпечені структура файлів має велику роль, адже якщо все не розмістити, як правильно, то бот не буде працювати правильно. Отже про структуру файлів також треба згадати.

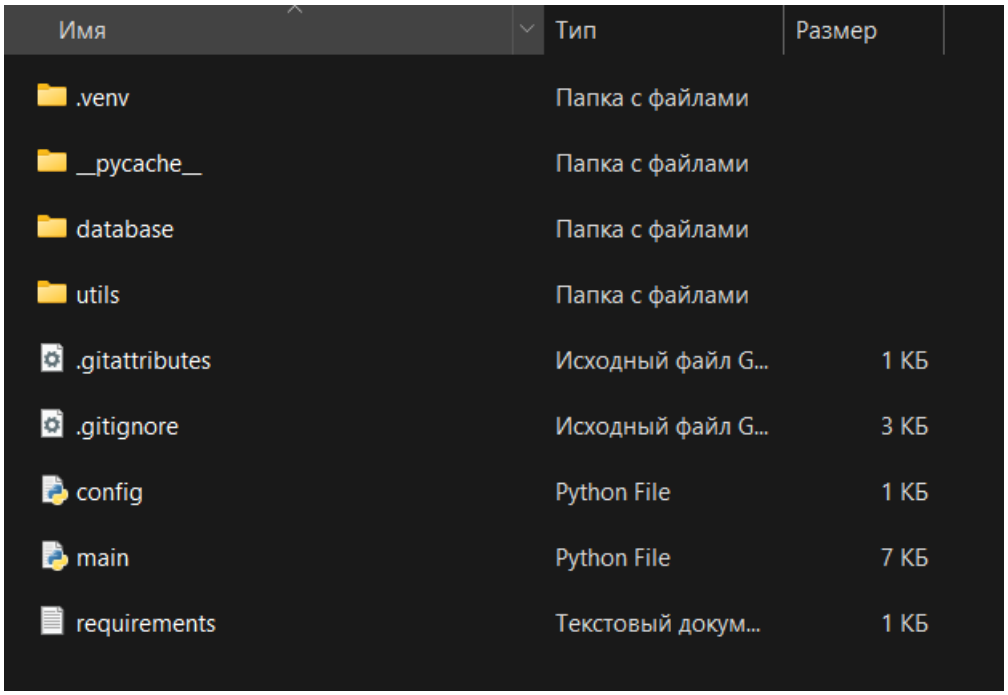


Рис. 2.26. Структура проекту

Самий головний файл main.py, за допомогою його ми запускаємо нашого бота, якій працює. Цей файл повинен мати доступ до всіх файлів проекту, для коректної роботи, тому сама база даних та допоміжні функції мають своє місце у папках.

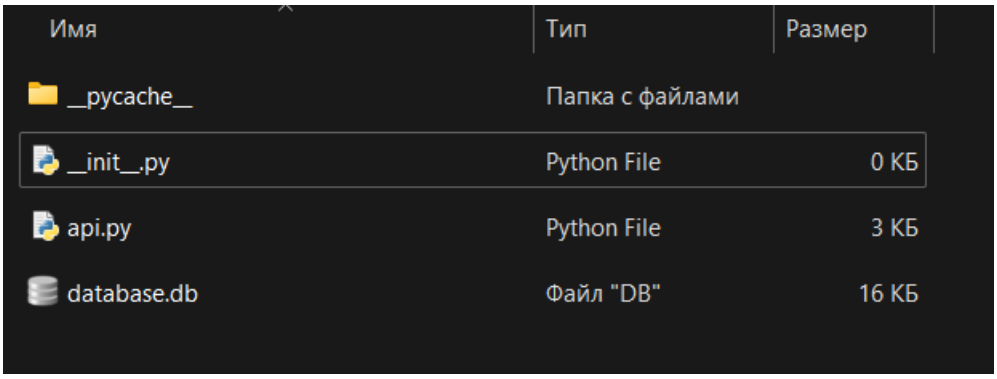


Рис. 2.27. Структура папки database

В даній папці ми маємо два файла.

Таблиця 1.3. Опис файлів, що міститься

Назва файла	Опис файлу
__init__.py	Даний файл помічає для Python, що ця папка може використовуватись як бібліотека. У версії Python3 не обов'язково створювати цей файл, але буде добре, якщо він буде. Також під час ініціалізації бібліотеки, весь код який записан у файлі, буде використовуватись лише 1 раз
api.py	В даному файлі знаходиться сам клас database_api, завдяки якому ми і маємо доступ до бази даних та до функцій запису, видалення редагування та пошуку.
Database.db	Сама база даних. В неї записується всі дані які бот використовує для коректної роботи та також для збереження даних.

Рис. 2.28. Структура папки utils

Отже, розглянуто взаємодію бота зі студентами та викладачами, надано опис функцій, що виконується та структуру файлів для вдалого запуску додатку.

Висновок до розділу 2

Бот для проведення дистанційного навчання реалізує функціонал для проведення занять. В процесі визначення задач до системи було розроблено саме такий фунціонал. В розділі детально розглянута структура, функції та всі дані з якими працює бот. Завдяки цьому, бот сможе коректно працювати.

Було розглянуто також додатково робота з ботов зі сторони викладача, студента та адміністратора. Була описана логіка бота, її взаємодія з кожним користувачем окремо. Також було представлена структура програми та функції, що демонструють основні методи роботи бота.

РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З СИСТЕМОЮ ДИСТАНЦІЙНОГО НАВЧАННЯ

3.1. Порядок налаштування та створення боту у сервісі Discord

Для того, щоб бот мав доступ до сервісу Discord, самого початку ми повині зареєструватись. Після чого нам буде доступно вкладка, як розробник програмного продукту.

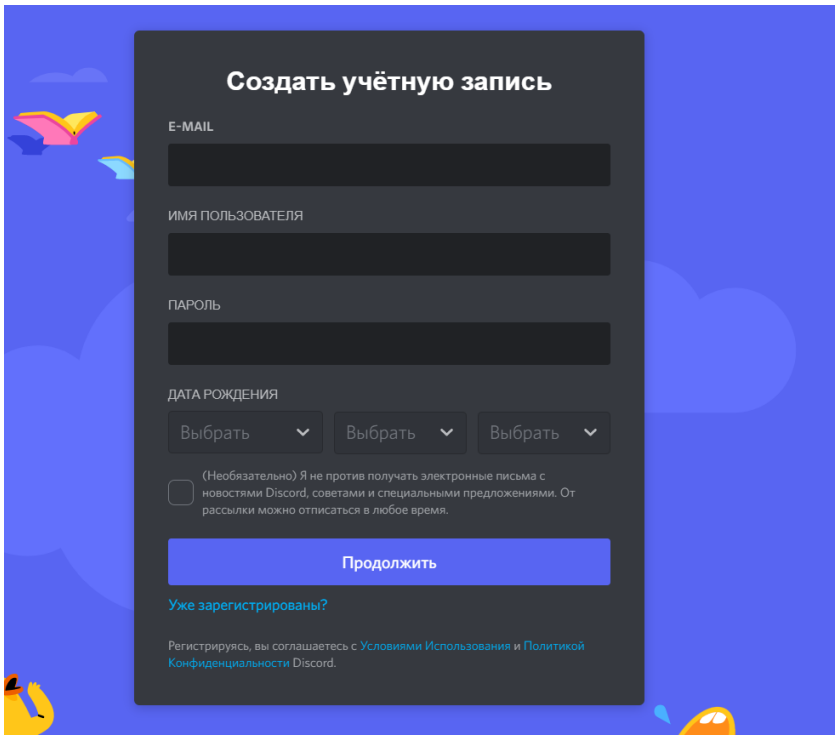


Рис. 3.1. Форма реєстрації у сервісі Discord

Після реєстрації, потрапляємо до самого сервісу Discord(рис.3.2).

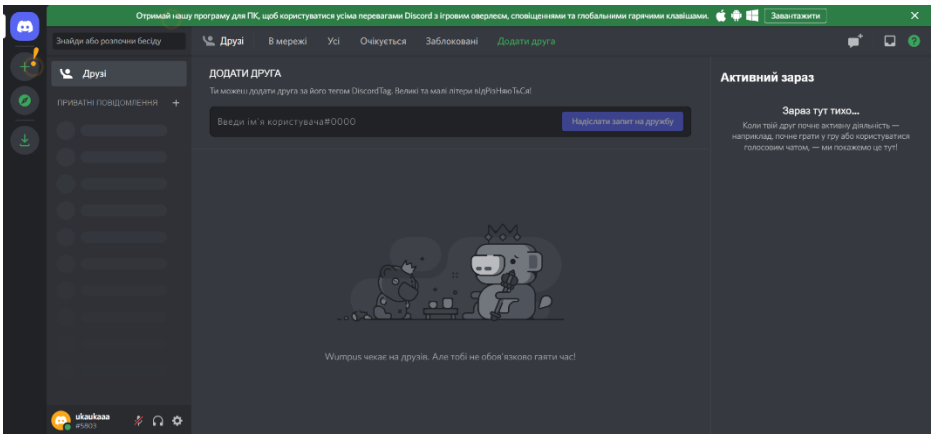


Рис 3.2. Головне меню сервісу Discord

Щоб потрапити у меню розробника, та стати розробником, потрібно перейти до сайту розробника та перейти натиснути на кнопку New Application. Також на цьому сайті ми можемо подивитись документацію до API самого сервісу.

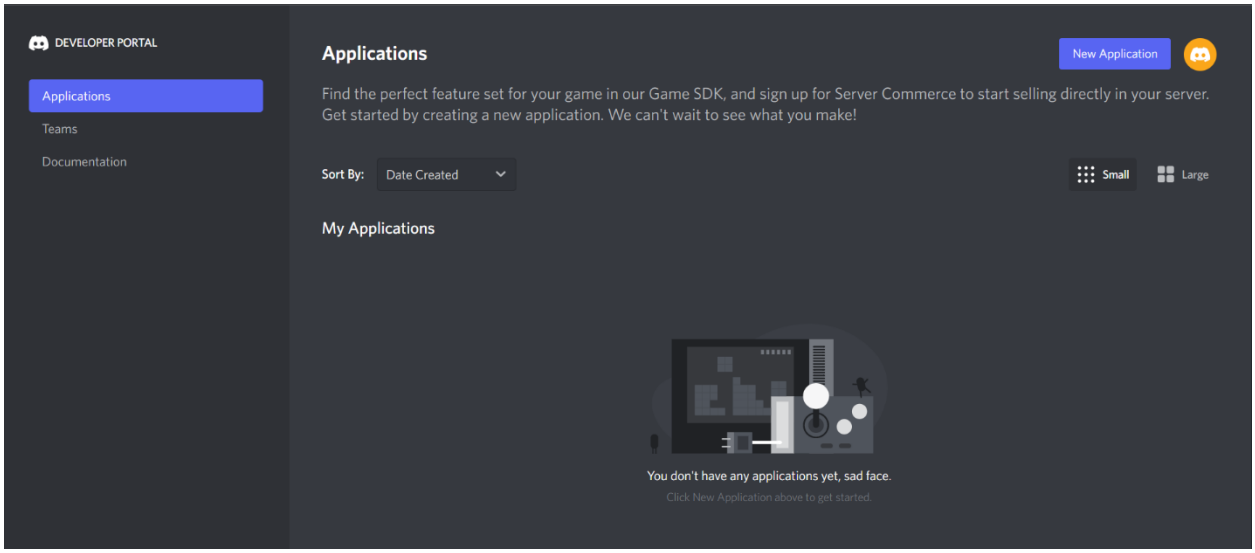


Рис 3.3. Сайт для розробників ботів

Буде запропоновано вести ім’я проекту. Це ім’я видно тільки розробнику і потрібно для того, щоб не сплутати, коли буде велика кількість проектів.

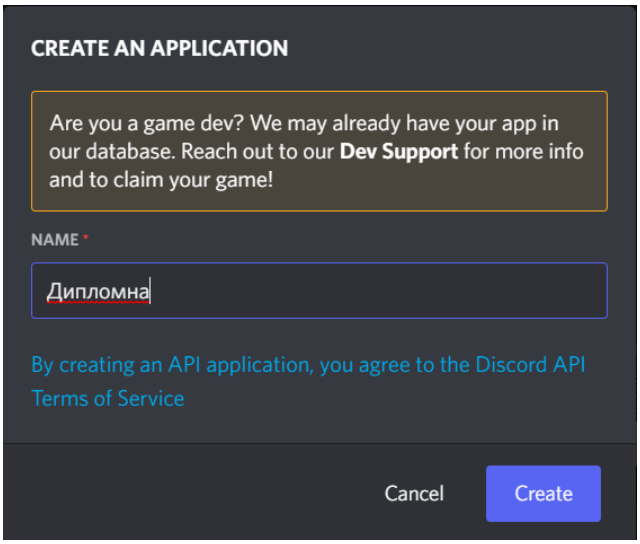


Рис 3.4. Створення проекту та надання назви

Після того як ми надали назву, нас перенаправляє до проекту, де ми маємо можливість налаштувати його. Але для того, щоб бот працював без помилок, потрібно налаштувати його під слеш команди сервісу Discord. З самого спочатку

нам потрібно створити бота та отримати API ключ щоби ми могли взаємодіяти з сервісом. В вкладці «Bot» натискаємо «Add Bot», після чого він створився і користувач має можливість його редагувати та налаштовувати.

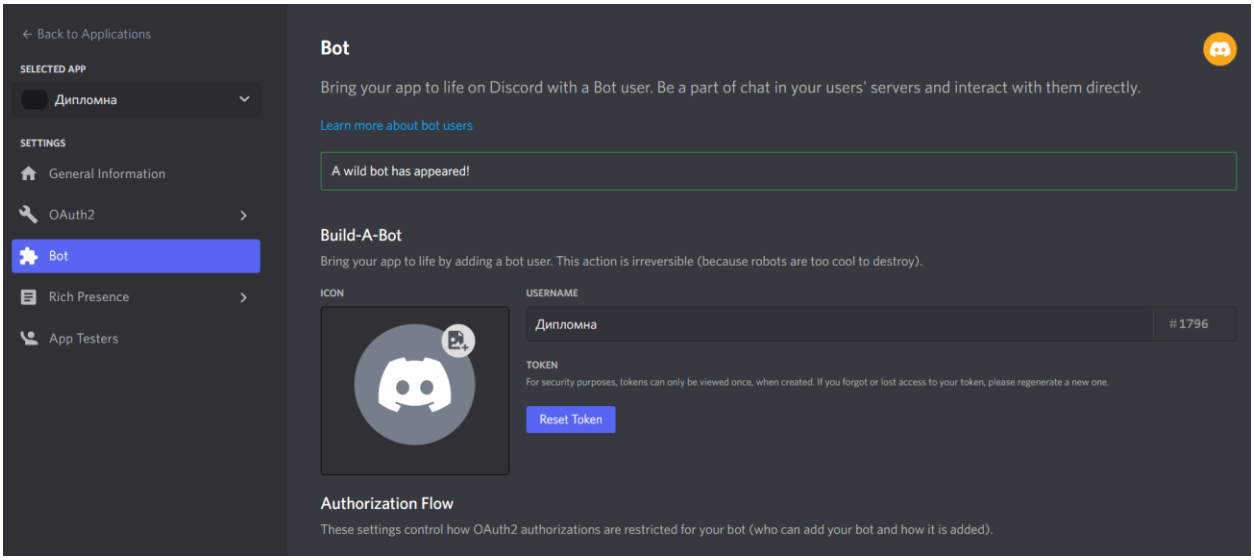
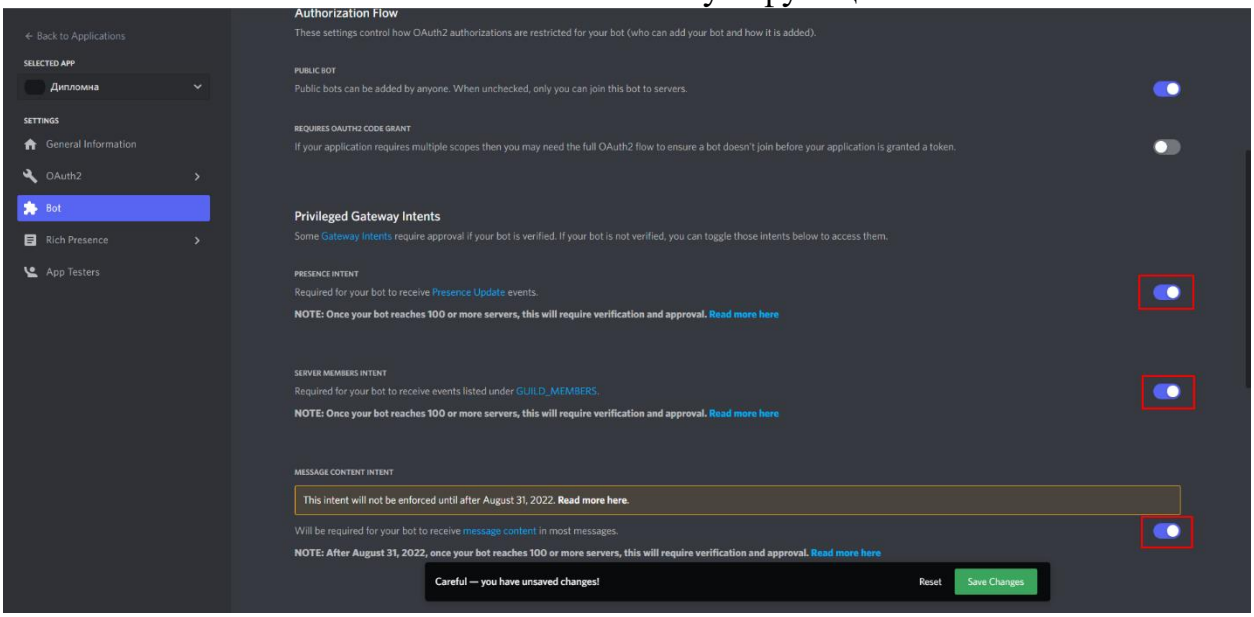


Рис. 3.5. Панель керування ботом.

Щоб отримати API-ключ (токен) користувач повинен натиснути «Reset Token», і токен буде показаний. Далі включаємо всі налаштування, щоб були увімкнуті (Рис 3.6.).

Рис. 3.6. Увімкнуті функції



Після збереження налаштувань, переходимо до вкладки «Oauth2» і вже після цього на додаткову вкладку «URL Generator». Саме в цій вкладці ми отримуємо посилання для запрошення боту до нашого серверу.

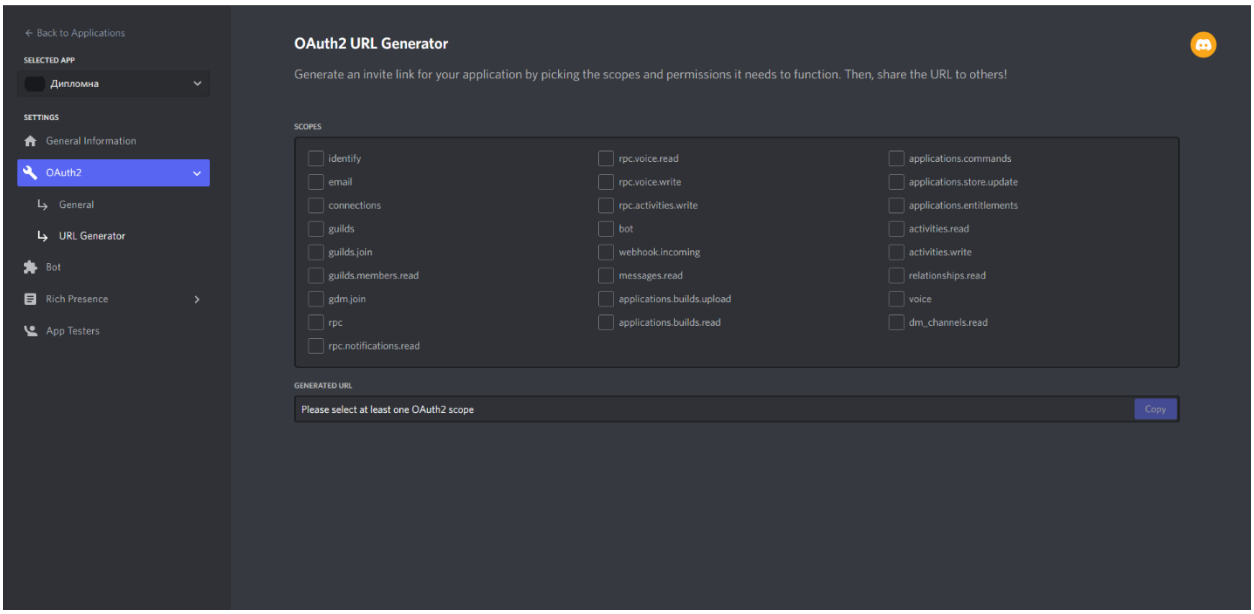


Рис. 3.9. Вкладка «OAuth2»

Саме тут обираємо два пункти: bot та application.commands. Bot означає, що ми запрошуємо бота на свій сервер. Після його вибору bot з'являється Bot permission. Ми вибираємо те, що бот буде вміти робити. Вибираємо Administrator і після того ми можемо копіювати запрошення для боту (рис 3.10.). Пункт application.commands означає, що бот може створювати слеш команди на нашому сервері.

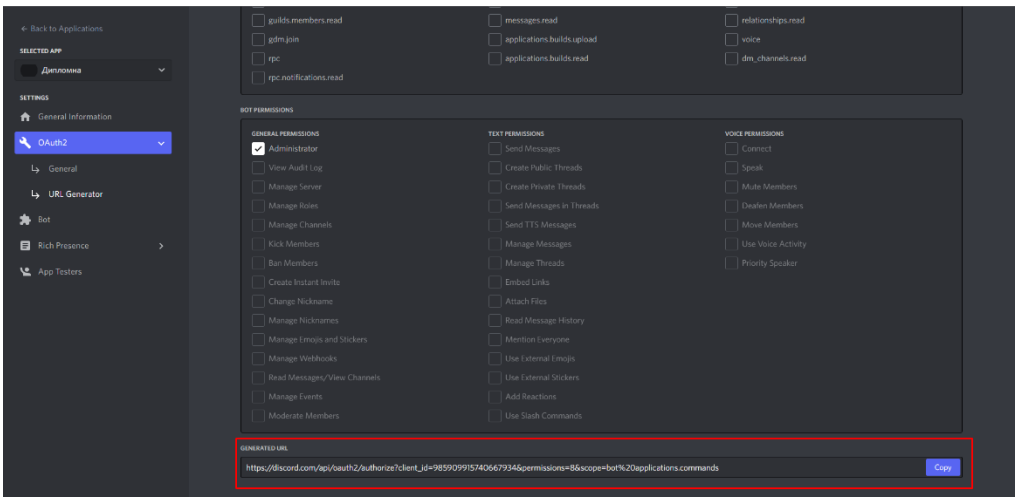


Рис. 3.10. Посилання на запрошення боту на сервер

Тепер переходимо на сам сервіс Discord і створюємо сам сервер. Зліва натискаємо на «+», обираємо пункт створити власний і далі «Для мене та моїх друзів» (Рис 3.11.) після чого водимо назву, буде можливо змінювати, и натискаємо кнопку «Створити»

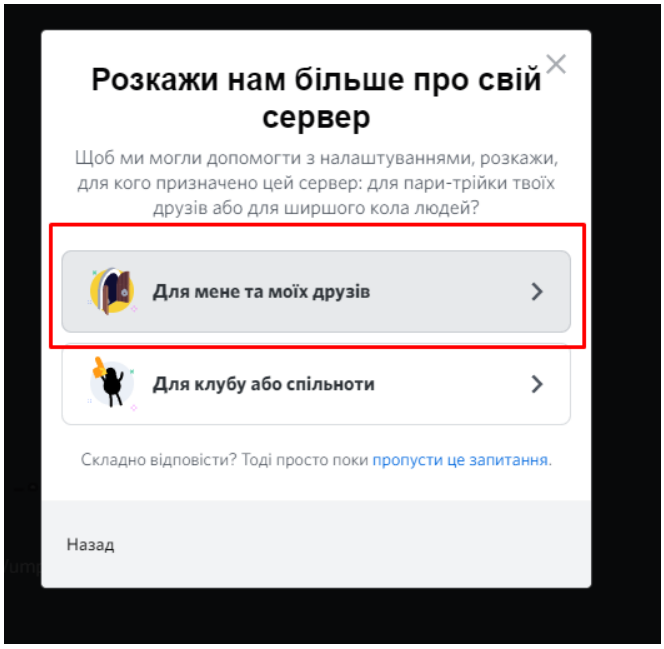


Рис 3.11. Створення серверу

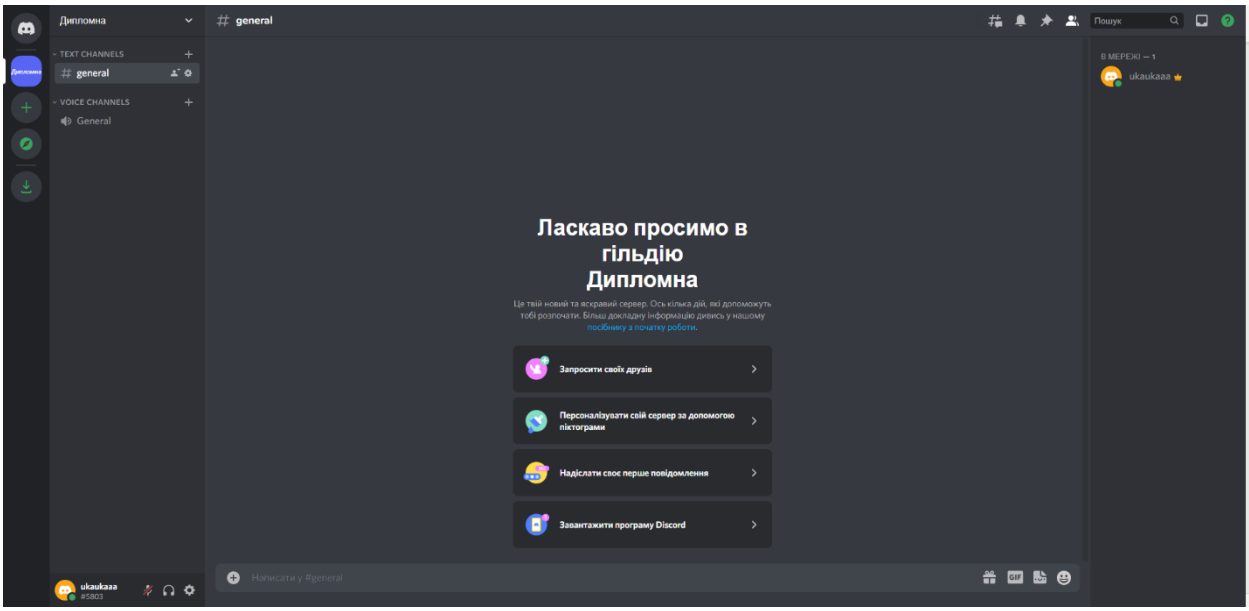


Рис. 3.12. Створений сервер

Після цього залишилось лише налаштування самого серверу для того, щоб бот працював коректно. Заходимо до «Налаштування серверу» та переходимо у вкладку «Увімкнути налаштування спільноти» та натискаємо «Розпочати»

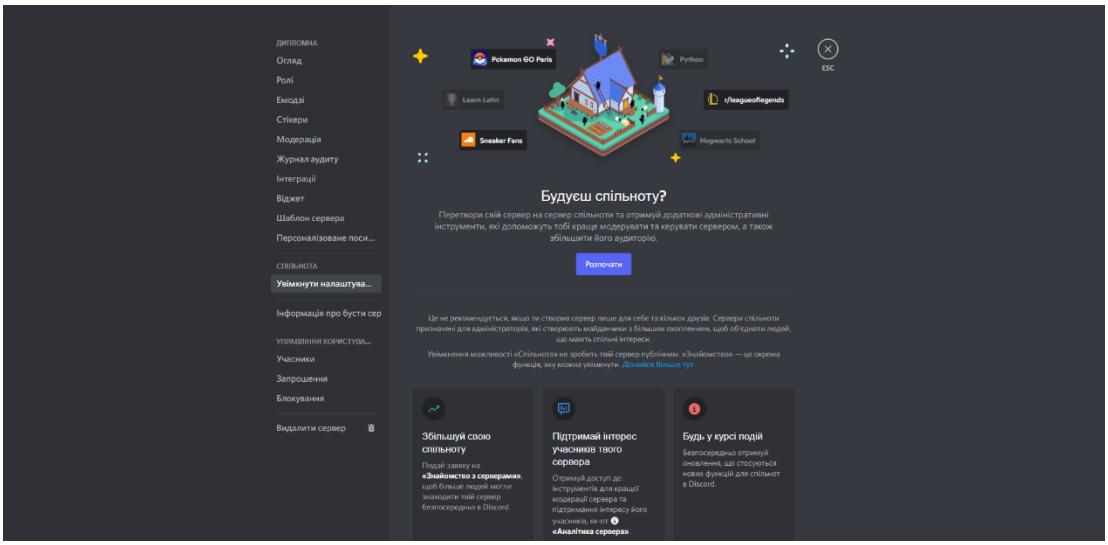


Рис. 3.13. Вкладка «Увімкнути налаштування спільноти»

Після чого ми погоджуємось з усім, що пропонують і в нас з'являються додаткові вкладки. Там ми можемо налаштувати сервер під наш стиль на під наші потреби, але на бота це ніяк не впливає. Далі переходимо до розділу «Ролі» та створюємо дві ролі: Адміністратор та Викладач. Також по замовчуванню буде роль «@everyone».

У Discord ми маємо можливість створювати категорії каналів. Це дуже зручно, тому ми можемо виділити типи каналів. Тому створюємо категорію «Заняття» і видаляємо категорії які було автоматично створені.

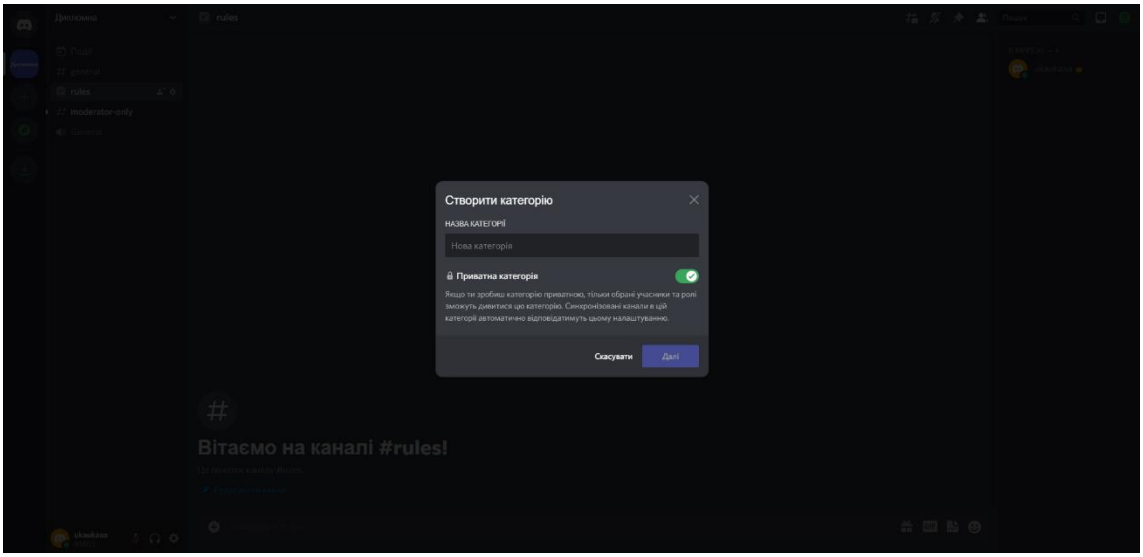


Рис. 3.14. Створення категорії

Перша категорія буде «Заняття». Включаємо налаштування «Приватна категорія». І після натиску далі, вибираємо роль «Викладач» та створюємо категорію. Цю категорію будуть бачити всі, хто має роль «Викладач» та всі користувачі із правами Адміністратор. В даній категорії будуть проводити заняття. В цій категорії створюємо канал, в якому будуть створюватись заняття. Це потрібно для того, щоб тільки викладачі змогли створити заняття. Біля назви категорії натискаємо «+» та обираємо «Text», назва каналу «Розпочаток заняття». Тут користувач може назвати як він бажає. Приватну кімніту не робимо, тому що всі права з категорії синхронізуються.

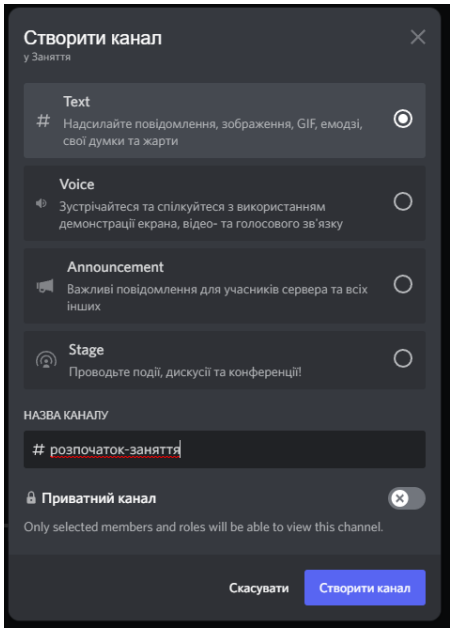


Рис. 3.15. Створення каналу.

Також робимо категорію для адміністрації, там адміністрація буде бачити початок заняття та кінець занять, також буде можливість прописувати всі адміністративні команди. Робимо такі самі дії, коли створювали категорію та канали для викладачів.

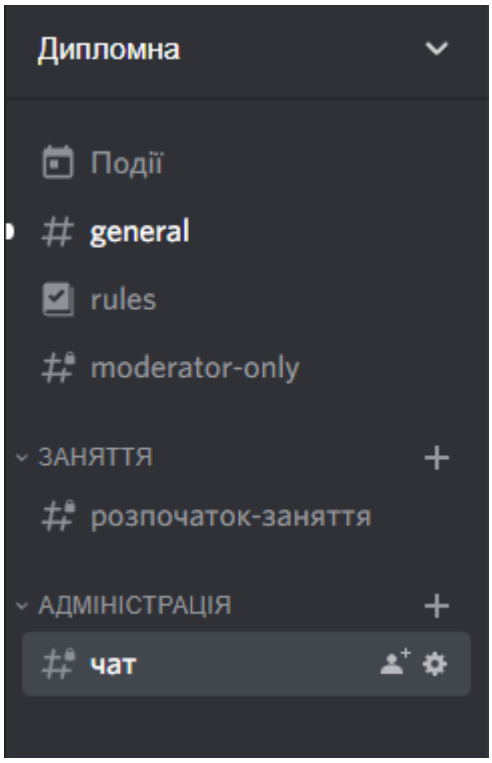


Рис 3.16. Результат створення

Робимо таку же категорію для студентів, там вони будуть мати можливість спілкуватись.

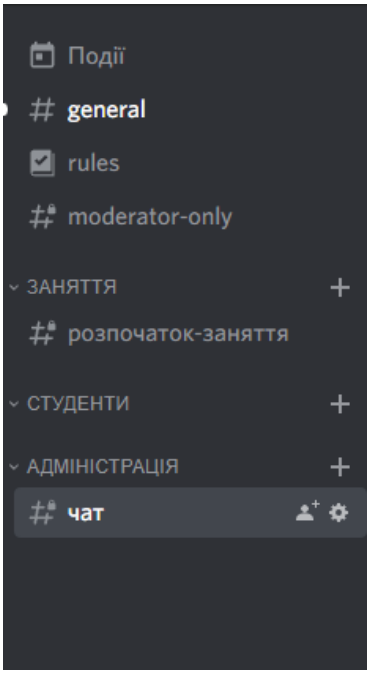


Рис. 3.17. Категорія «Студенти»

Після того, коли ми створили сервер та налаштували, нам потрібно додати бота до серверу. Ми копіюємо посилання на запрошення з Рис. 3.10. та вставляємо до адресної строки.

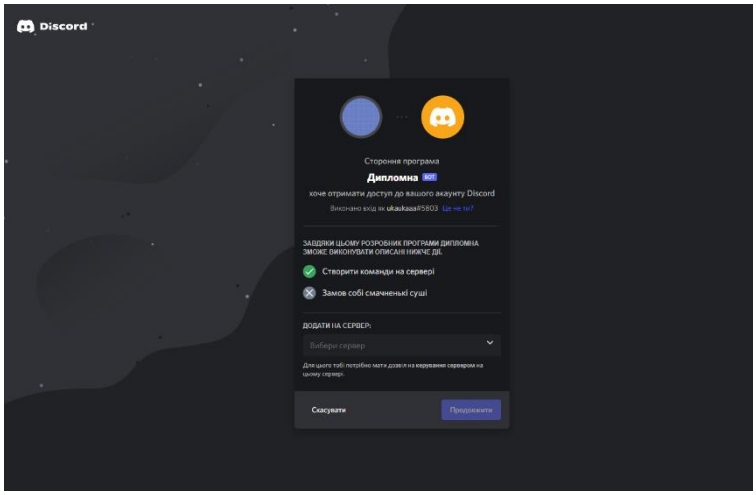


Рис. 3.19. Запрошення боту

Вибираємо наш сервер та натискаємо спочатку «Продовжити», а потім «Авторизувати», після чого нам покажуть, що бот авторизован та автоматично долучився до серверу (рис 3.21.).

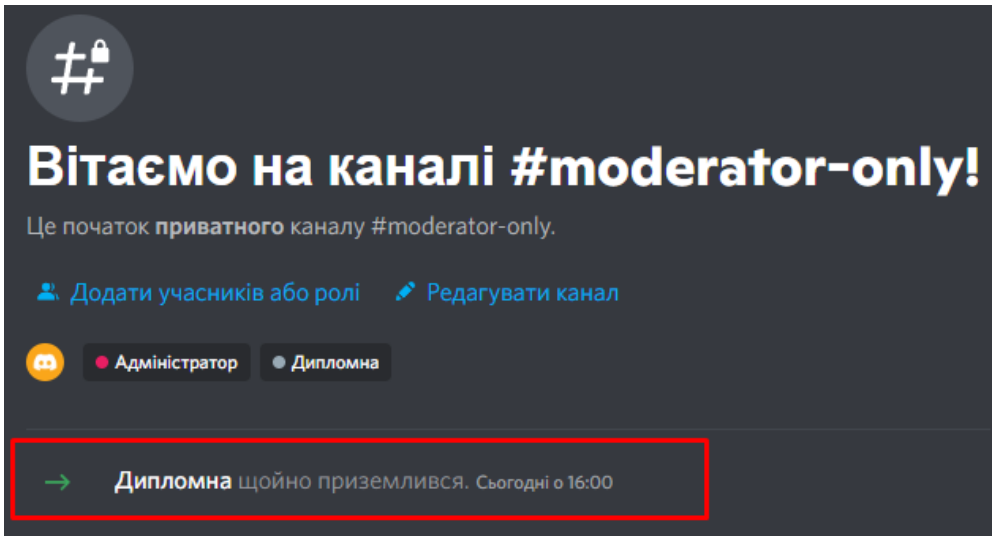


Рис. 3.21. Долучення боту до серверу

Тепер на потрібно видати боту роль «Адміністратор», натискаємо правою кнопкою миші на його справа, де показуються всі користувачі, наводимо на вкладку «Ролі» та надаємо роль «Адміністратор»

Все, налаштування самого серверу на сервісі Discord закінчено. Самий останній крок, потрібно увімкнути у профілі користувача функцію «Режим розробника». Біля свого нікнейма натискаємо на шестерню, після цього переходимо до «Додаткові налаштування» и там активуємо «Режим розробник».

Саме дане налаштування дасть нам змогу копіювати унікальні ідентифікатор для налаштування програми.

3.2. Налаштування інформаційної системи для дистанційного навчання

Налаштування самої програми не є складним. Дане налаштування потрібно для того, щоб бот розумів, у який канал що писати, в якій категорії треба додати канал, а де видаляти.

Для налаштування програми відкриваємо файл у проекті під назвою config.py

```
TOKEN = TOKEN                          # TOKEN BOT

SERVER_ID = [ID]                       # ID Серверу

CHANNEL_START_LEASON = ID              # Чат для початку пари
CATEGORIES_LEASON_ID = ID              # Категорія для проведения пари
CATEGORIES_STUDENTD_ID = ID            # Категорія для студентів

LOG_CHANNEL = ID                       # ID кімнати для логування дій викладача
```

Рис. 3.24. Код файлу config.py

Самого початку нам потрібно скопіювати ідентифікатор серверу. Натискаємо зліва, там де список серверів, правою кнопкою миші, на натискаємо «Копіювати ID». Даний ID нам потрібний для того, щоб бот міг створити команди для використання. Після того як ми скопіювали, вставляємо його замість ID у зміну SERVER_ID

```
TOKEN = 'OTg1OTA5OTE1NzQwNjY3OTM0.Gfp35i.HxqxGLFDw2CBC4_zK548D1fm-GEOAtUR9RguDc' # TOKEN BOT

SERVER_ID = [976970895564369971]       # ID Серверу

CHANNEL_START_LEASON = ID              # Чат для початку пари
CATEGORIES_LEASON_ID = ID              # Категорія для проведения пари
CATEGORIES_STUDENT_ID = ID             # Категорія для студентів

LOG_CHANNEL = ID                       # ID кімнати для логування дій викладача
```

Рис. 3.25. Вставлення ID серверу

Також, із сайту розробника ми копіюємо ключ який ми отримали від бота, та вставляємо в TOKEN.

Далі копіюємо ID чату «розпочаток-заняття». Все тіж самі дії, як і з копіюванням ID серверу, і вставляємо в CHANNEL_START_LEASON.

Так же само копіюємо чат «чат» в категорії «Адміністрації» та вставляємо замість «LOG_CHANNEL».

Залишились лише категорії. Натискаємо ПКМ та копіюємо їх індентифікатор та вставляжмо відповідно. Категорія «Заняття» в зміну CATEGORIES_LEASON_ID, а категорію «Студенти» до CATEGORIES_STUDENT_ID.

```
TOKEN = 'OTg1OTA5OTE1NzQwNjY3OTM0.Gfp35i.HxqxGLFDw2CBC4_zK548D1fm-GEOAtUR9RguDc' # TOKEN BOT

SERVER_ID = [976970895564369971]                        # ID Серверу

CHANNEL_START_LEASON = 986248785825005570               # Чат для початку пари
CATEGORIES_LEASON_ID = 986244430824636467               # Категорія для проведения пари
CATEGORIES_STUDENT_ID = 986250495838527490              # Категорія для студентів

LOG_CHANNEL = 986249722698301451                        # ID кімнати для логування дій викладача
```

Рис. 3.26 Заповнення файлу config.py

Все, налаштування бота ми виконали. Залишилось лише протестувати його на нашому сервері.

3.3 Тестування бота

Для запуску бота, спочатку треба встановити відповідні бібліотеки. Це ми робимо завдяки команді pip install -r requirements.txt це на ОС Windows або pip3 install -r requirements.txt на всіх інших ОС. Після того, як ми запустили бота, консоль нам напише назву нашого боту, та статус «запустився».

```
(.venv) PS C:\Users\Funny\Documents\GitHub\Diplom> & c:/Users/Funny/Documents/GitHub/Diplom/.venv/Scripts/python.exe c:/Users/Funny/Documents/GitHub/Diplom/main.py
Bot Дипломна#1796 Started
```

Рис. 3.27. Статус бота

Після цього ми переходимо до сервісу Discord та тестуємо функції:

– Тестування функції create_group:

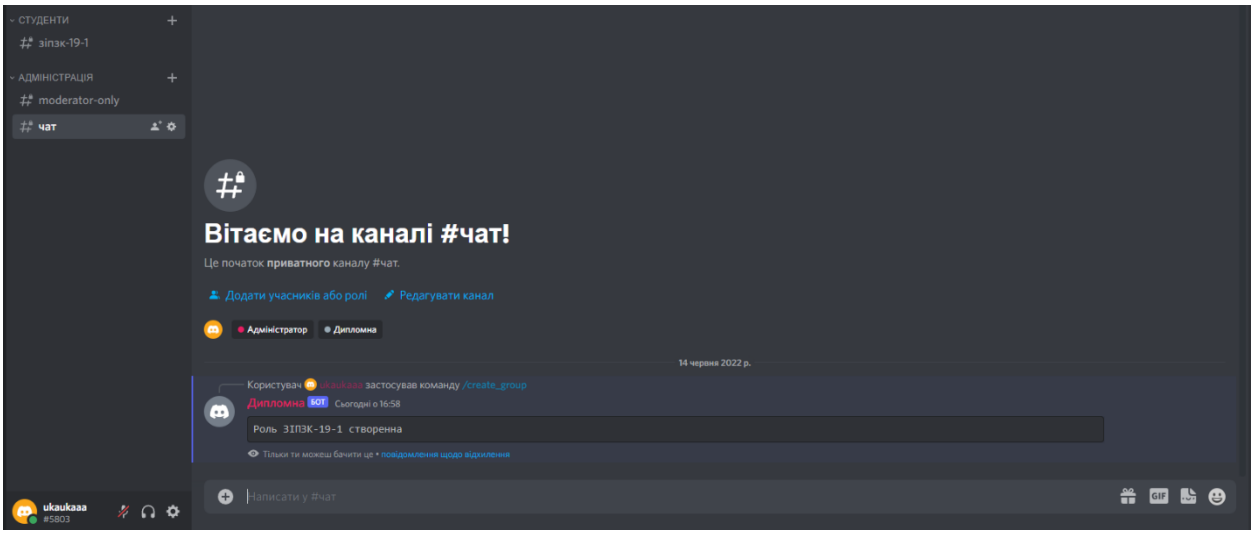


Рис. 3.28. Тестування функції

Як ми можемо бачити, створилась роль для групи та створився чат для студентів цієї групи.

– Тестування функції start_leason

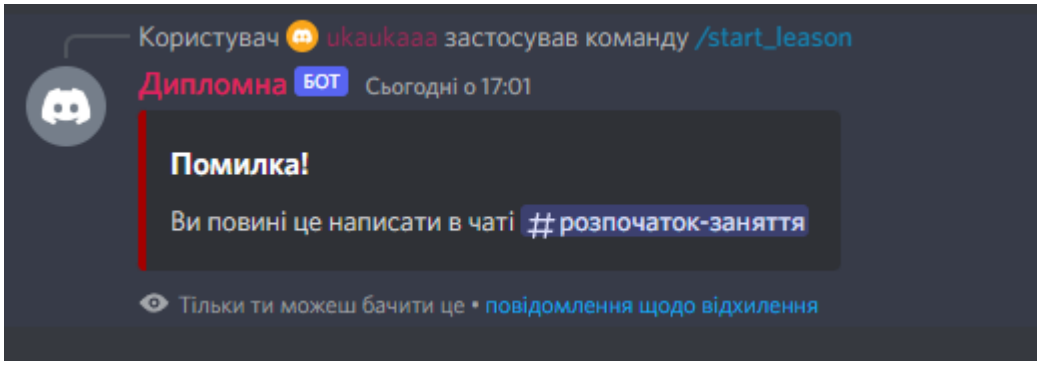


Рис. 3.29. Виклик функції не чаті «розпочаток-заняття»

Якщо ми пишемо цю команду в будь-якому іншому чаті, то бот дає помилку нам. Але якщо знову напишемо фунуцію, але вже в тому чаті, то заняття буде створено.

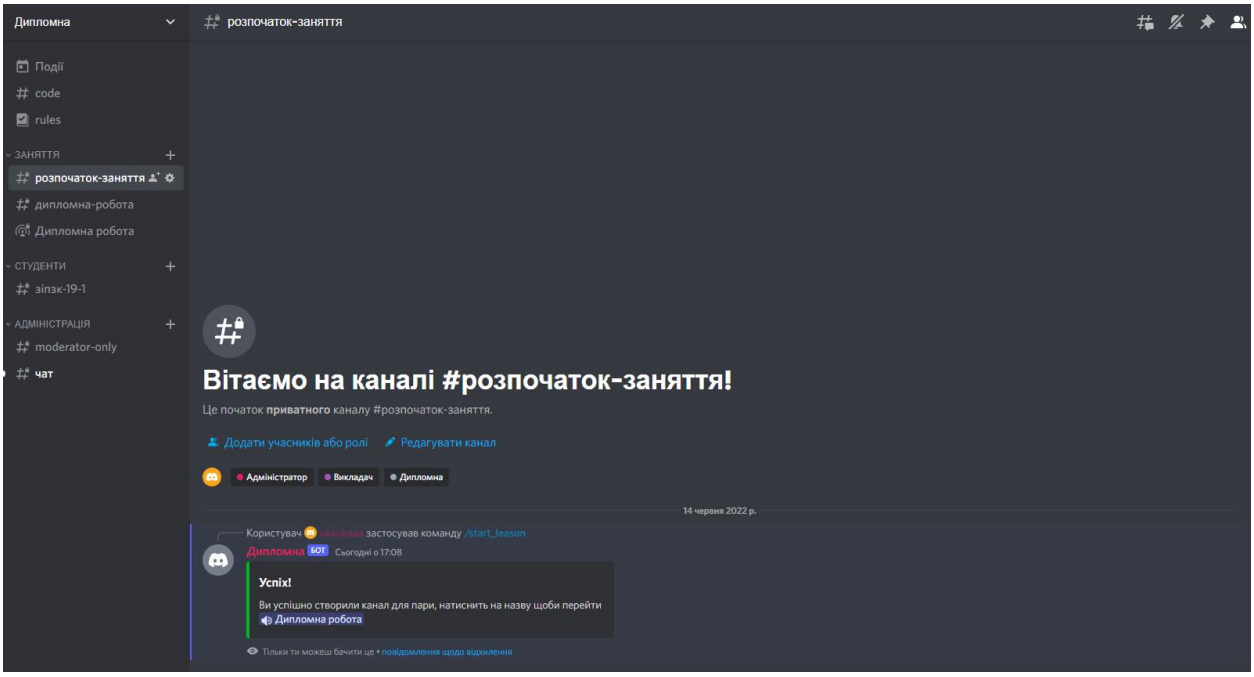


Рис. 3.30 Успішне виконання команди

Кімнати були створені, теперь викладач може переходити до каналу та проводити заняття.

– Функція кінця пари

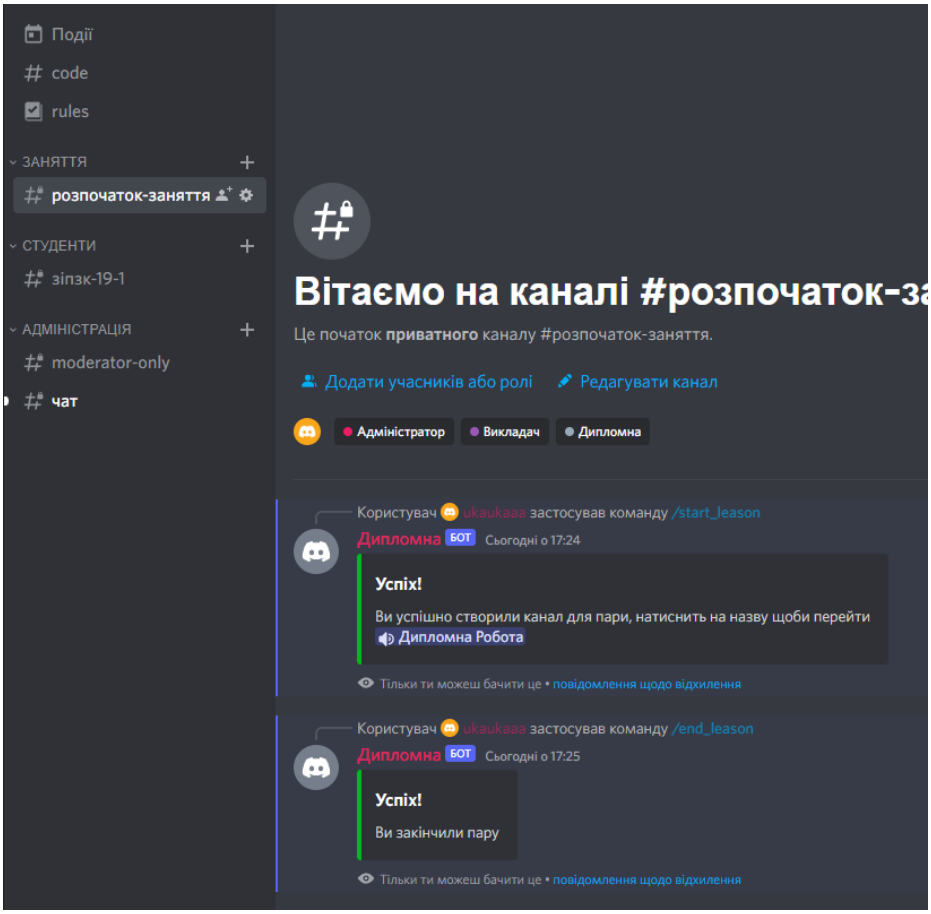


Рис. 3.31. Успішне закінчення заняття.

Як тільки викладач закінчив заняття автоматично видаляються всі додаткові канали. Також адміністрації приходять повідомлення як про початок пари, так і про завершення.

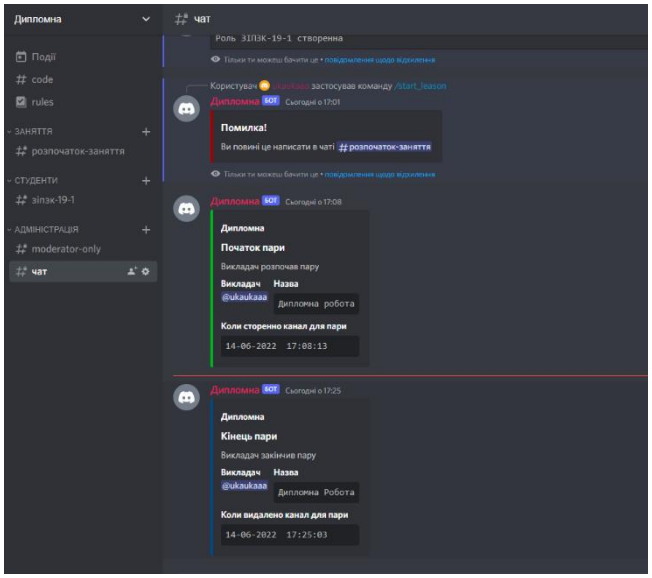


Рис. 3.32. Повідомлення про заняття

Адміністратор створює код і він вноситься до бази даних. По цьому коду студенти мають можливість приєднатись до групи. Коли вони це зроблять, вони автоматично будуть бачити всі заняття своєї групи, та бачити чат своєї групи.

– Захід по коду

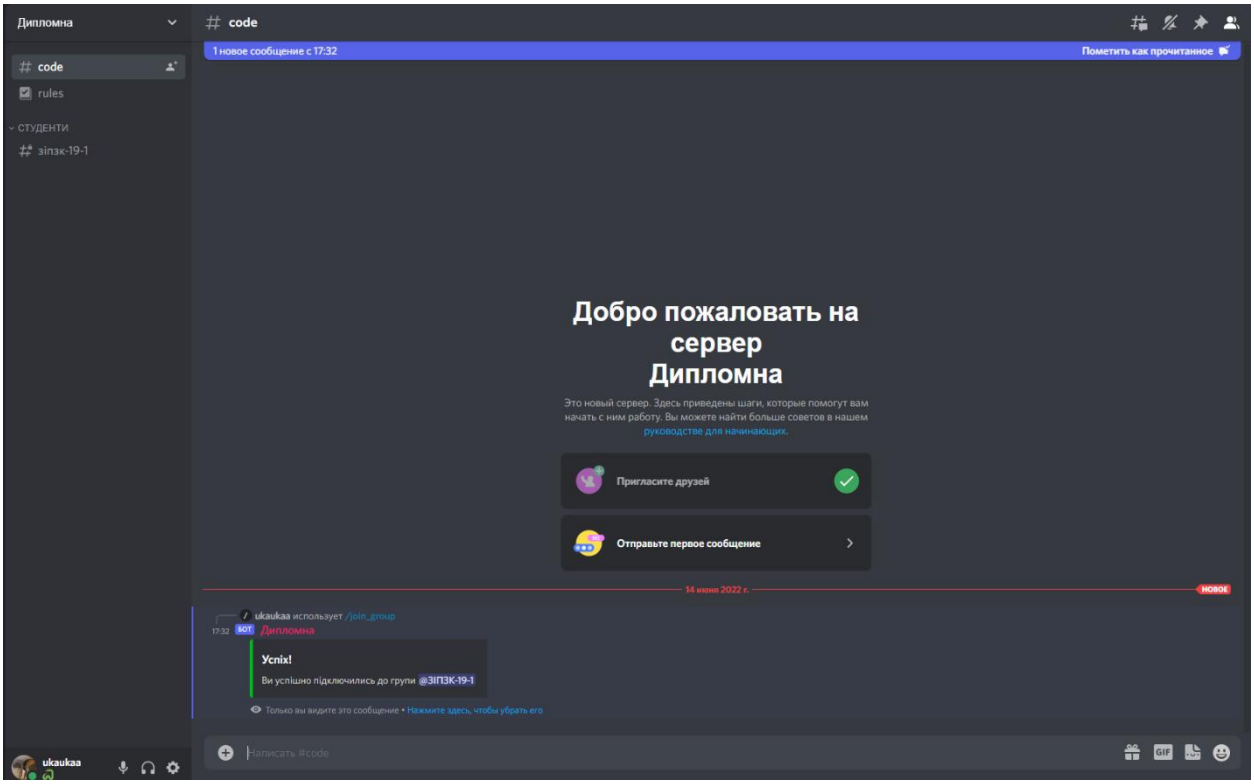


Рис. 3.34. Заход по коду.

На Рис. 3.34 ми бачимо екран зі сторони студента, він бачить тільки чат своєї групи. Так як в даний час немає активних занятть. Але якщо вони будуть, вони з’являться над чатом, де буде можливість зайти до заняття.

Висновки до розділу 3

В даному розділі було продемонстровано кожен крок по налаштуванню боту. Розглянуті нюанси по створенню як самого бота на сервісі, так і створення серверу.

Було детально розписано про стовщенню серверу у сервісі. Всі налаштування і також приєднання бота до серверу.

Було також продемонстровано роботу всі команд від лиця звичайного користувача, викладача та адміністрації. Також проглянуті помилки, які можуть утворитись під час роботи с ботом.

ВИСНОВОК

Під час кваліфікаційної роботи було спроектовано та реалізовано бота для дистанційного навчання. Основна увага при розробці приділялось розробці логіки роботи самої системи. При розробці бота для дистанційного навчання були вирішені наступні завдання:

1. Задля розуміння даної задачі було детально досліджено саме принцип проведення заняття та його тонкість. Проаналізовані сервіси для дистанційного навчання та популярні бібліотеки для створення бота.
2. Були проаналізовані функціональні та нефункціональні вимоги до бота, для чого побудовано діаграму варіантів використання бота та розділено на декілька ролей.
3. Проведено аналіз з роботою баз даних, вибрана база даних та написано функціонал, для виростання.
4. Було вибрано найкращу мову програмування для виконання даної задачі.
5. Протягом всього часу розробки, проводилось постійне оновлення та оптимізації програмного коду, тестування роботи окремих функцій, бази даних щоби все виконувалось коректно та не було ніяких помилок під час роботи бота.

В результаті роботи було створено бота для дистанційного навчання який готовий до використання. Система може бути використана в будь яких навчальних закладах та бути налаштована під неї.

В подальшому даний бот може бути розширений за допомогою доопрацювання та додавання нового функціонала. Завдяки простоті та легкості додавання нового функціонала буде зроблено швидко та просто.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Документація Disnake [Електронний ресурс] – Режим доступу до ресурсу: https://docs.disnake.dev/en/latest/
2. Python [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Python
3. Discord.py Documentation [Електронний ресурс] – Режим доступу до ресурсу: https://discordpy.readthedocs.io/en/stable/
4. SQL in 10 Minutes, Sams Teach Yourself // Бен Форт – 2012 – 288 ст.
5. The Blender Python API: Precision 3D Modeling and Add-on Development // Крис Конлан – 2017 – 158 ст.
6. MySQL for Python // Элберт Лукашевский - 2010 – 440 ст.
7. SQL Сборник рецептов // Энтони Молинаро – 2009 – 668 ст.
8. Using SQLite // Jay A. Kreibich – 2010 – 530 ст.
9. Документація SQLite [Електронний ресурс] – Режим доступу до ресурсу: https://www.sqlite.org/docs.html
10. Python для сетевых инженеров [Електронний ресурс] – Режим доступу до ресурсу: https://pyneng.readthedocs.io/ru/latest/
11. Документація Discord [Електронний ресурс] – Режим доступу до ресурсу: https://discord.com/developers/docs/intro
12. Понимание SQL // Мартин Грубер – 1993 – 291 ст.
13. Discord [Електронний ресурс] – Режим доступу до ресурсу: https://discord.com
14. Python. Книга рецептов // Дэвид Бизли, Брайан К. Джонс – 2019– 650 ст
15. Python. Лучшие практики и инструменты // Яворски Михал, Зиаде Тарек – 2021 – 560 ст.