ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО–НАУКОВИЙ ІНСТИТУТ
ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
Кафедра інженерії програмного забезпечення

ПОЯСНЮВАЛЬНА ЗАПИСКА

до бакалаврської роботи
на ступінь вищої освіти бакалавр
на тему: « Розробка web-додатку « Веб-порталу Університету » мовою C# »

Виконав: студент 4 курсу, групи ПД– 43
спеціальності
121___Інженерія програмного забезпечення
(шифр і назва спеціальності)
_____Мельник М.А._____
(прізвище та ініціали)
Керівник Поперешняк С.В._____
(прізвище та ініціали)
Рецензент_____
(прізвище та ініціали)
Нормоконтроль _____
(прізвище та ініціали)

Київ – 2022

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра Інженерії програмного забезпечення

Ступінь вищої освіти - «Бакалавр»

Спеціальність - 121 «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного
забезпечення
О.В. Негоденко
“____” _____2022 року

З А В Д А Н Н Я

НА БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТУ

Мельнику Максиму Аркадійовичу

(прізвище, ім‘я, по батькові)

1. Тема роботи: « Розробка web-додатку «Веб-порталу Університету» мовою C# ».

   Керівник роботи Поперешняк С.В., к.ф.-м. н., доц. кафедри ІПЗ,

   (прізвище, ім‘я, по батькові, науковий ступінь, вчене звання)

2. Строк подання студентом роботи 30.05.2022.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської роботи	Строк виконання етапів роботи	Примітка
1	Підбір науково-технічної літератури	11.04.2022	Виконано
2	Дослідження існуючих інструментів для автоматизації тестування ПЗ	15.04.2022	Виконано
3	Проєктування архітектури системи	18.04.2022	Виконано
4	Розробка системи для автоматизації тестування ПЗ	22.04.2022	Виконано
5	Висновки, оформлення роботи	26.04.2022	Виконано
6	Розробка демонстраційних матеріалів	05.05.2022	Виконано
7	Попередній захист роботи	03.06.2022	Виконано
8	Здача роботи		

Студент _____ __Мельник М.А.__

(підпис) (прізвище та ініціали)

Керівник роботи _____ __Поперешняк С.В.__

(підпис) (прізвище та ініціали)

Зміст

Вступ

Ще за початку існування шкіл та навчальних закладів, люди збиралися в невеликі групи де обговорювали між собою те, чи інше питання, де під час такої дискусії виявлялася істина і таким чином люди дізнавалися про щось нове і розвивали свій кругозір. Після чого з'явилися навчальні заклади, де знання добувалися шляхом проходження навчальної програми, і все це відбувалося в очному вигляді, куди люди приходили безпосередньо самі. Але технології не стоять на місці, тому з розвитком комп'ютерів у нас з'явилась можливість проводити навчання в дистанційній формі. Які переваги при цьому ми отримуємо?

- Зручність у процесі навчання. Учню чи студенту не потрібно витрачати час на поїздку, при цьому учень завжди буде встигати на заняття та бути в домашній атмосфері.
- Успішність. Студент може не встигати за навчальною програмою або навпаки її переганяти, і завдяки дистанційному навчанні студент може отримати навчальні матеріали та власноруч опрацювати їх. Також таким чином формується самостійність та бажання до навчання.
- Уніфікація. Завдяки тому, що багато елементів навчання перенесені в дистанційну форму, легше знаходити ту, чи іншу інформацію. Журнал оцінок, питання до викладачів або інших студентів, здача робіт та інших документів – все це знаходиться в одному місці та має легкий доступ.

Чи не важливим серед всіх цих переваг є те, що завдяки дистанційній формі навчання, процес вивчення та засвоєння матеріалу не порушується завдяки впливу зовнішніх факторів. COVID-19, погодні умови та інші фактори – все це не є завадою для студентів.

1. ОПИС ПРЕДМЕТНОЇ ГАЛУЗІ

1.1. Аналіз аналогів

1.1.1. Moodle

Moodle — це навчальна платформа, розроблена для забезпечення викладачів, адміністраторів та учнів єдиною надійною, безпечною та інтегрованою системою для створення персоналізованих навчальних середовищ [1].

Moodle напевне є однією з найпопулярніших прикладів навчальної платформи, якою користуються багато навчальних закладів. В чому її особливості?

- Простота у використанні. Завдяки написаному інтерфейсу, користувачам дуже легко орієнтуватися та налаштовувати платформу.
- Широкий інструментарій. Система завантаження файлів, побудова ієрархії ролей в системі, назначення домашніх завдань, профілі студентів, створення курсів та дисциплін.
- Адаптивність. Її можна використовувати як на мобільному пристрої, так і на ноутбуках та персональних комп’ютерах.
- Масштабованість до будь-яких розмірів. Система може працювати стабільно навіть якщо в ній зареєстровано більше ніж мільйон користувачів. Дана платформа може використовуватись як і в малих організаціях, так і в дуже великих.
- Підтримка. Даний проєкт підтримується розробниками та оновлюється час від часу. Завдяки чому завжди є можливість звернутися у разі виявлення помилок, недоліків, або якщо потрібно задати питання.

Можливостей в даній платформі багато, через що вона і стала такою популярною. Але в такої системи є і свої недоліки:

- Напівзакрите API. З технічної точки зору, закрите API не дає змогу створити власний клієнт, через що ми маємо користуватися вже готовим рішенням. Через що, можливість інтегрування його в інші застосунки, написання власного інтерфейсу довколо такого API, являється дуже важкою задачею.
- Відсутня система мотивації. Щоб заохочувати студентів до вивчення матеріалу, потрібна система мотивації. Як приклад, досягнення, рівень прогресу та ін.
- Обмеження розміру завантажувальних файлів. В сучасному світі, обмеження у вигляді максимум 10 мегабайтів виглядає дуже смішним. Тут має бути хоча б можливість вибору цього обмеження або його зменшення.

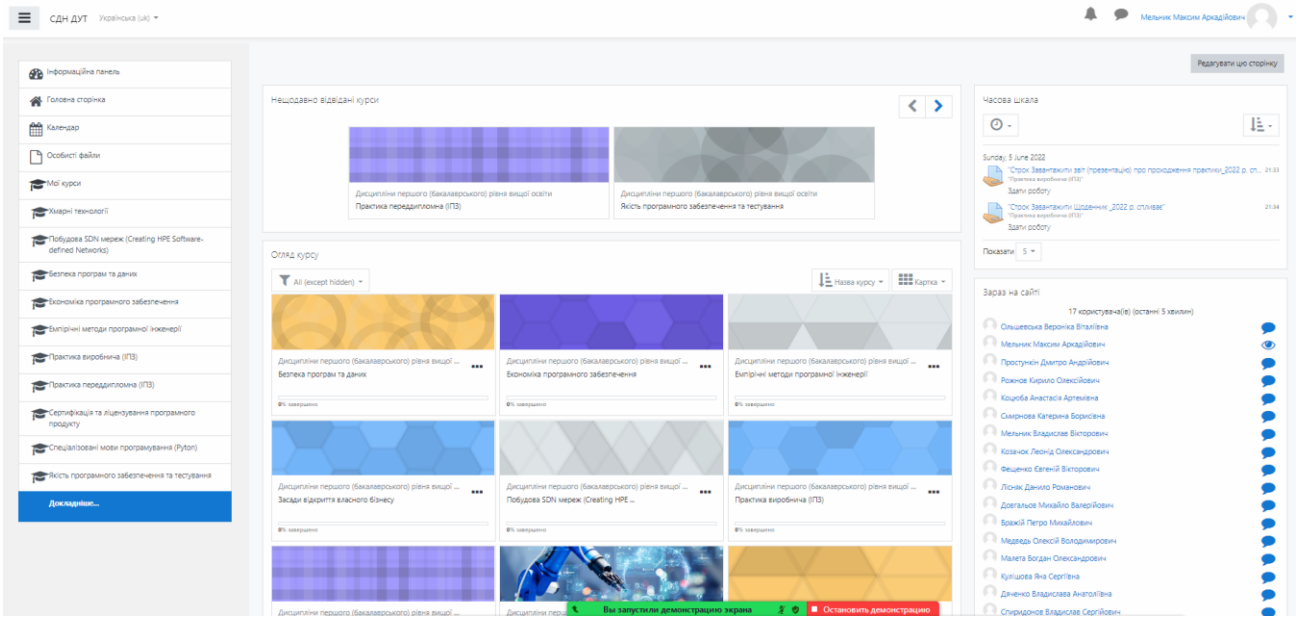


Рис. 1. – Головна сторінка платформи

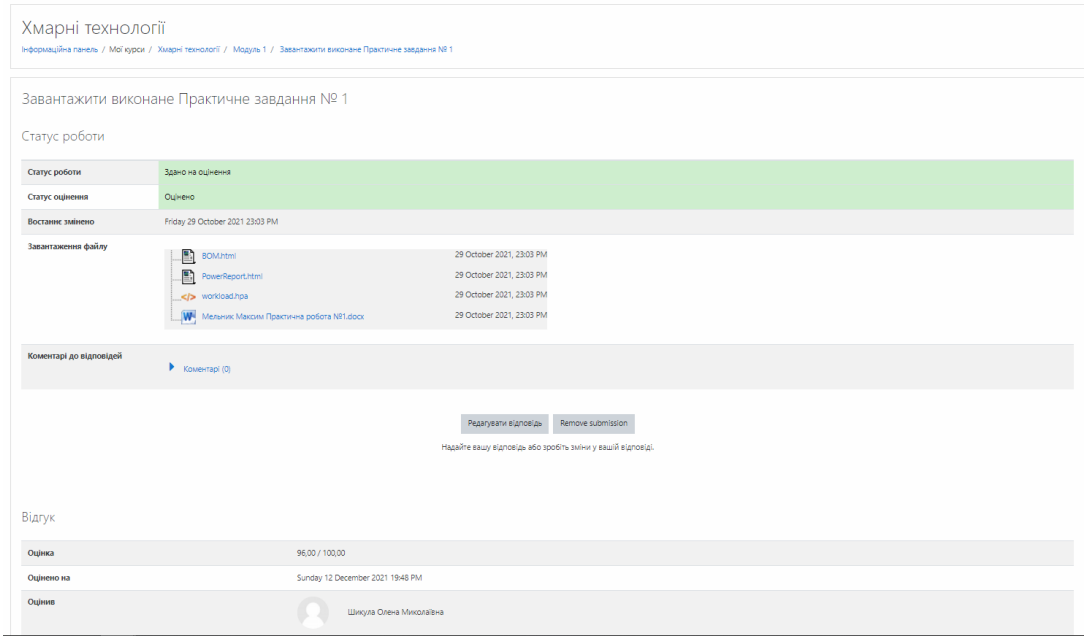


Рис. 2. – Приклад зданої роботи

1.1.2. НОВІ ЗНАННЯ

Нові знання – це портал електронних класних щоденників та журналів, яка під'єднана до системи КУРС "Школа". На ньому є можливість реєстрації, перегляду особистий щоденник, журнал знань, перегляд останніх шкільних новин, спілкування з викладачами та іншими батьками і учнями школи. Особливості такої системи:

- Вузькоспрямованість. Завдяки тому, що основна система розбита на такі частини, це дозволяє контролювати навантаженість та зробити таку систему менш централізованою.
- Легковажкість. Так, як портал виконує одну або декілька конкретних задач, він не буде навантажувати систему користувача, що дає змогу користуватися ним на багатьох платформах та пристроях.

Недоліки:

- Малозадачність. Він виникає з першого пункту особливостей такої системи. Така малозадачність змушує нас користуватися не однією системою, в якій можливо виконувати всі ці дії та мати багато інших можливостей, а одразу декілька, в кожній із яких ми маємо проходити процес аутентифікації.
- Застарілий інтерфейс. Через те, що інтерфейс не відповідає сучасним вимогам, він є неадаптивним та незручним, через що ним неможливо користуватися на деяких пристроях, навіть попри те, що через його легковажність він для них підходить.
- Нестійка до навантажень. Система має малий запас міцності, через що велика навантаженість на систему може спровокувати її повільну роботу або взагалі призупинити.

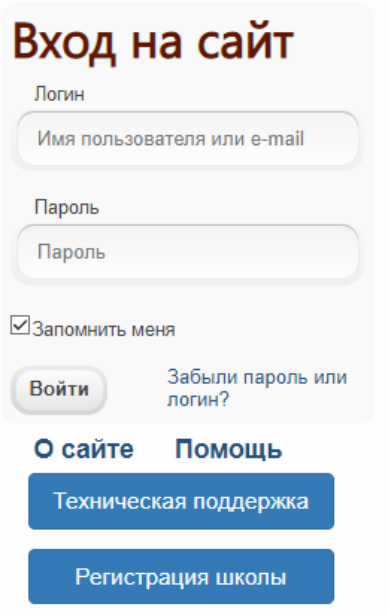


Рис. 3. – Форма входу ”-

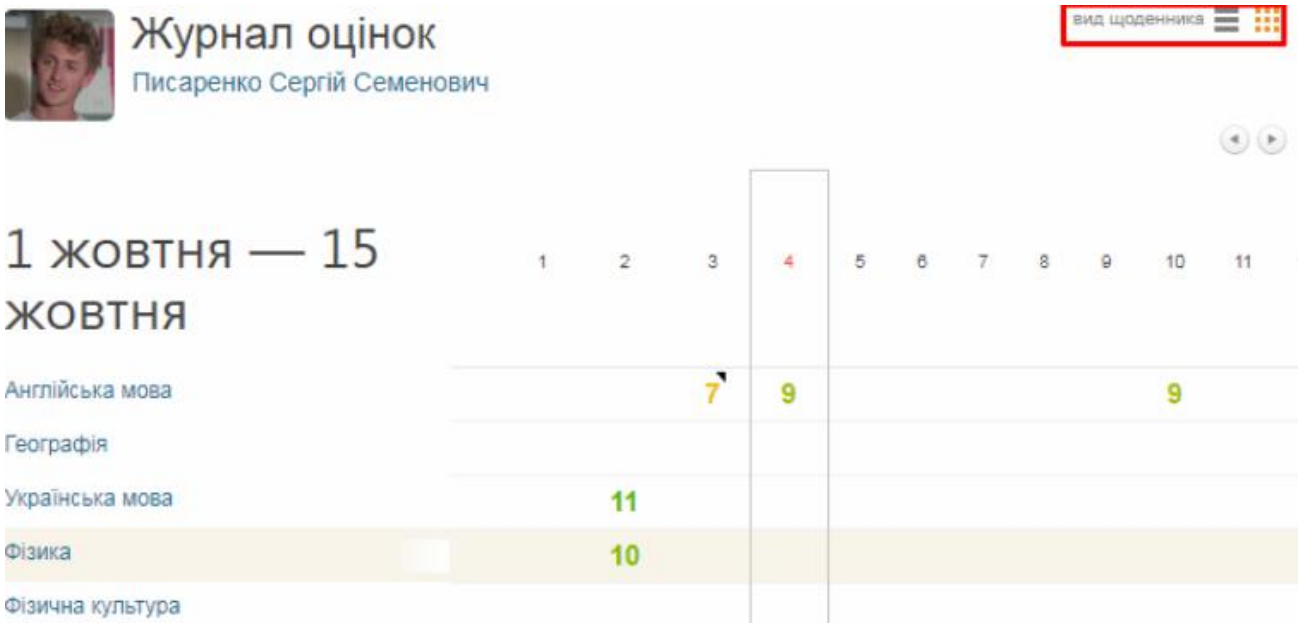


Рис. 4. – Інтерфейс системи

1.1.3. GOOGLE CLASSROOM

Google classroom – це навчальна платформа для студентів та викладачів, в якій зібрані інструменти для управління навчальним процесом між багатьма учасниками. Завдяки доступності такої платформи можна дуже легко організувати навчальний процес.

Переваги такої платформи:

- Загальний сервіс. Так, як дана платформа є однією із багатьох розробок компанії Google, тут можна скористатися вже відповідним створеним аккаунтом, що дозволяє нам не створювати кожного разу нові дані для подальшої авторизації. Також, через спільний сервіс можна буде прив’язати інші рішення даної компанії, як приклад Google Drive.
- Простота в організації навчального процесу. Виходячи з попередньої переваги маємо те, що організувати навчальний процес між учнями дуже легко – все, що потрібно це знати пошту або інші дані користувачів, та

додати їх до відповідної групи. Після чого вже в додану групу можна буде додавати домашнє завдання, файли та оцінки.

- Мультиплатформність. Завдяки адаптивності такої платформи, нею можна користуватися як з мобільного пристрою, так і персонального комп’ютера.

Недоліки:

- Малозадачність. Не дивлячись на те, що платформа дозволяє дуже легко організувати навчальний процес та має інструментарій для цього, його не достатньо. Багато чого доводиться просто завантажувати у вигляді файлів, якщо справа доходить до журналу або оцінювання. Дуже сильно не вистачає якоїсь системи сортування, яка дозволила б зручно переглядати всі доступні завдання.
- Потребує постійного доступу до інтернету. Можливо в сучасному світі це не така і велика проблема, але у разі виникнення непланових ситуацій навіть просто передивитися власні завдання не буде можливості.
- Обмеженість в сервісах. Окрім власних сервіс від компанії Google, під’єднання зовнішніх стає вже важкою задачею. Частіше за все це просто неможливо зробити або дається з великими зусиллями.

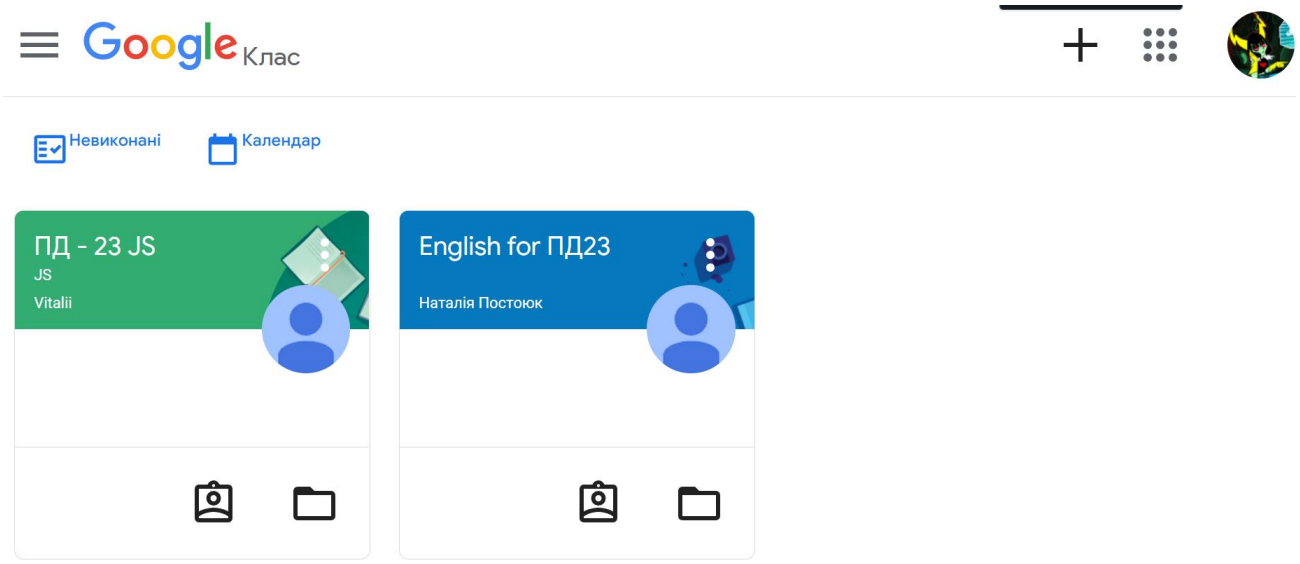


Рис. 5. – Головне меню

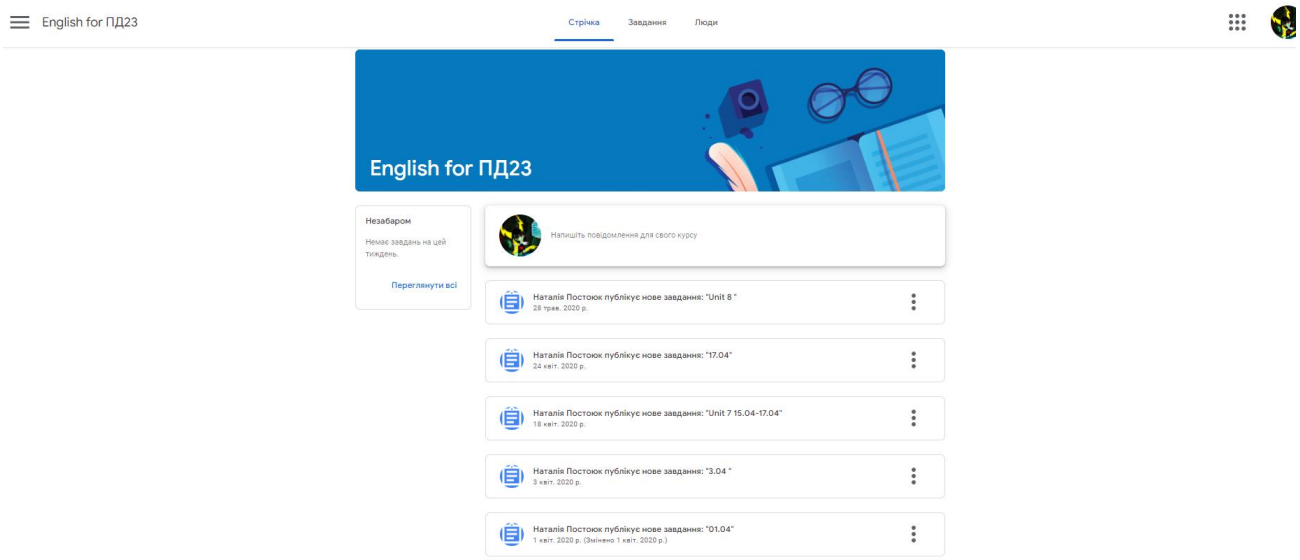


Рис. 6. – Інтерфейс системи

Не дивлячись на те, що всі системи підходять для проведення навчального процесу в дистанційних умовах, у всіх із них є свої переваги та недоліки. Якісь мають широкий функціонал, але вимогливий до системи, деякі навпаки. Але у

всіх них відсутній один елемент, який на мою думку є дуже важливим – система прогресії та мотивації. Завдяки такої маленької, але не менш важливої деталі процес навчання можна перетворити в захопливий процес.

1.2. Опис ІТ-засобів

1.2.1. Мова програмування C#

При розробці дипломного проєкту було обрану мову програмування C#. Чому саме вона? По-перше – це об’єктно-орієнтована мова програмування, що дозволяє нам створювати багатофункціональні додатки. Окрім цього, через потужну спільноту, люди створили і продовжують створювати популярні бібліотеки, які спрощують процес розробки та дозволяють користуватися вже написаним функціоналом, але під власні потреби. Як приклад таких бібліотек: AutoMapper, Bogus, FakeItEasy, FluentValidation, BCrypt та багато інших. Та й окрім додаткових бібліотек, корінева бібліотека, яка доступна напочатку, дає нам тисячі утіліт, структур даних, методів та багато чого іншого.

Важливо також згадати і особливості ООП – це успадкування, що дозволяє нам створювати так звані “ контракти ” та їх реалізацію, що спрощує написання коду та розв’язує нам руки в плані редагування його в майбутньому, що теж величезний плюс, адже в будь якому випадку це доведеться робити. Система класів, яка дозволяє розміщувати логіку в одному місці і за потреби успадковувати її для інших класів. Багато модифікаторів доступу, строга типізація та інше. Ще однією великою перевагою є те, що C# відноситься до C – подібних мов, із за чого люди, які програмували до цього на C або C++ зможуть легше перейти на C#.

Рис. 7. – Мова програмування C#

1.2.2. MySQL

Наступним на що було звернуто увагу при розробці це вибір бази даних. Серед всіх доступних на ринку, MySQL напевне найпростіша в використані. Але не потрібно думати, що простота це завжди менша функціональність. Перша її перевага це відкритий код. Завдяки спільноті, яка зацікавлена в даній БД, її функціонал потрохи розширюється і доповнюється, тому вона на даний момент не є застарілою, а отже її використання не несе за собою типові проблеми застарілих БД. Також вихідні коди сервера компілюються на багатьох платформах. Найповніше можливості сервера виявляються в UNIX-системах, де є підтримка багатоканальності, що підвищує продуктивність системи в цілому.

В даному проєкті важливим є підтримка швидкодії та ефективності БД, адже кількість студенів та даних може бути високою, все це присутнє в MySQL. Також з нею не виникне проблем при використанні в веб-додатках, адже процес її під’єднання дуже простий. Ще однією вагомою перевагою є те, що сама БД працює на різних платформах і не має проблем з переносом даних.

Рис. 8. – База даних MySQL

1.2.3. JetBrains Rider

Не менш важливим був вибір IDE, так як в майбутньому це нам допоможе швидше писати та аналізувати код. Величезною перевагою саме цього програмного середовища є те, що в ньому є дуже потужний інструментарій відлагодження програми. Завдяки ньому в подальшому буде набагато легше розроблювати програмне забезпечення, через те, що помилки та баги будуть виявлятися на ранніх етапах проєкту, а навіть, якщо баг виявився досить пізно, цей інструментарій дозволить відстежити всі етапи, на яких даний баг з'являвся.

Сама компанія JetBrains розроблює свої рішення не тільки для мови програмування C#, але і Java, Python, C, C++ та інші. В самому IDE JetBrains Rider є вбудований інструмент для підключення пакетів – NuGet, що дає змогу швидко знайти та підключити потрібний пакет або бібліотеку, замість того, щоб шукати в інтернеті його найновішу версію, опис та строку підключення. Також, вбудований звіт продуктивності дозволить виявити слабкі сторони програми, виявити на якому етапі продуктивність падає та знайти можливе рішення. Як приклад в даному дипломному проєкті була проблема з тим, що запит в БД

займав дуже велику кількість часу, поки не стало зрозуміло, що випадково було задіяно логіку іншої БД, яка і спричинила уповільнення. Тож вибір IDE – це теж одне із важливих питань при розробці програмного забезпечення.

Рис. 9. – Середовище розробки JetBrains Rider

1.2.4. DBeaver

DBeaver – це інструмент для роботи з такими популярними базами даних як MySQL, PostgreSQL, SQLite, Oracle, DB2, SQL Server, Sybase, MS Access, Teradata, Firebird, Apache Hive, Phoenix, Presto та інші. Так, як не завжди в базах даних вбудований зручний інтерфейс, ми можемо скористатися подібним рішенням. Якщо казати простими словами, то DBeaver – це вже написаний розробниками інтерфейс, через який ми безпосередньо керуємо під'єднаними БД. Коли ми в цьому середовищі будуємо наприклад таблицю, насправді генерується запит, який відповідає мові БД та виконується, за рахунок чого в нашій БД з'являється таблиця. Так, як це запит, ми можемо його редагувати, якщо нам не подобається як він згенерувався.

З ним також зручно працювати, якщо ви маєте керувати одразу декількома базами даних, так як список підтримуваних БД великий, і є можливість одночасно працювати с багатьма. В цьому ж проєкті було

використано базу MySQL та Azure blob storage. Хоча і Azure Blob storage не доступний в DBeaver, але він допоміг з MySQL, так як в ній була доступна лише консоль.

Рис. 10. – Інструмент для роботи з БД DBeaver

1.2.5. Azure blob storage

Azure blob storage - це рішення корпорації Майкрософт для зберігання об'єктів у хмарі. Сховище BLOB-об'єктів оптимізовано для зберігання великих обсягів неструктурованих даних, наприклад, текстових або двійкових даних.[2] Задля того, аби зменшити навантаження на основну базу даних та розбити місця для зберігання файлів та фотографій, було вирішено використовувати дане сховище. По-перше це дозволить нам розділити логіку і не зачіпляти основну базу. По-друге, для роботи з такими файлами в Azure blob storage підходить більше тому, що файли зберігаються в спеціальному вигляді, що дозволяє швидше записувати та витягувати дані. В порівнянні зі звичайною базою, сховище Azure буде працювати швидше саме для таких типів об’єктів.

Також не потрібно забувати про шифрування та налаштування самих даних. На сайті або окремому додатку можна налаштувати як саме буде

відбуватися перевірка, рівень доступності, мінімальну версію протокола та багато чого іншого. Ось приклад таких налаштувань через офіційний веб-сайт:

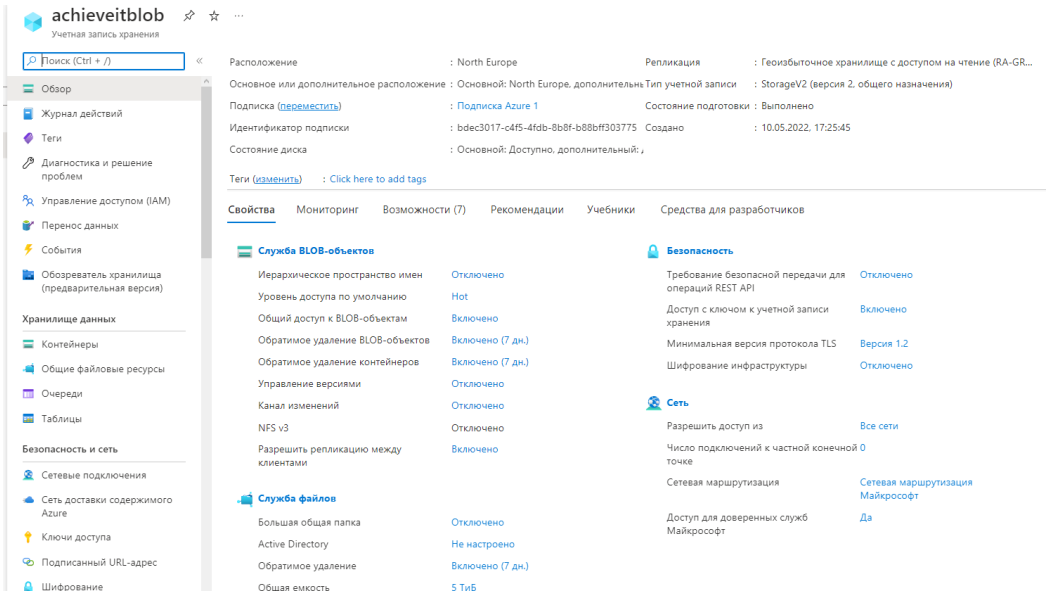


Рис. 11. – Параметри налаштувань Azure blob storage

1.2.6. Postman

Postman – це програма для тестування функціональної частини API. З його допомогою можна тестувати відповіді, отримувані з серверу або API. Таким чином можна дізнатися, як реагує програма на ті, чи інші вхідні дані, чи передбачено розробниками валідація помилок при їх виникненні, що ми отримуємо та чи задовольняє це описаним вимогам. Також при довготривалому тестуванні дуже незручно будувати запит для отримання відповіді від серверу, особливо коли на сервері ввімкнена автентифікація і потрібно до кожного запиту додавати токен. Тут вбудовані інструменти, історія запитів, редагування вже створених та багато чого іншого. Через це, ми витрачаємо менше часу і

будуємо один стиль для всіх запитів, які потрібні для розробників, які займаються Front-end частиною проєкту.

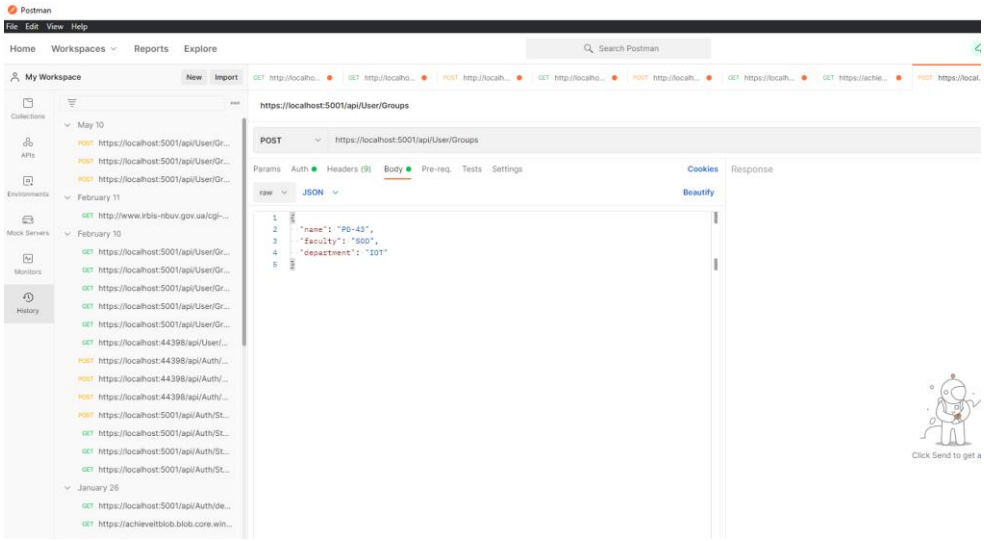


Рис. 12. – Приклад інтерфейсу та використання Postman

Рис. 13. – Програма для тестування API Postman

1.2.7. Git

Git – це відкрита і безкоштовна система контролю версію, створений для керування проєктами та їхніми версіями. З його допомогою ми маємо можливість завантажувати проєкти, комітити зміни, проглядати їх, змінювати, а також кооперуватися з іншими користувачами для створення спільного

програмного забезпечення. Напевно без контролю версія неможливо нормально розроблювати будь-який проєкт, адже якщо потрібно буде повернутися до минулої версії програми, або продивитися зміни, які були внесені на тому чи іншому етапі, буде дуже проблематично. Git завдяки своїм можливостям дозволяє відстежувати всі ці зміни та навіть позначати ким ці зміни були внесені. Також, є можливість кооперацій з іншими сервісами. Як приклад, на Trello є можливість прикріпити посилання на коміт, який одразу відобразиться відповідним чином.

Ще однією такою кооперацією можна назвати з вище вже вказаним JetBrains Rider, де є можливість продивитися список віток та комітів, переключитись на відповідну вітку, а також через термінал виконувати команди для git bash. Завдяки взаємозв'язку між трьома цими сервісами, можна створити зручну систему для командної розробки, або навіть для розробки наодинці, адже всі можливості в сумі дають зручність та зрозумілість цілісності проєкту, де в будь-якому місці можна переглянути зміни, виявити недоліки, повернутися до певної версії, зробити позначки на дошці Trello. Саме такий принцип був задіяний в цьому дипломному проєкті.

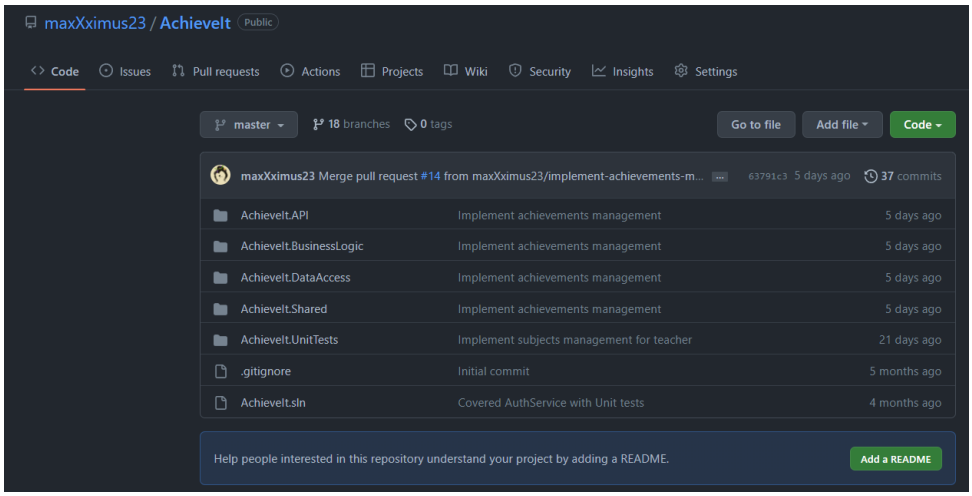


Рис. 14. – Приклад роботи в Git -

Рис. 15. – Система контролю версій Git

1.2.8. Trello

Trello – це інтерактивна дошка, на якій можна розміщувати карточки з описами завдання для керування проєктом та команди. Завдяки такій дошці між командой можна створити зв'язок та розподілити ролі. Один займається тестуванням, інший розробкою, третій взагалі інтерфейсом, і за допомогою готових рішень є можливість позначати карточки із завданнями тегами, кольором, додатками та багато чим іншим. Таким чином команда не плутається у завданнях та знає, які ще не готові, а які, можливо, потрібно переробити.

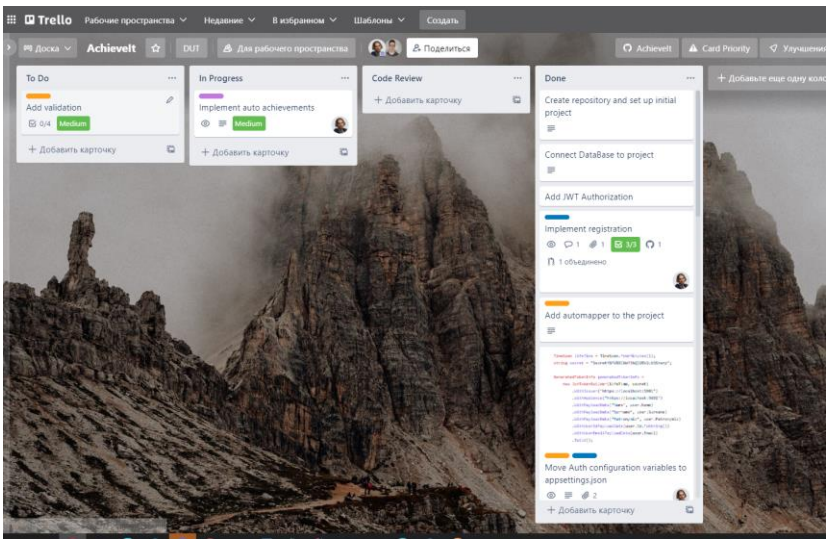


Рис. 16. – Приклад організованої інтерактивної дошки Trello

Рис. 17. – Інтерактивна дошка Trello

1.3. Визначення об’єкта, предмета, мети та задачі

Мета роботи – підвищення мотивації студентів при процесі дистанційного навчання завдяки впровадженню елементів гейміфікації.

Об’єкт дослідження – процес дистанційного навчання.

Предмет дослідження - web-додаток веб-порталу Університету.

Задачі дослідження:

- проаналізувати сучасні аналоги
- виявити недоліки систем дистанційного навчання
- спроектувати ієрархію ролей в системі
- реалізувати систему авторизації та автентифікації
- вирішити головну мету дослідження
- побудувати кінцевий проєкт з усіма врахуваннями.

2. РОЗРОБКА СТРУКТУРИ WEB-ДОДАТКУ « ВЕБ-ПОРТАЛУ УНІВЕРСИТЕТУ »

2.1. Завдання web-додатку « Веб-порталу Університету »

Навчання завжди було і залишається одним із етапів в житті людини. Починаючи від шкільного віку, людині потрібно відвідувати навчальні заклади та здобувати знання. Але все частіше робити це стає складніше в силу стихійних лих та інших загроз, які заважають процесу навчання. З цього моменту все популярнішим та розповсюдженим стає система дистанційного навчання. Які цілі та завдання в першу чергу ставить перед собою дана система:

1. досягти максимальної зручності для студентів та зробити процес навчання безперервним;
2. введення у процес навчання нових освітніх технологій та створення за допомогою цього сучасного освітнього простору;
3. поступовий перехід від репродуктивного навчання до більш сучасного: креативного;
4. стимулювання самостійної пошукової роботи учнів, що спрямовується викладачами.

В дипломному проєкті було реалізовано по можливості всі ці завдання, але головною метою цього проєкту було підвищення мотивації студентів при процесі дистанційного навчання завдяки впровадженню елементів гейміфікації. Які саме це елементи: в першу чергу – це система досягнень, які надаються студентові при виконанні певних вимог. Як приклад, завантаживши 5 домашніх завдань, студент отримує досягнення “ Новачок ” та відображається на його особистій сторінці. Таким чином у студента з’являється додаткова мотивація до

вивчення нових знань та з’являється деяке змагання між студентами, у кого таких досягнень більше. Завдяки цьому в навчальному процесі можна організувати систему гри, в якій будуть задіяні всі студенти та видавати нагороди за те, чи інше досягнення. Як приклад, можна організовувати поїздки або екскурсії, нагороди для самих завзятих студентів, або зробити список, куди входили б студенти з найбільшою кількістю нагород.

2.2. Моделювання об’єкту проєктування

2.2.1. Діаграма прецедентів системи

Головна мета такої діаграми – показати, який функціонал програми доступний кожному її учаснику. У даному проєкті присутні 3 учасники:

- Студент – представляє собою звичайного студента або учня та має доступ до незначної частини системи;
- Вчитель – відповідно представляє вчителя або педагога, який вже має ширший набір доступу, але все ще обмежений у своїх можливостях;
- Адмін – має повний доступ до елементів системи, які присутні для редагування та зміни даних. Створений для налаштування груп та завдань, досягнень, можливостей реєстрації користувачів, редагування їх даних, видалення і т.д.

На діаграмі нижче показано як саме взаємодіють актори з системою;

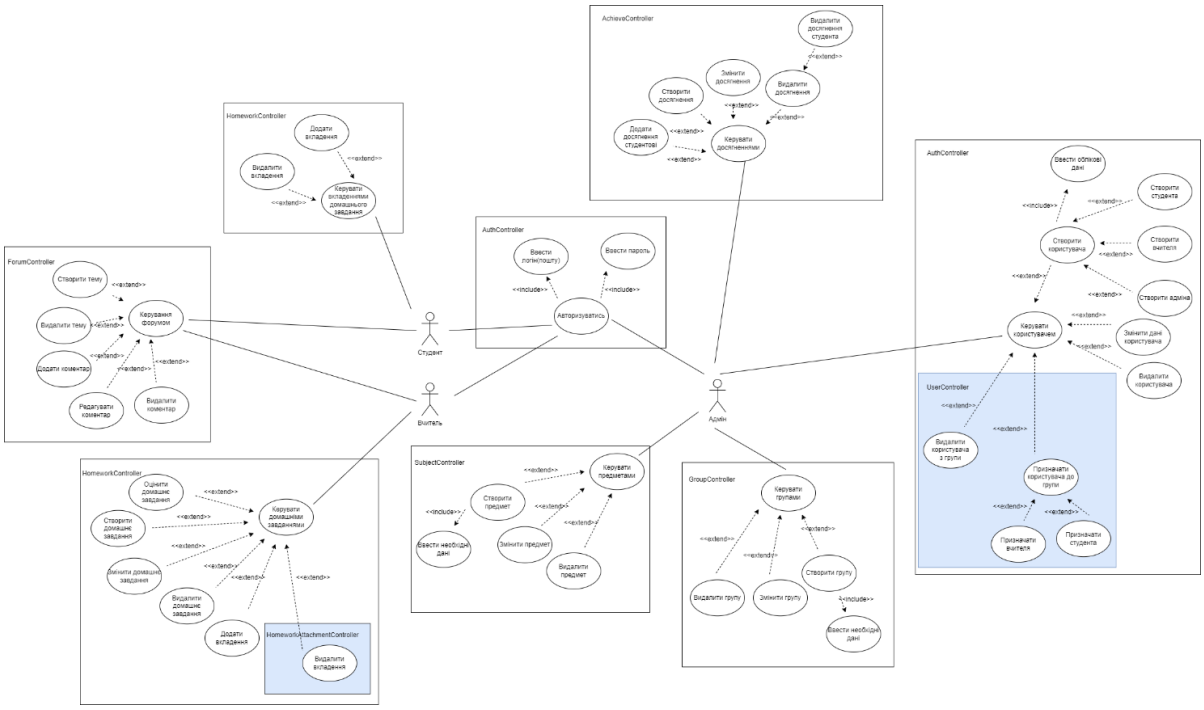


Рис. 18. – UML Діаграма прецедентів

Перед тим, як користувач отримає доступ то тих, чи інших можливостей, йому потрібно зареєструватися та авторизуватися в системі. Після чого йому надається відповідна роль та можливості. По основним місцям проєкту розташовані перевірки, які не дають користувачу, який не відповідає ролі в програмі використовувати функції не його рівня. Завдяки цьому студент не зможе підправити собі оцінки в журналі або змінити зміст завдання.

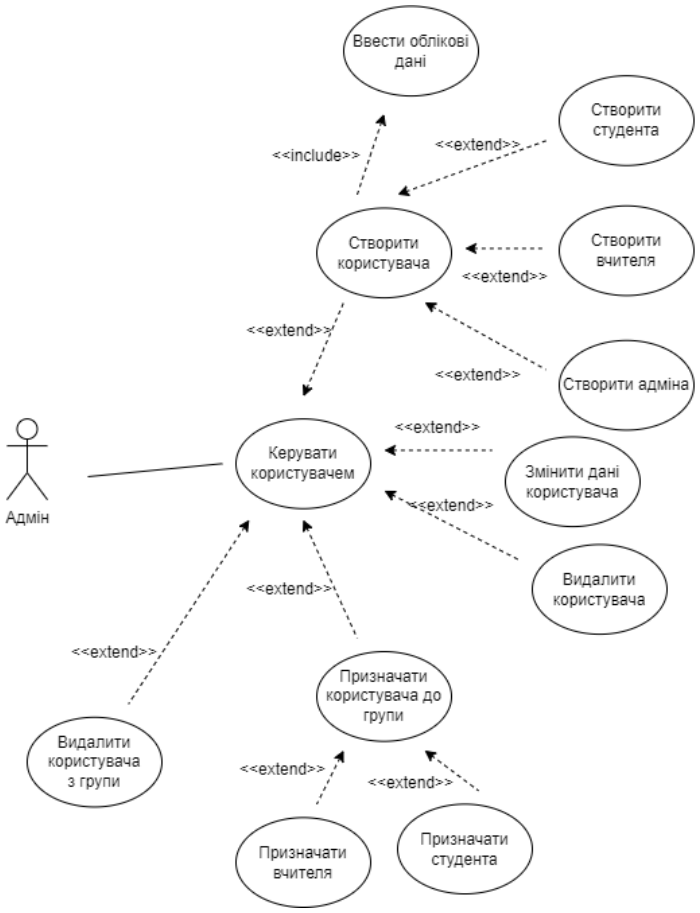


Рис. 19. – UML Діаграма прецедентів функції керування користувачами

2.2.2. Діаграма класів

Діаграма класів – призначена для формування логічної моделі програми на рівні класів. Вона дає розуміння, з яких класів складається система та як між собою вони взаємодіють та з чого складаються.

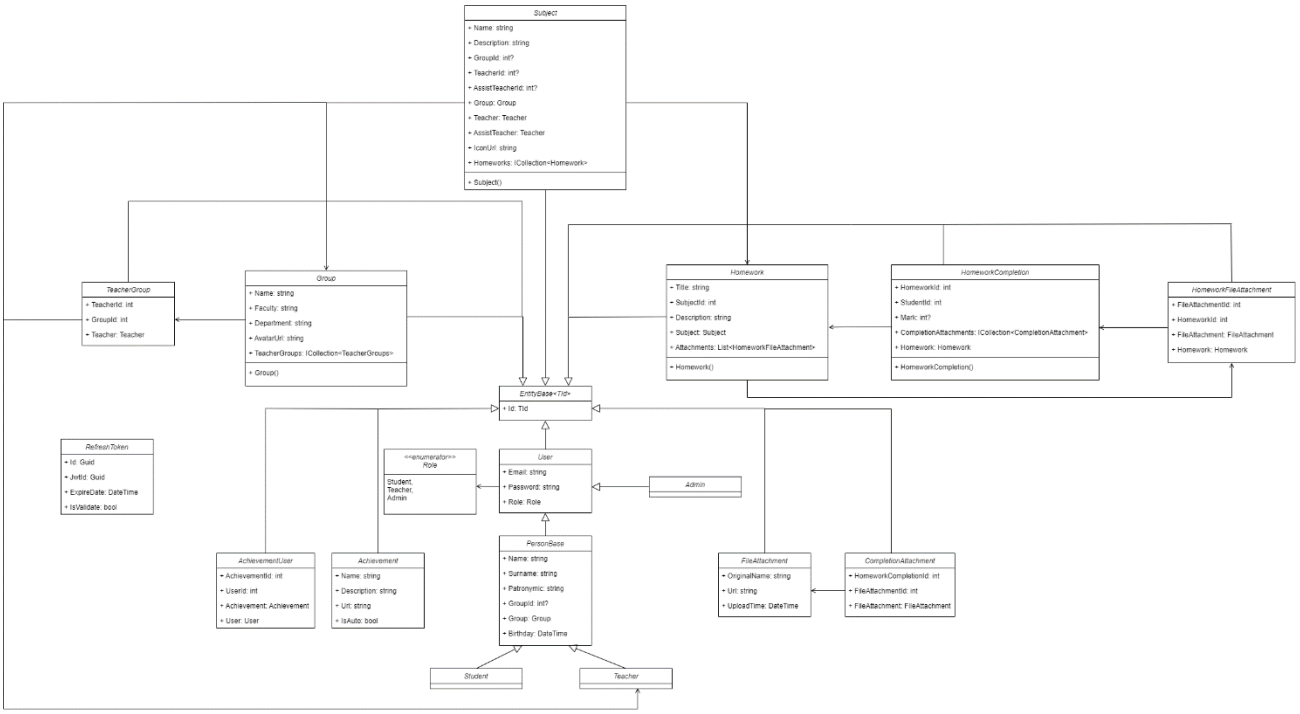


Рис. 20. – UML Діаграма класів

На діаграмі вище представлено структура класів дипломного проєкту. Клас EntityBase представляє собою абстрактний клас, який має одне поле Id та використовується для наслідування в інших класах. Клас, який унаслідується від EntityBase визначає, якого типу даних буде у нього поле Id і таким чином можна контролювати для кожного класу тип даних для поля Id. Також на діаграмі представлено основні класи User та PersonBase, в яких знаходяться основні поля кожного користувача: пошта, ПІБ, група, пароль та ін.

Завдяки такому поділу, ми можемо в будь який момент змінити те, чи інше поле, при цьому не міняючи інші класи. Це дозволить нам краще розширювати функціонал програми.

2.2.3. Використання баз даних

При розробці веб-додатка було застосовано дві бази даних: MySQL та Azure blob storage. Перша використовується для зберігання даних користувачів,

груп, досягнень, зв’язних таблиць та ін. Друга застосовується для зберігання BLOB – файлів, адже для цього в ній є багато інструментів та можливостей. Завдяки Azure blob storage ми зможемо генерувати посилання на файл та по цьому посиланню або переглядати або завантажувати файли. Це дуже сильно полегшить при розробці Fron-end частини проєкту.

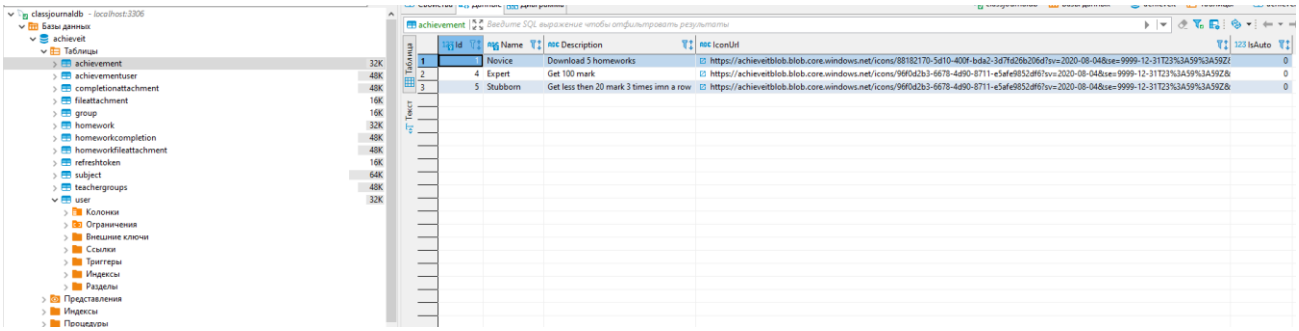


Рис. 21. – Приклад таблиць в базі даних MySQL

Як ми бачимо с рисунка вище, в базі даних MySQL розміщені таблиці achievement, в якій відповідно знаходиться інформація про досягнення та поле з посиланням на його іконку, яке ми отримуємо завдяки іншої бази Azure blob storage, на якій вона і розміщується. Якщо б ми зберігали наш файл в MySQL нам потрібно було б завжди завантажувати цілий файл одразу с бази, а так ми отримуємо посилання, за яким ми вже можемо перейти та переглянути його, або завантажити. По-перше, цей спосіб дозволить нам менше навантажувати базу MySQL, а по-друге отримувати зручний спосіб керування файлами. І це не тільки отримання вже готового посилання, але й переглядати дату та час коли був завантажений файл, можливість відновлення файлів (бекап). Також, так як ми генеруємо це посилання на базі, ми можемо вказати скільки це посилання буде дійсним, максимальна кількість зчитування цього файлу.

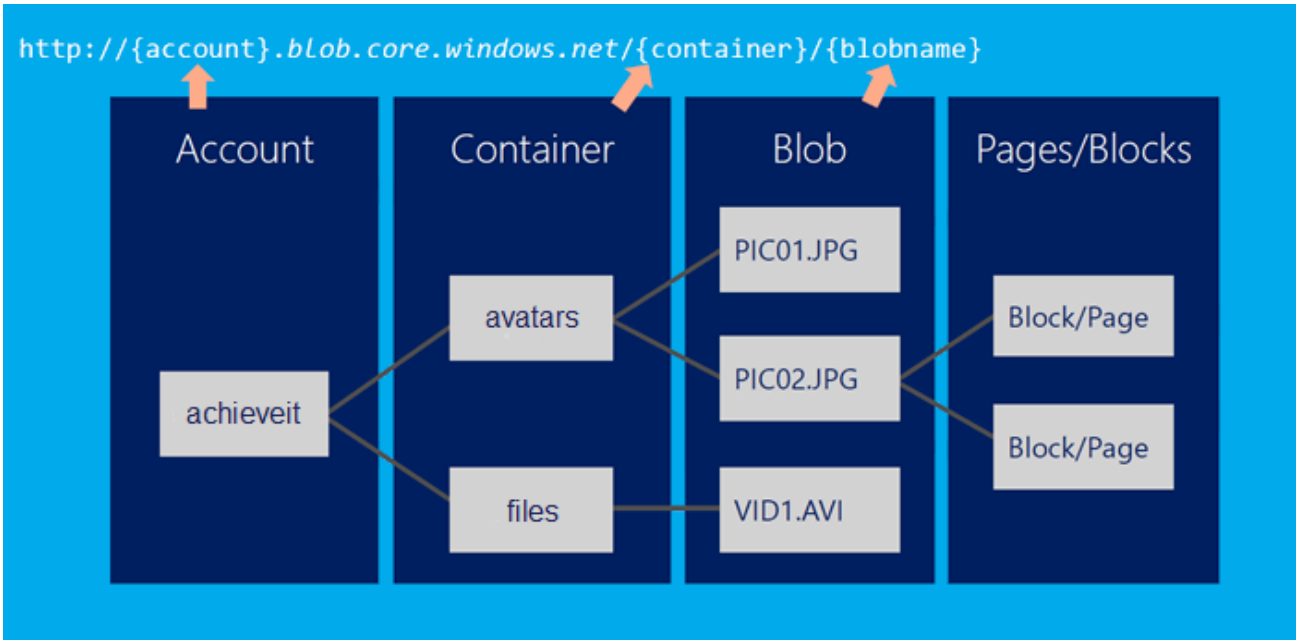


Рис. 22. – Приклад побудови посилання на файл в Azure blob storage

2.3. Конфігурація та налаштування додатку

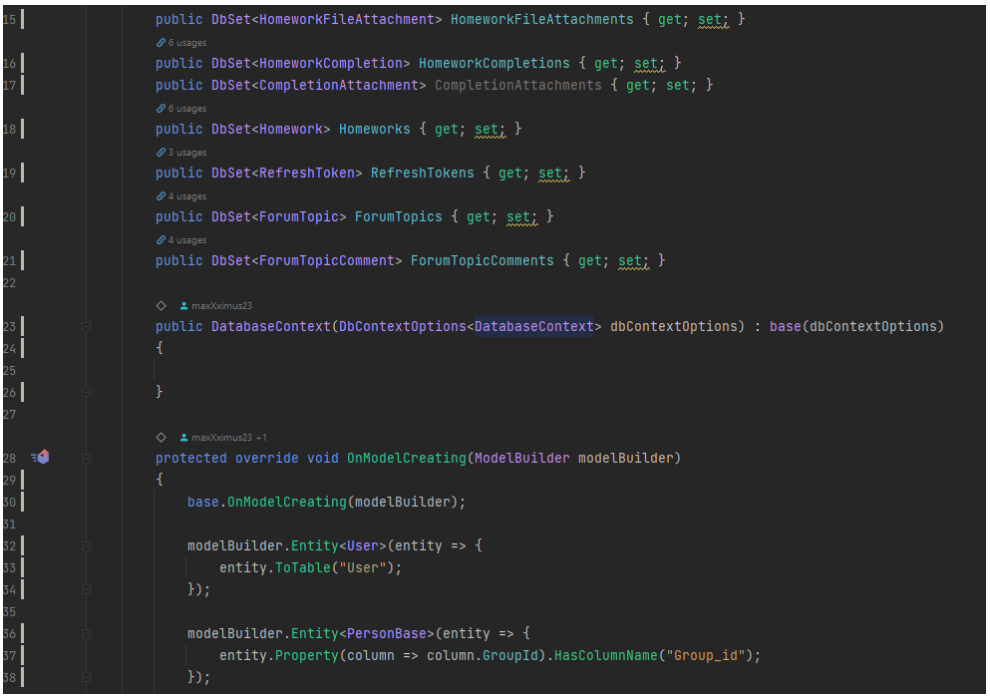
2.3.1. Налаштування appsettings

Appsettings слугує для конфігурації налаштувань елементів програми. Через нього ми можемо виставити термін життя токену, вказати строку підключення для баз даних, налаштувати логування та ін. Завдяки цьому, не потрібно в самій програмі в різних місцях налаштовувати кожний елемент окремо, всі налаштування знаходяться в одному місці. В самому файлі не виставляються значення до тих пір, поки вся система не буде розгортуватись, тобто всі критично необхідні значення, які не варто знати звичайному користувачеві або розробнику можна приховати. Таким чином можна одну й ту саму систему розгорнути в багатьох місцях з різними значеннями в appsetting. Як приклад це змінити строку підключення до баз, або змінити налаштування токену.

Рис. 23. – Налаштування appsettings

2.3.2. Налаштування DatabaseContext

В цьому класі ми описуємо таблиці, які знаходяться в нашій базі даних MySQL. В подальшому ми будемо таким чином звертатися до цих таблиць та виконувати запити. Щоб Entity Framework розумів в яку саме таблицю в БД звертатися, ми маємо вказати в налаштуваннях назву таблиці в БД.



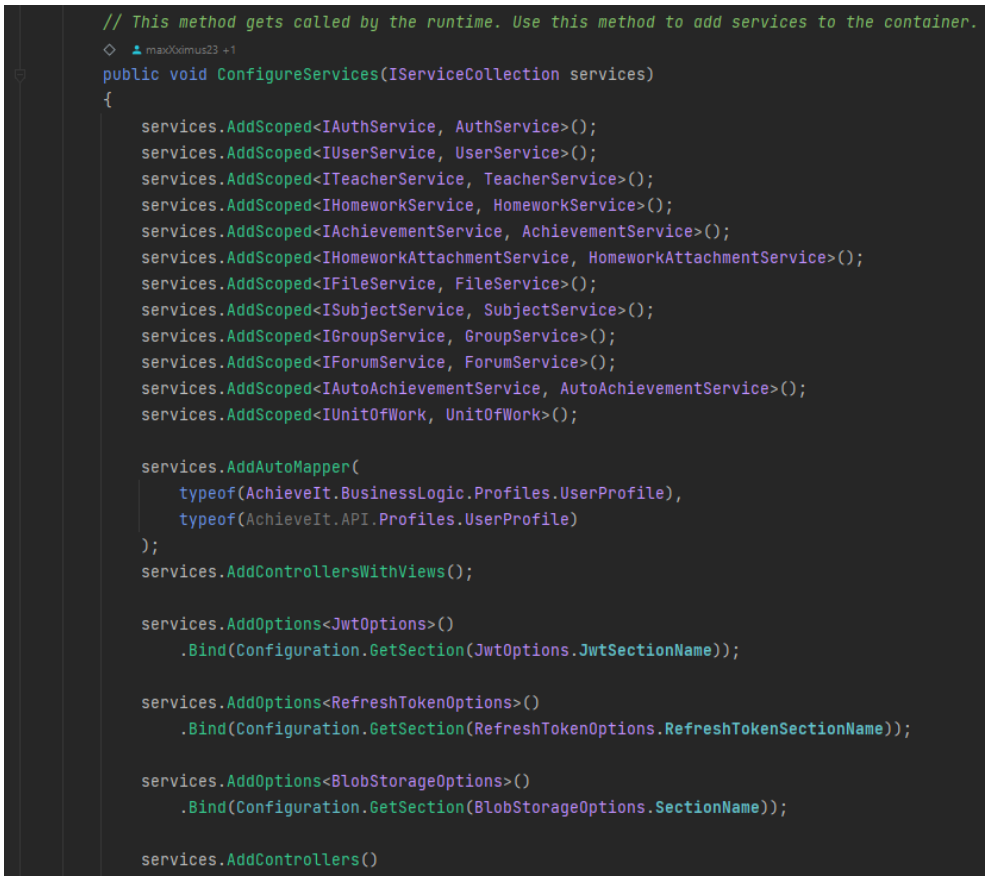
```
public DbSet<HomeworkFileAttachment> HomeworkFileAttachments { get; set; }
6 usages
public DbSet<HomeworkCompletion> HomeworkCompletions { get; set; }
public DbSet<CompletionAttachment> CompletionAttachments { get; set; }
6 usages
public DbSet<Homework> Homeworks { get; set; }
3 usages
public DbSet<RefreshToken> RefreshTokens { get; set; }
4 usages
public DbSet<ForumTopic> ForumTopics { get; set; }
4 usages
public DbSet<ForumTopicComment> ForumTopicComments { get; set; }

maxXximus23
public DatabaseContext(DbContextOptions<DatabaseContext> dbContextOptions) : base(dbContextOptions)
{

}

maxXximus23 +1
protected override void OnModelCreating(ModelBuilder modelBuilder)
{
    base.OnModelCreating(modelBuilder);

    modelBuilder.Entity<User>(entity => {
        entity.ToTable("User");
    });

    modelBuilder.Entity<PersonBase>(entity => {
        entity.Property(column => column.GroupId).HasColumnName("Group_id");
    });
```

Рис. 24. – Налаштування DatabaseContext

На рисунку вище зображено які таблиці присутні в базі даних і з якими ми будемо в подальшому взаємодіяти в програмі. В методі OnModelCreating ми позначаємо як таблиця називається в БД, щоб Entity Framework зв’язав її з відповідним entity в коді. Завдяки цьому ми можемо назвати entity будь яким ім’ям та зв’язати з потрібною нам таблицею. Окрім цього в цьому методі можна подібним чином налаштувати поля.

2.3.3. Конфігурація Startup

Startup призначений для конфігурації Middleware Pipeline та контейнер залежностей. Завдяки цьому, ми можемо прив’язати контракт з його реалізацією. Тому нам не потрібно постійно створювати екземпляри, за нас це зробить Entity Framework. До того ж тут є можливість конфігурувати контролери, токен, логування та інше.



```
// This method gets called by the runtime. Use this method to add services to the container.
◇  maxXximus23 +1
public void ConfigureServices(IServiceCollection services)
{
    services.AddScoped<IAuthService, AuthService>();
    services.AddScoped<IUserService, UserService>();
    services.AddScoped<ITeacherService, TeacherService>();
    services.AddScoped<IHomeworkService, HomeworkService>();
    services.AddScoped<IAchievementService, AchievementService>();
    services.AddScoped<IHomeworkAttachmentService, HomeworkAttachmentService>();
    services.AddScoped<IFileService, FileService>();
    services.AddScoped<ISubjectService, SubjectService>();
    services.AddScoped<IGroupService, GroupService>();
    services.AddScoped<IForumService, ForumService>();
    services.AddScoped<IAutoAchievementService, AutoAchievementService>();
    services.AddScoped<IUnitOfWork, UnitOfWork>();

    services.AddAutoMapper(
        typeof(AchieveIt.BusinessLogic.Profiles.UserProfile),
        typeof(AchieveIt.API.Profiles.UserProfile)
    );
    services.AddControllersWithViews();

    services.AddOptions<JwtOptions>()
        .Bind(Configuration.GetSection(JwtOptions.JwtSectionName));

    services.AddOptions<RefreshTokenOptions>()
        .Bind(Configuration.GetSection(RefreshTokenOptions.RefreshTokenSectionName));

    services.AddOptions<BlobStorageOptions>()
        .Bind(Configuration.GetSection(BlobStorageOptions.SectionName));

    services.AddControllers()
```

Рис. 25. – Конфігурація Startup

2.4. Структура додатку

Web-додаток взаємодіє з користувачем через прошарок Controller, який приймає Http – запити та вже передає дані в наступний прошарок Service. Він в свою чергу формує модель та передає в Repository, де формується модель Entity та вже дані звідси подаються в базу. Таким чином у нас з’являться можливість віддавати ту модель, яка нам потрібна на стороні клієнта, при цьому маючи можливість формувати її будь яким зручним для нас чином. Якщо нам потрібно приймати від користувача пароль або пошту типу string, але у самої сутності (Entity) поля не відповідають своїм типам, нам не потрібно буде змінювати цю сутність, достатньо буде приймати дані з моделі та перемапити їх в Entity.

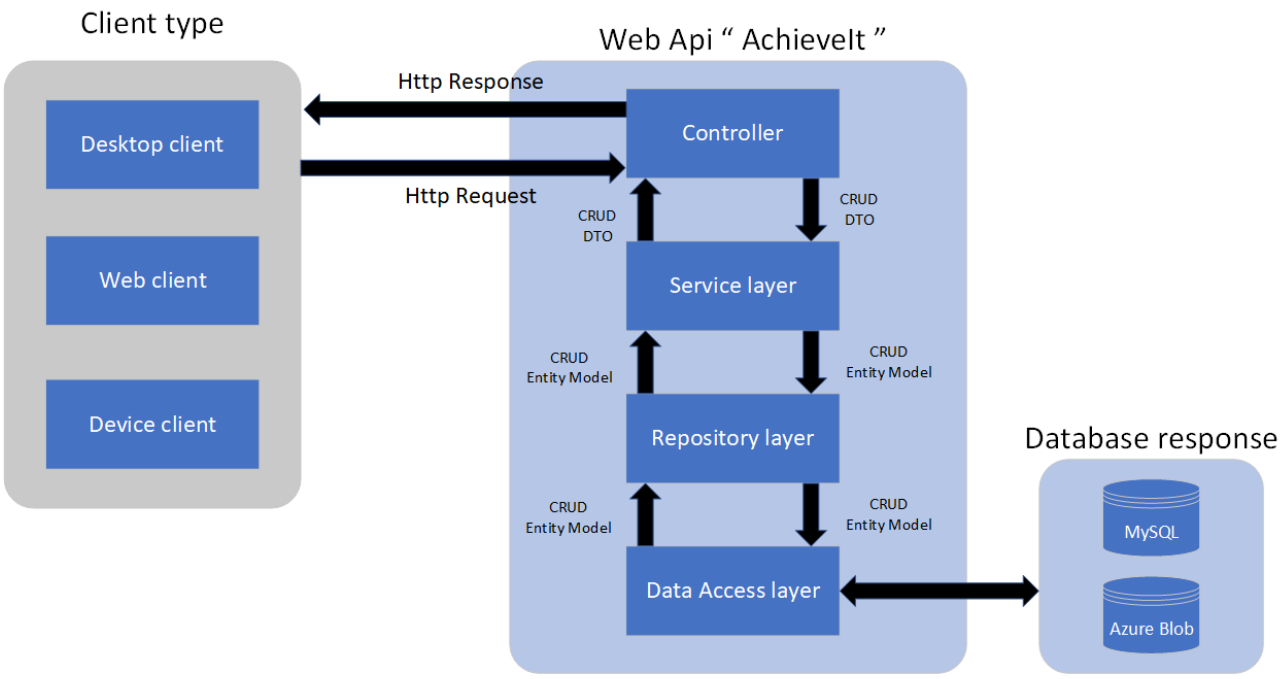


Рис. 26. – Взаємодія клієнта з Web API “ AchieveIt “

Після запиту клієнта, з серверу приходить відповідь відповідній до того, який саме запит це був. Якщо це дія створення (акаунту, досягнення, домашнього завдання, тощо), сервер має повернути або 200 OK, або 204 No Content. У випадку, коли на стороні сервера щось пішло не так (помилка запиту, відсутність значень в БД, помилка в алгоритмі) повертається код помилки та її текст, щоб користувачу було зрозуміло, що пішло не так.

3. РОЗРОБКА СТРУКТУРИ WEB-ДОДАТКУ « ВЕБ-ПОРТАЛУ УНІВЕРСИТЕТУ »

3.1. Використання IT-засобів

Задля того, аби контролювати весь процес розробки, всі задачі та описи були розміщені у спеціальному засобі – Trello. Завдяки цьому можна побачити коли і на якому етапі було розроблено, описано або виконано те, чи інше завдання. При цьому у нас є можливість під кожним таким завданням робити опис того, що вимагається, та залишати корисні посилання.

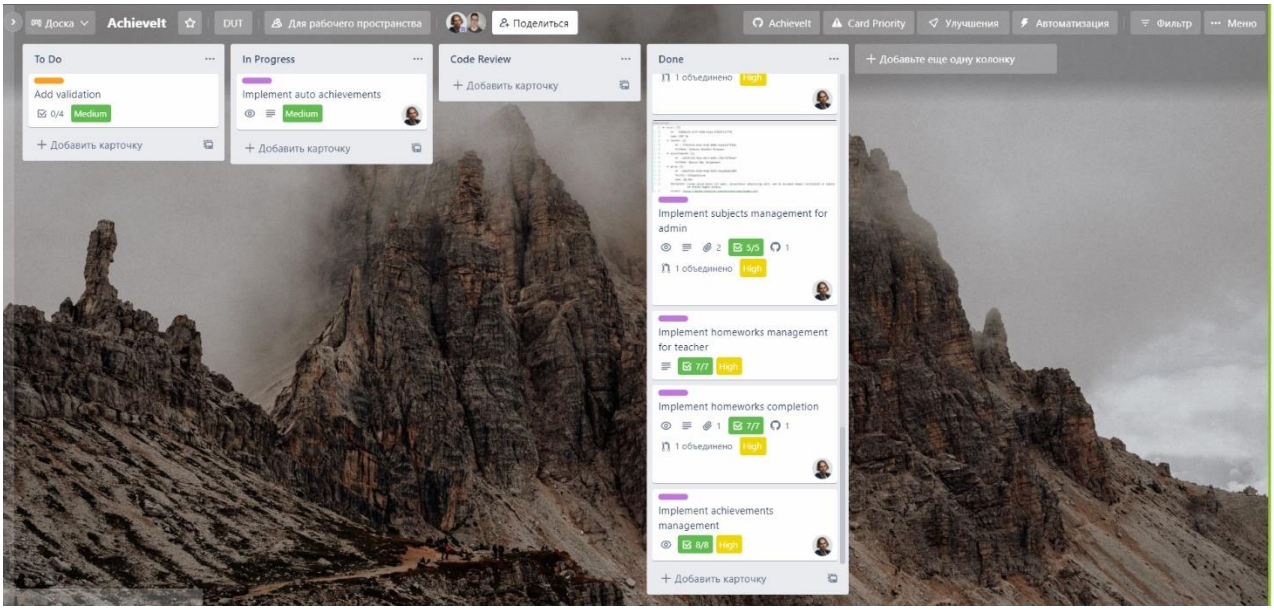


Рис. 27. – Використання віртуальної дошки Trello

Опис під кожним завданням дає нам розуміння того, що має бути виконано та які поставлені вимоги. В дипломному проєкту дуже часто описувались вимоги до ендпоінтів, та як вони мають виглядати. Завдяки такому підходу можна

досягти розуміння з Front-end розробником, адже йому важливо знати як саме вони мають виглядати.

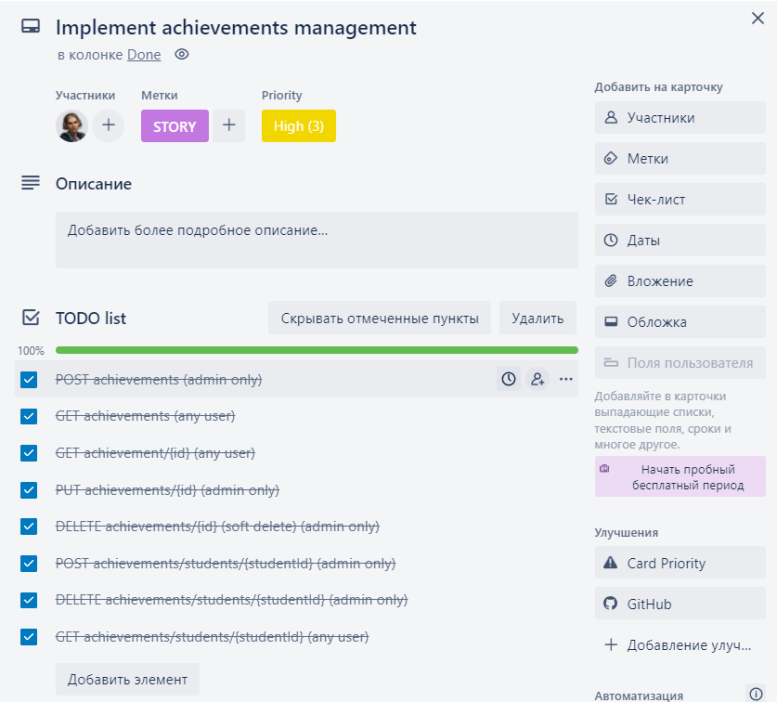


Рис. 28. – Приклад опису завдання для ендпоінтів на дошці Trello

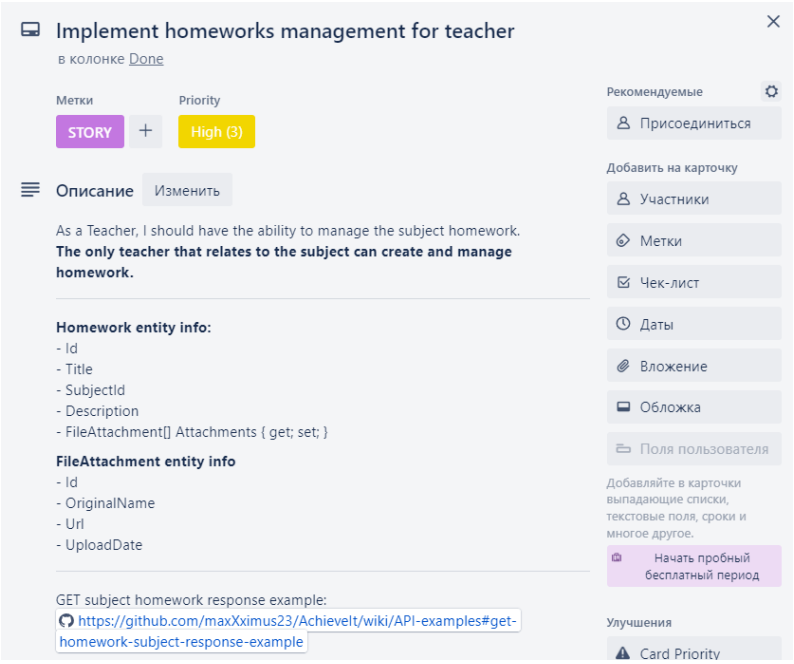


Рис. 29. – Приклад опису завдання для Entity на дошці Trello

Для успішного контролю між версіями програми та етапами розробки, було вирішено розмістити проєкт на Git репозиторії. Це дозволить нам переглядати зміни, зроблені кожним із розробників та у разі виникнення критичної ситуації відкотити ті, чи інші зміни. При цьому є можливість обговорення виконаного завдання методом перегляду коду, зробленим учасником, та позначення помилок у ньому. Якщо один із учасників виявить помилку, він може на це вказати та заборонити внесення цих змін в основну вітку проєкта, що дозволяє зберегти захищеність від побідних ситуацій.

Рис. 30. – Pull request на додавання змін до основного проєкту з їх описом

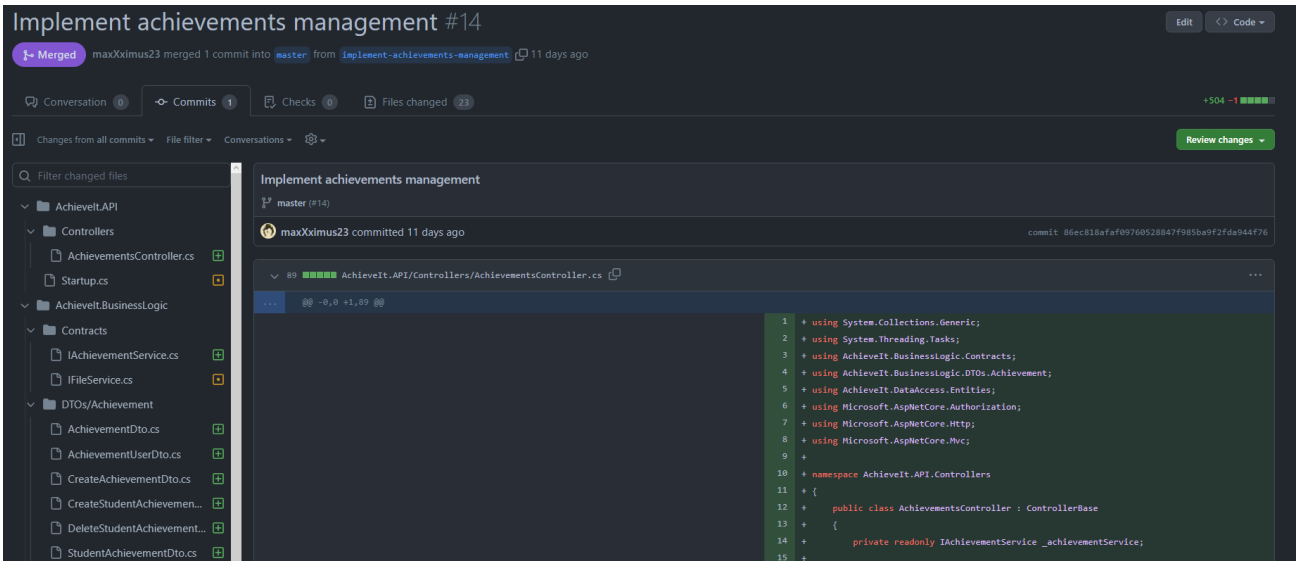


Рис. 31. – Перегляд коду та змін в pull request

Зручний інтерфейс дозволяє нам зрозуміти в яких місцях коду були зміни та зручно переглядати їх. Також, напроти цих змін можна додати коментар, в якому буде вказівка на помилку або можливе покращення коду.

Для перегляду та взаємодії з БД було обрано інструмент DBeaver. З його допомогою є можливість підключення до багатьох сучасних БД. Завдяки ньому з’являється можливість взаємодії з БД через зручний інтерфейс, який дозволяє редагувати запити, змінювати таблиці, одночасно взаємодіяти з декількома базами даних та ін. Через це, навіть якщо в БД немає свого інтерфейсу або присутня тільки консоль, DBeaver – гарне рішення таких проблем.

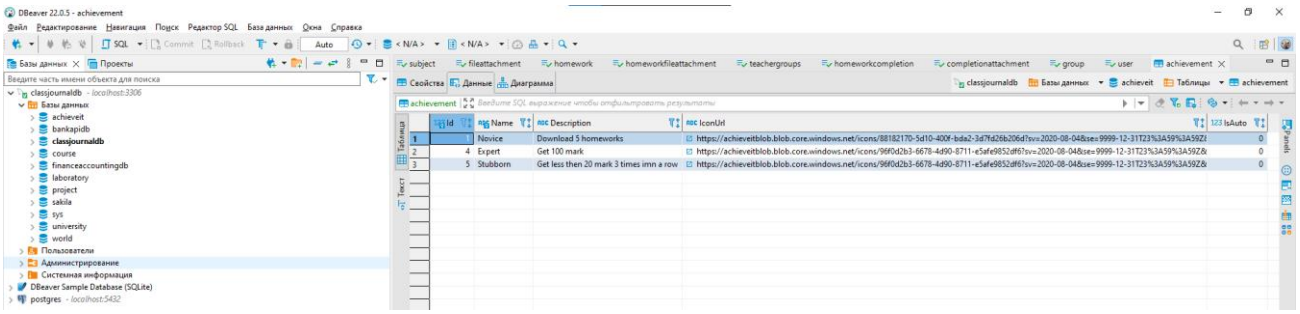


Рис. 32. – Перегляд коду та змін в pull request

Головним середовищем розробки було обрано JetBrains Rider. Через його переваги в тестуванні коду, виявленні попередніх помилок, підказках, дозволяє нам легше писати та тестувати код а також підключати сторонні бібліотеки. Основні переваги та можливості цього IDE:

- Зручний інтерфейс
- Можливість взаємодії з NuGet пакетами
- Великий набір розширення та конфігурації
- Просунута система відлагодження
- Інтеграція із системою контролю версій Git
- Постійне оновлення та доповнення середовища розробки
- Кросплатформність
- Можливість інтеграції з багатьма трекерами проблем, як приклад Team Foundation Server, JIRA Software
- Рішення і проєкти повністю сумісні з середовищем розробки Visual Studio.

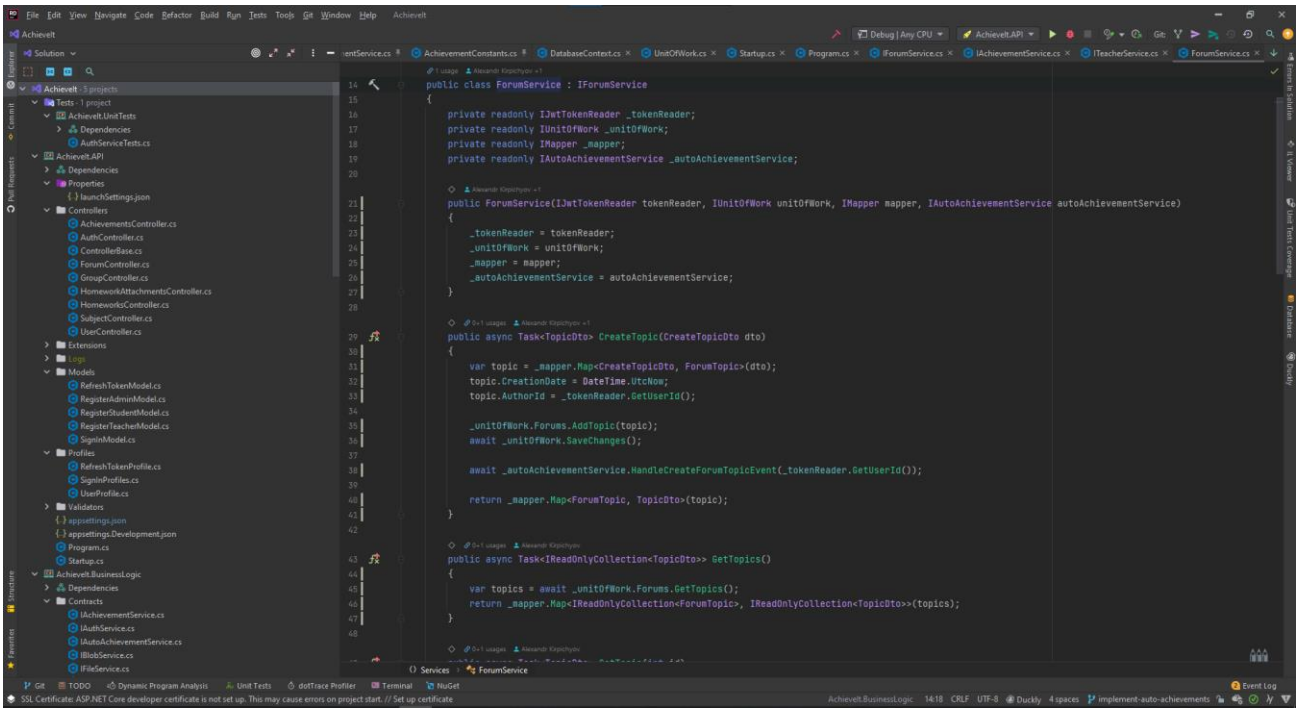


Рис. 33. – Середовище розробки JetBrains Rider

Для тестування ендпоінтів проєкту було обрано Postman. Його інтерфейс дозволяє зручно будувати запити та зберігати їх для повторного використання. Це дозволяє не писати кожного разу один і той самий запит, особливо коли вимагається токен для доступу до того, чи іншого ендпоінту.

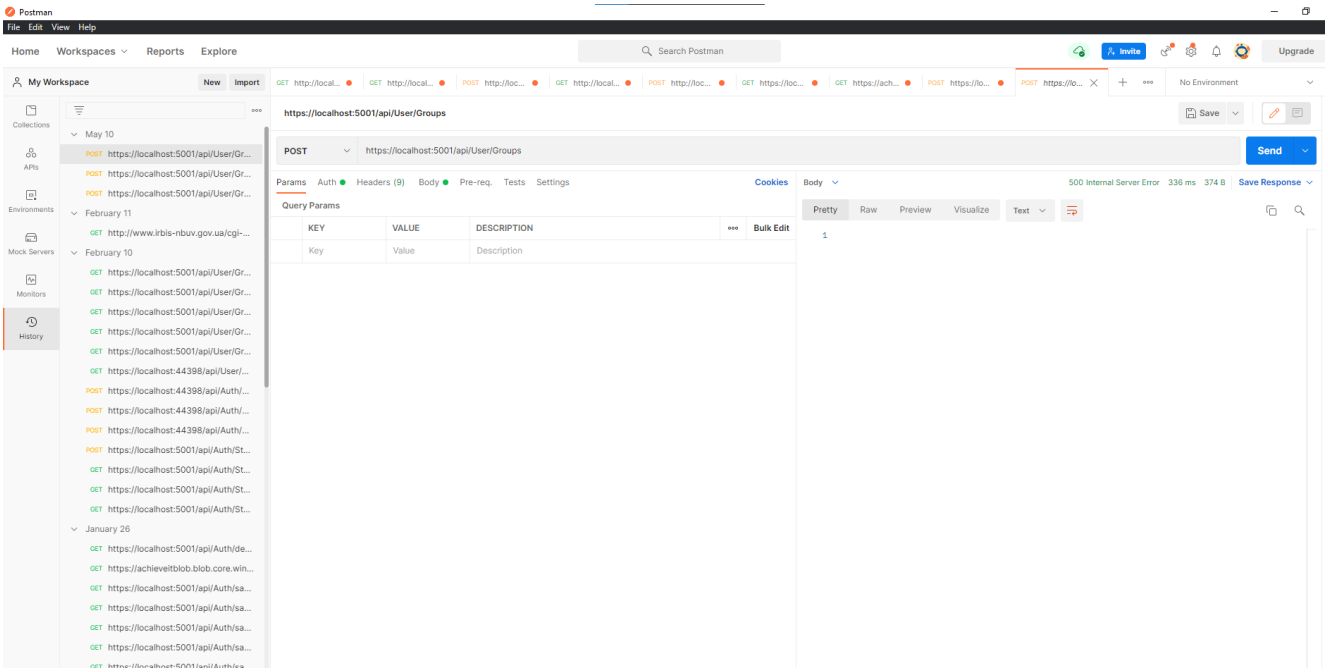


Рис. 34. – Програма для тестування API Postman

4. ПРИКЛАДИ ВИКОРИСТАННЯ ТА ТЕСТУВАННЯ СИСТЕМИ

4.1. Опис та приклад роботи системи

4.1.1. Авторизація користувача

Для того, щоб користувач мав доступ до власних можливостей в системі йому потрібно авторизуватися в системі. Для цього йому потрібно ввести пошту та пароль. До цього моменту користувача має зареєструвати адмін.

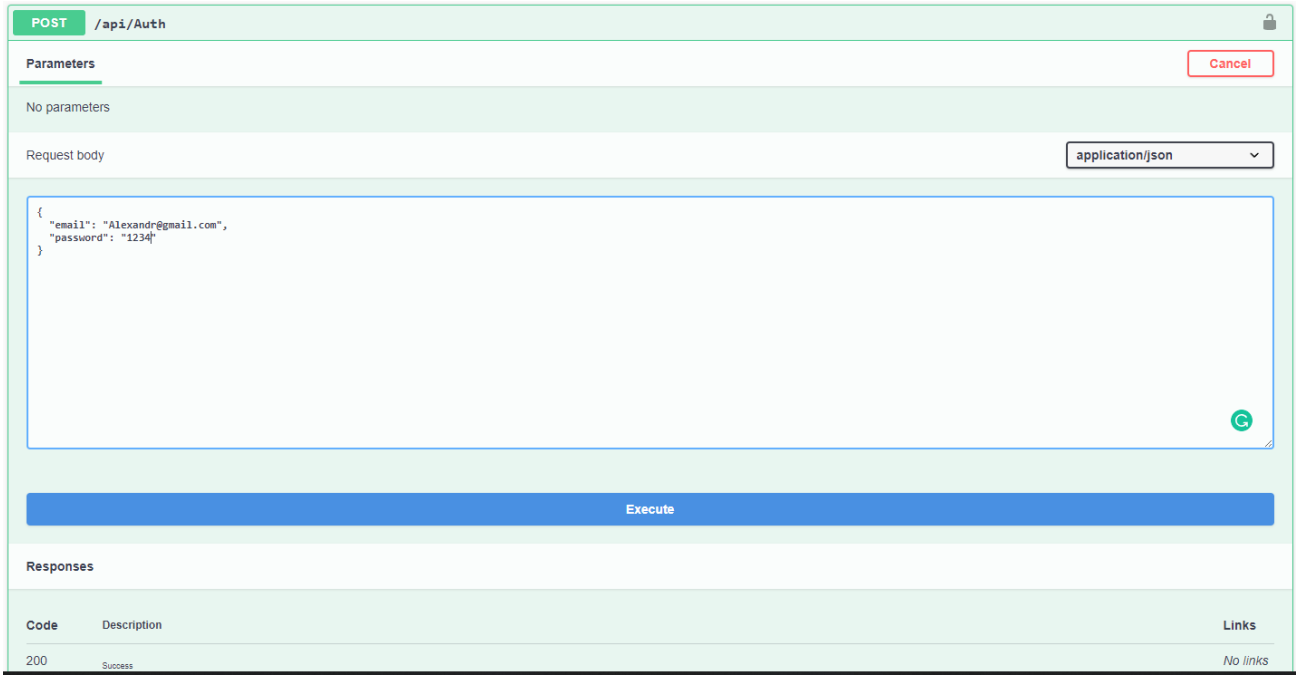


Рис. 35. – Ендпоінт для авторизації

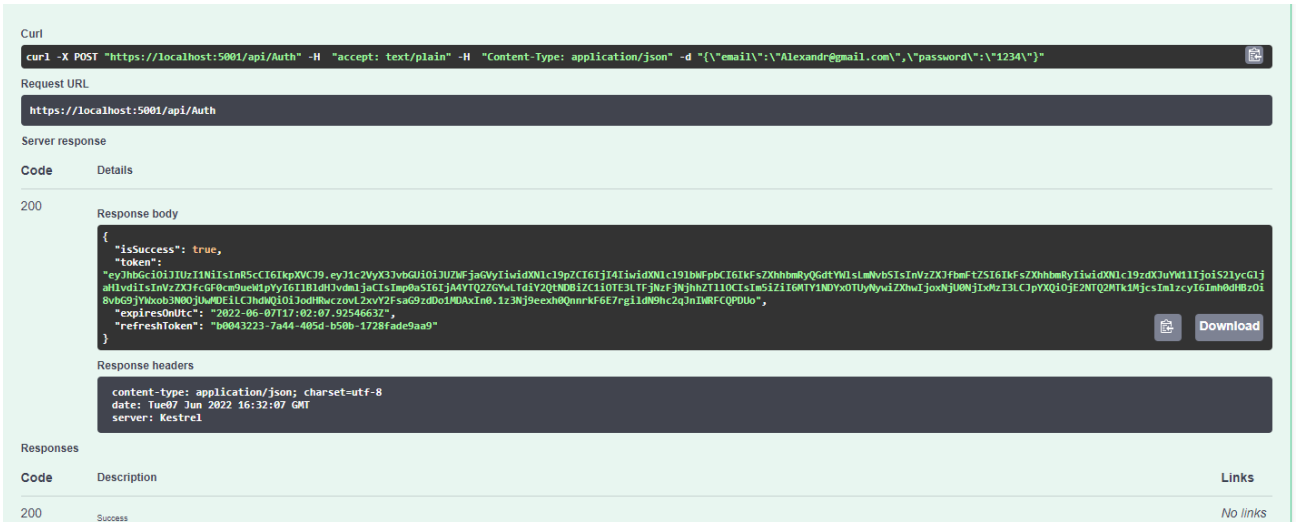


Рис. 36. – Відповідь серверу

Після успішної авторизації, сервер повертає відповідь, в якому знаходиться токен, завдяки якому користувач зможе переходити по іншим ендпоінтам. У самому токені знаходиться його роль та інші дані, за якими сервер перевіряє чи може користувач виконувати ті, чи інші дії. У разі неуспішної авторизації сервер повертає наступне:

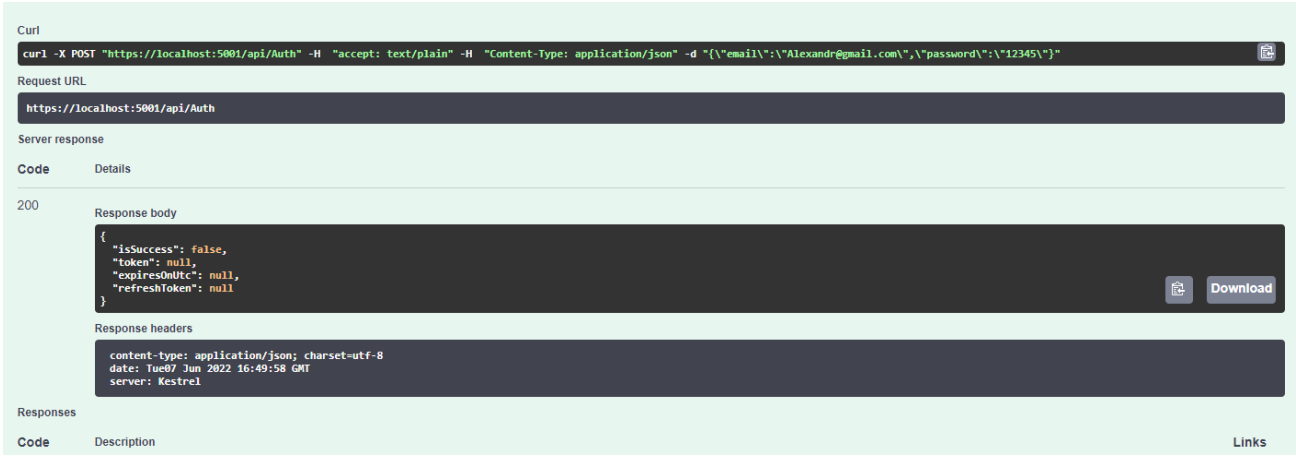


Рис. 37. – Відповідь серверу при введені некоректних даних

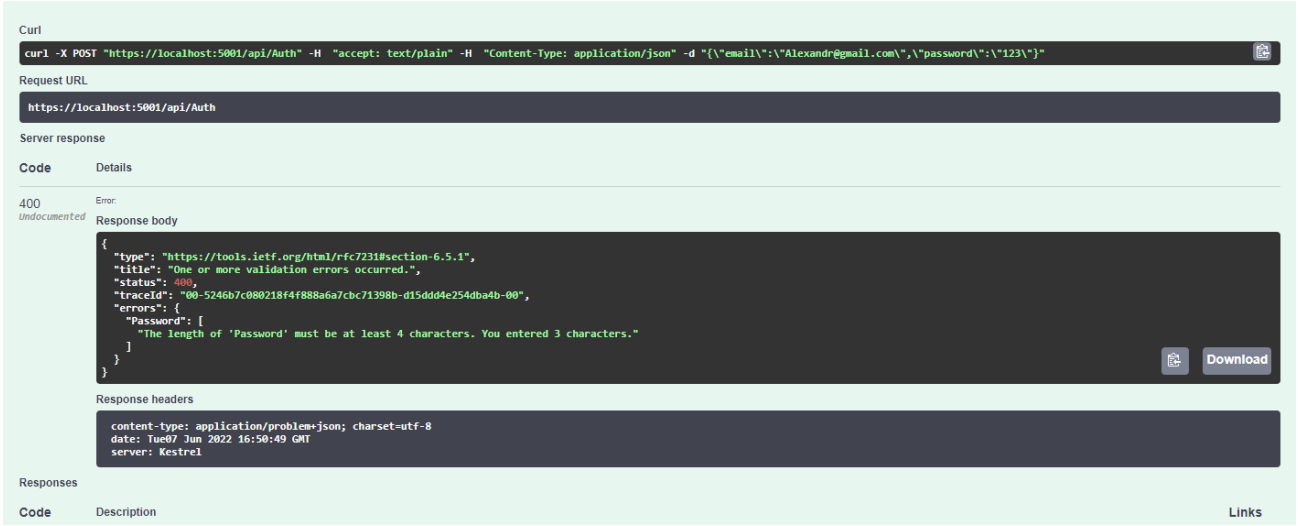


Рис. 38. – Відповідь серверу при введені даних, які не пройшли валідацію (в даному випадку пароль)

4.1.2. Додавання студента в систему

Перед тим, як додати студента в систему, адміну потрібно авторизуватися в систему. Надалі цей пункт при подальшому огляді програми буде опущено, так як процедура однотипна для всіх користувачів та ендпоінтів. Тому після авторизації отримуємо відповідь серверу з відповідним токеном:

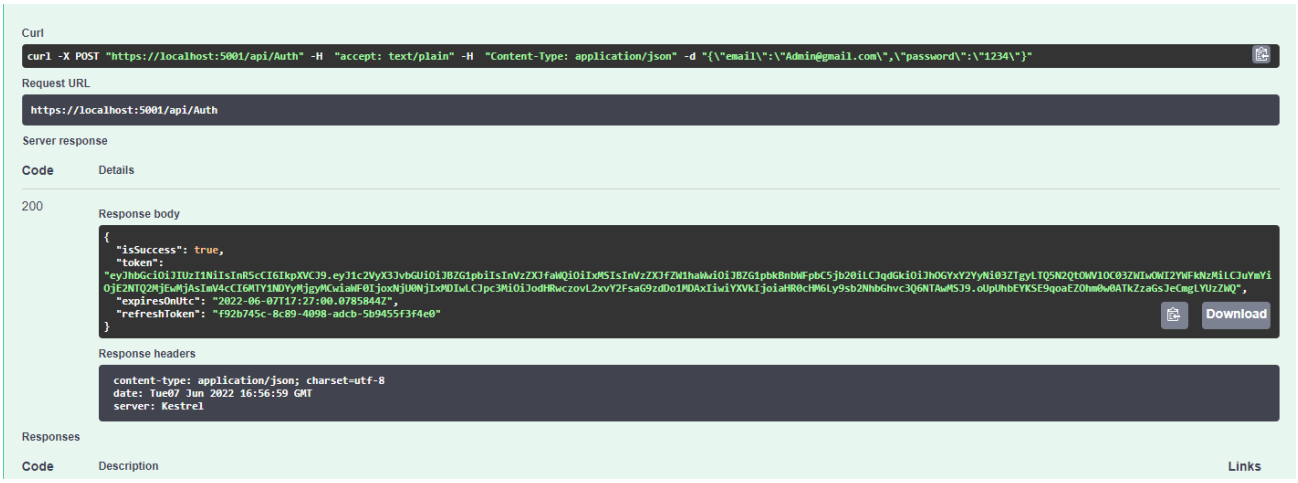


Рис. 39. – Відповідь серверу при авторизації адміна

Після його авторизації, він має доступ до всіх своїх можливих дій. Щоб додати студента в систему, йому потрібно ввести відповідні дані та відправити на сервер. Якщо дані не проходять валідацію, як приклад, пошта не відповідає своєму формату, ми отримаємо відповідну відповідь від сервера з описом такої помилки:

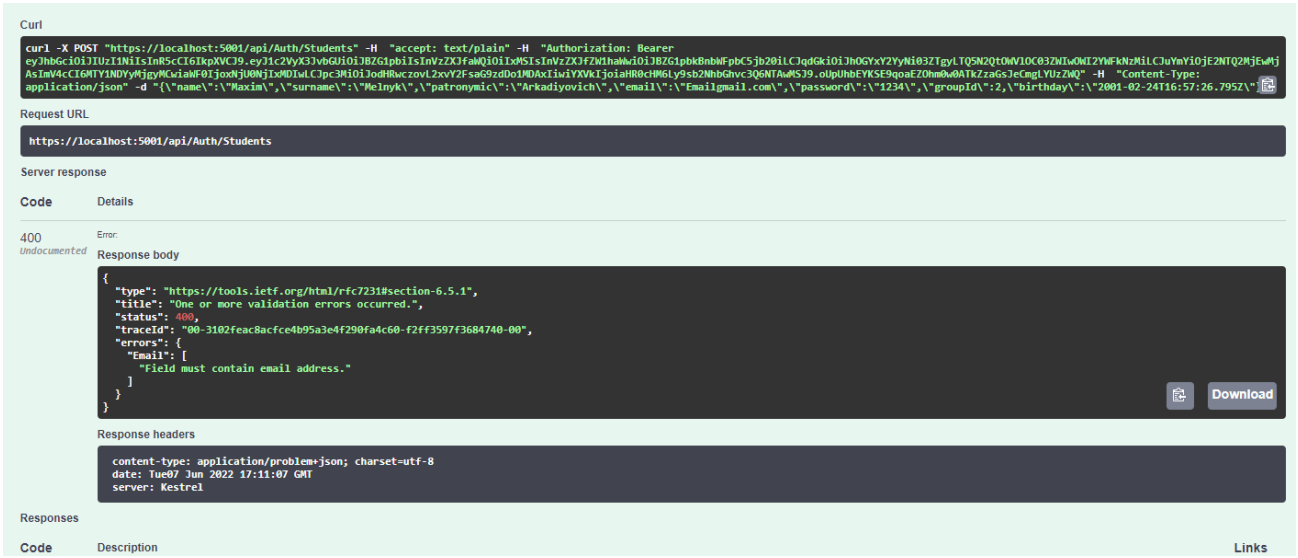


Рис. 40. – Відповідь серверу при спробі додати студента з неправильним форматом пошти

Якщо ж інший користувач, який не являється адміном, спробує виконати таку дію, він теж отримає відповідь від серверу з кодом 403, який означає, що сервер зрозумів запит, проте через внутрішні обмеження не виконає його, в даному випадку через те, що користувач не має таких прав:

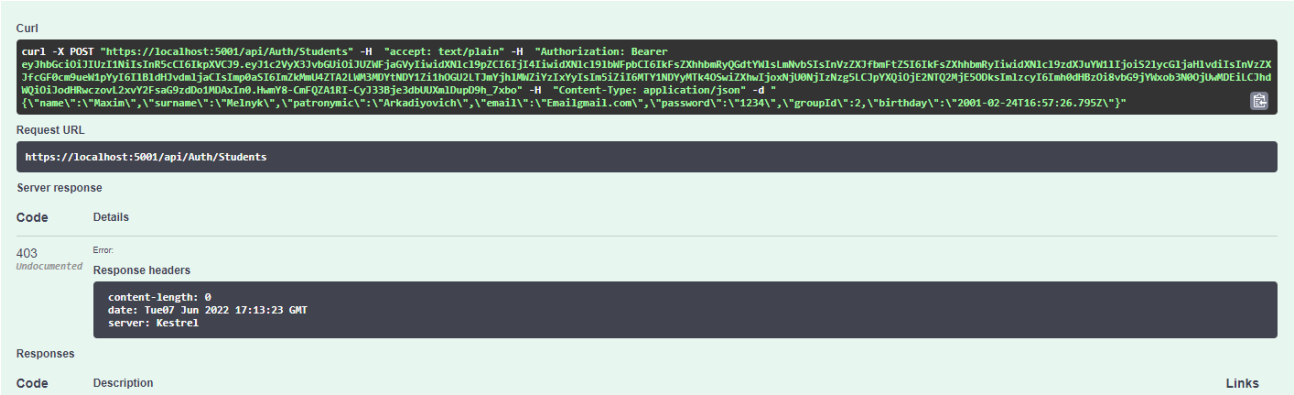


Рис. 41. – Відповідь серверу при спробі додання користувача з невідповідними правами

У разі, коли всі вимоги виконані, користувач являється адміном та ввів коректно дані, сервер повертає наступну відповідь:

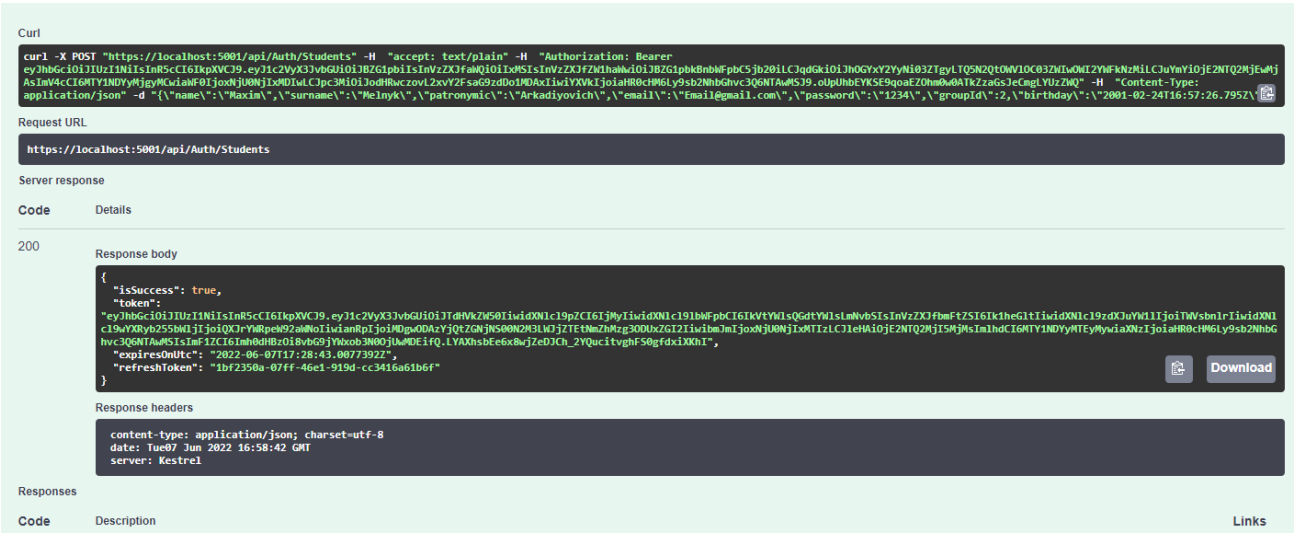


Рис. 42. – Відповідь серверу при успішному додаванні студента

Ми отримуємо результат, що все гаразд, та одразу ж отримуємо токен, який можемо дати студентові. Сам запис тепер знаходиться в БД:

19	29	Student	Student	Student	Student@gmail.com	$2a$11$NWJCsHW9qwagVh/u1EVOjebatJid.Xf6QZhAL23i41gnko6QoNIs(	3	2001-05-23	0	Student
20	30	Student2	Student2	Student2	Student2@gmail.com	$2a$11$Yv4Tk5c/or2ZhIM3yGkZqu1VgfCmoREU10fFPEO.iZlx2IaUHgCXW	3	2001-05-24	0	Student
21	32	Maxim	Melnyk	Arkadiyovich	Email@gmail.com	$2a$11$45nLCvvNcc.FNEy1JcdgL.UOpyOjzlsDt4cef8oMv/5YRfgMc1oLe	2	2001-02-24	0	Student

Рис. 43. – Запис в БД

4.1.3. Створення питання на форумі

Для додавання питання на форумі, користувачу потрібно бути авторизованим як вчитель або студент. Кожен із них має можливість задати питання та дати відповідь. Для цього викликаємо відповідний ендпоінт:

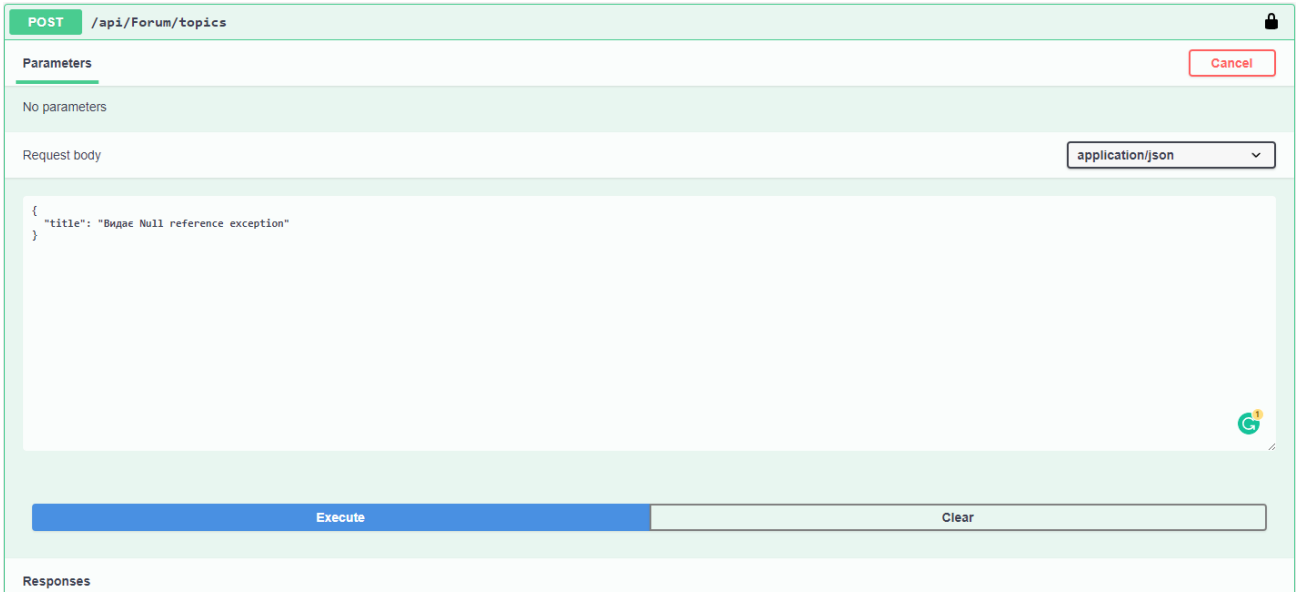


Рис. 44. – Форма відправки створення форуму на сервер

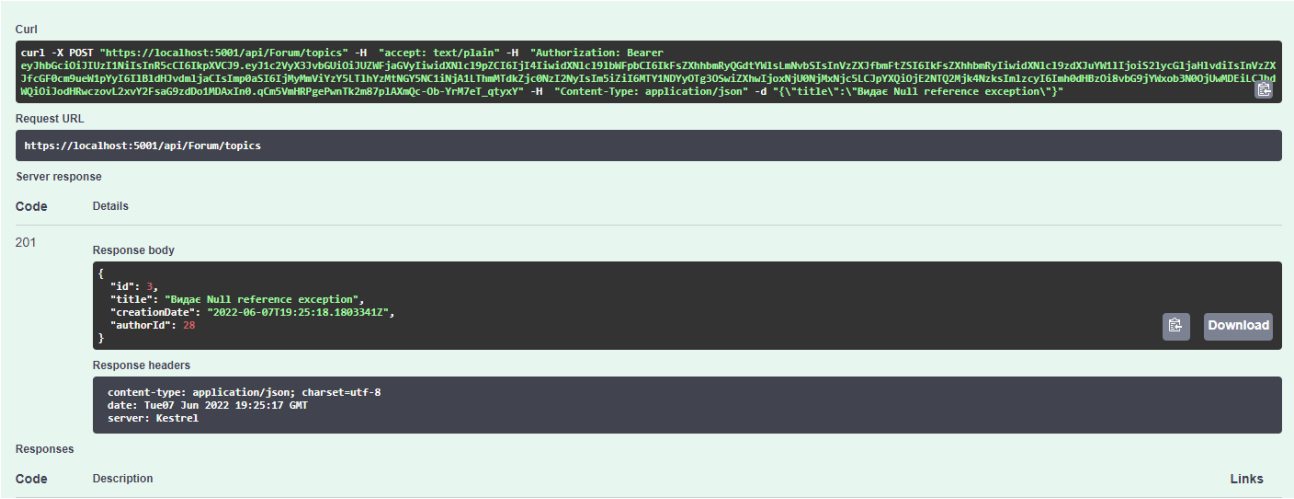


Рис. 45. – Відповідь серверу при успішному створенні питання на форумі

4.1.4. Відповідь на питання на форумі

Для створення відповіді на форумі, потрібно бути авторизованим під вчителя або студента, та обрати сам форум. Для цього викликаємо ендпоінт та подаємо туди Id форуму:

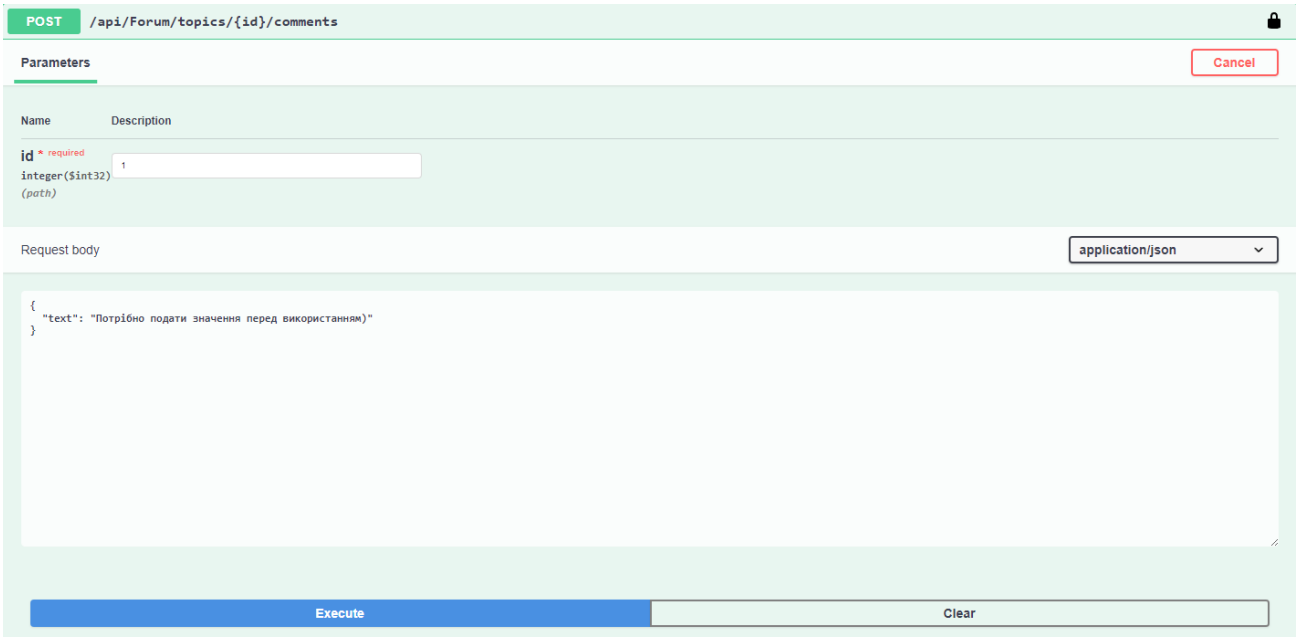


Рис. 46. – Форма відправки для додавання коментаря

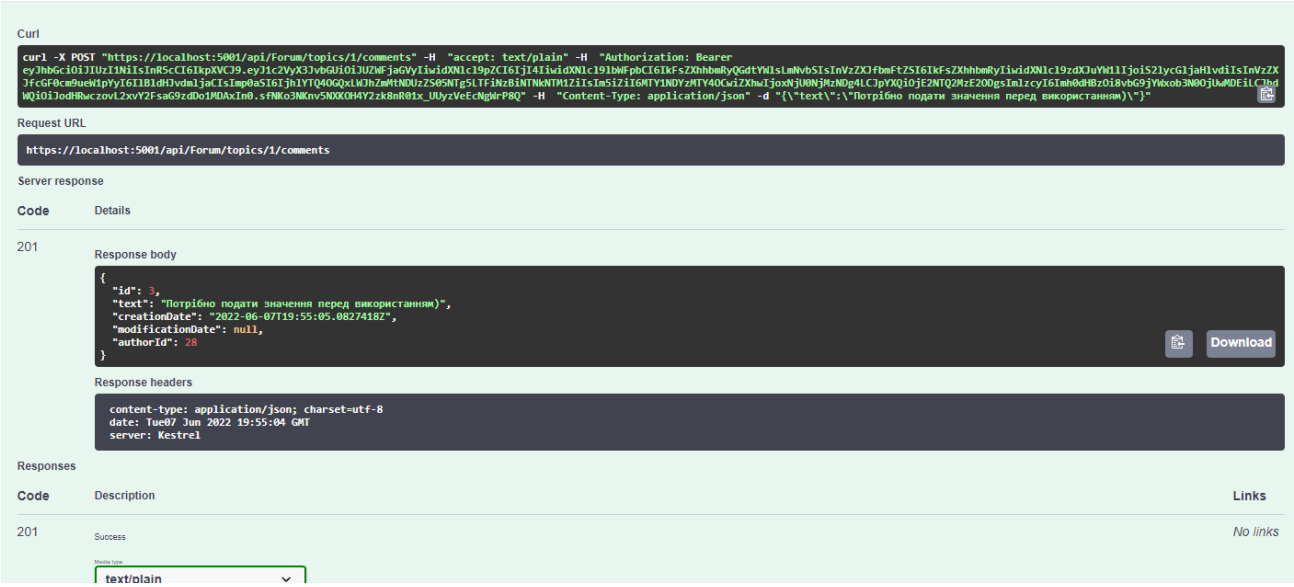


Рис. 47. – Відповідь серверу при успішному додаванню коментаря

4.1.5. Створення та отримання всіх досягнень користувача

Щоб створити досягнення, користувач має бути адміном. Для цього потрібно ввести ім’я, опис досягнення та прикріпити фото досягнення:

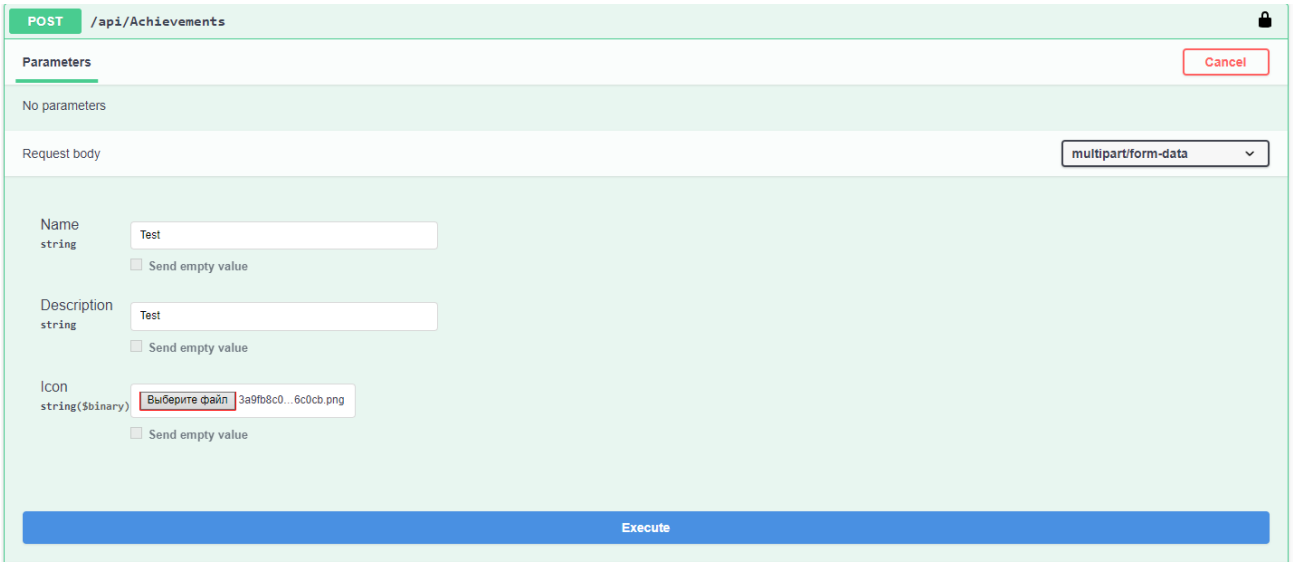
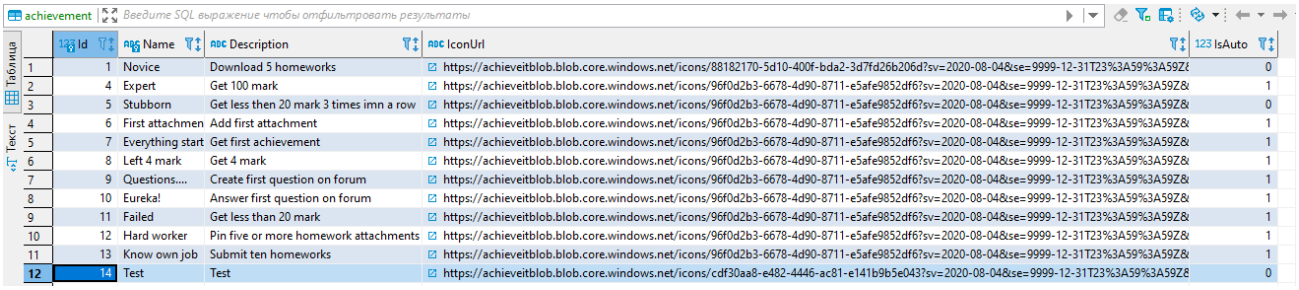


Рис. 48. – Форма відправки для створення досягнення

Рис. 49. – Відповідь серверу після додавання досягнення в базу



	Id	Name	Description	IconUrl	IsAuto
1	1	Novice	Download 5 homeworks	https://achieveitblob.blob.core.windows.net/icons/88182170-5d10-400f-bda2-3d7fd26b206d?sv=2020-08-04&se=9999-12-31T23%3A59%3A59Z&	0
2	4	Expert	Get 100 mark	https://achieveitblob.blob.core.windows.net/icons/96f0d2b3-6678-4d90-8711-e5afe9852df6?sv=2020-08-04&se=9999-12-31T23%3A59%3A59Z&	1
3	5	Stubborn	Get less then 20 mark 3 times imn a row	https://achieveitblob.blob.core.windows.net/icons/96f0d2b3-6678-4d90-8711-e5afe9852df6?sv=2020-08-04&se=9999-12-31T23%3A59%3A59Z&	0
4	6	First attachmen	Add first attachment	https://achieveitblob.blob.core.windows.net/icons/96f0d2b3-6678-4d90-8711-e5afe9852df6?sv=2020-08-04&se=9999-12-31T23%3A59%3A59Z&	1
5	7	Everything start	Get first achievement	https://achieveitblob.blob.core.windows.net/icons/96f0d2b3-6678-4d90-8711-e5afe9852df6?sv=2020-08-04&se=9999-12-31T23%3A59%3A59Z&	1
6	8	Left 4 mark	Get 4 mark	https://achieveitblob.blob.core.windows.net/icons/96f0d2b3-6678-4d90-8711-e5afe9852df6?sv=2020-08-04&se=9999-12-31T23%3A59%3A59Z&	1
7	9	Questions....	Create first question on forum	https://achieveitblob.blob.core.windows.net/icons/96f0d2b3-6678-4d90-8711-e5afe9852df6?sv=2020-08-04&se=9999-12-31T23%3A59%3A59Z&	1
8	10	Eureka!	Answer first question on forum	https://achieveitblob.blob.core.windows.net/icons/96f0d2b3-6678-4d90-8711-e5afe9852df6?sv=2020-08-04&se=9999-12-31T23%3A59%3A59Z&	1
9	11	Failed	Get less than 20 mark	https://achieveitblob.blob.core.windows.net/icons/96f0d2b3-6678-4d90-8711-e5afe9852df6?sv=2020-08-04&se=9999-12-31T23%3A59%3A59Z&	1
10	12	Hard worker	Pin five or more homework attachments	https://achieveitblob.blob.core.windows.net/icons/96f0d2b3-6678-4d90-8711-e5afe9852df6?sv=2020-08-04&se=9999-12-31T23%3A59%3A59Z&	1
11	13	Know own job	Submit ten homeworks	https://achieveitblob.blob.core.windows.net/icons/96f0d2b3-6678-4d90-8711-e5afe9852df6?sv=2020-08-04&se=9999-12-31T23%3A59%3A59Z&	1
12	14	Test	Test	https://achieveitblob.blob.core.windows.net/icons/cdf30aa8-e482-4446-ac81-e141b9b5e043?sv=2020-08-04&se=9999-12-31T23%3A59%3A59Z&	0

Рис. 50. – Додання запису в БД

Для того, щоб отримати всі досягнення користувача, потрібно лише подати його Id. Це може виконати кожен користувач, не залежно від того, яка в нього роль. Це зроблено для того, щоб можливо було переглядати його досягнення в аккаунті.

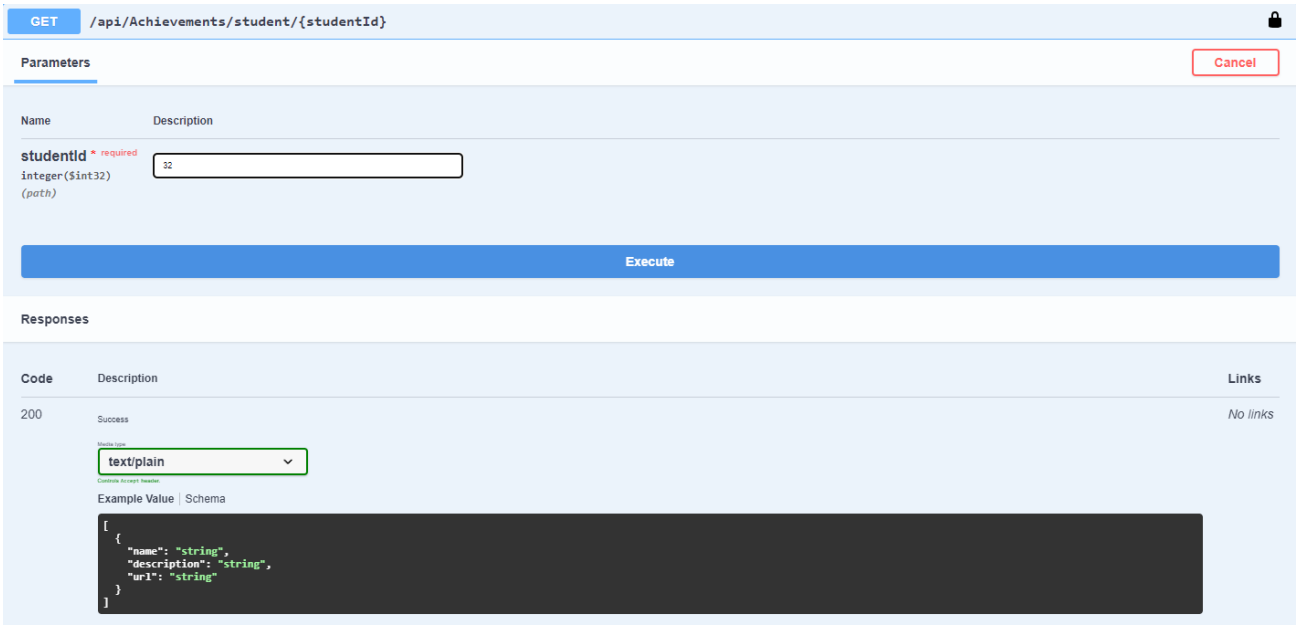


Рис. 51. – Форма відправки для отримання всіх досягнень студента

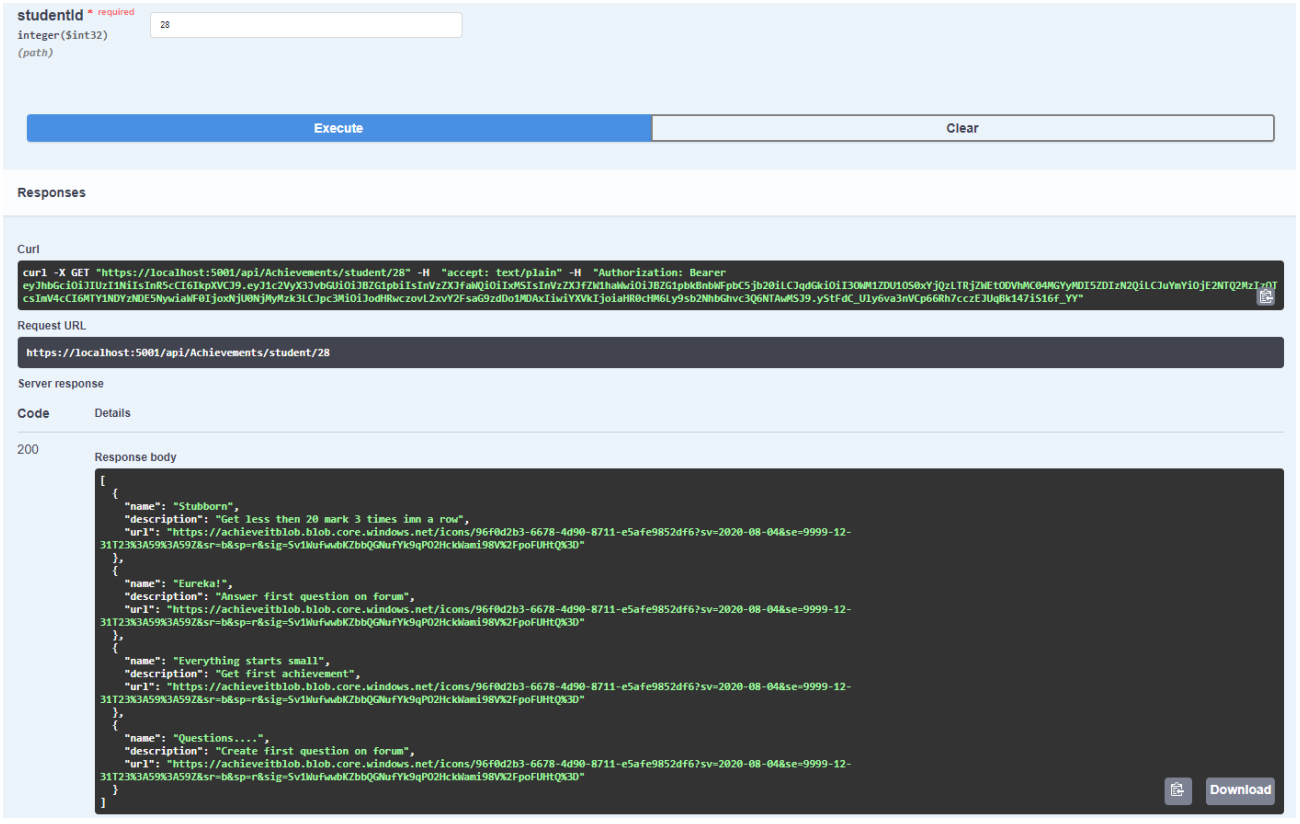


Рис. 52. – Відповідь серверу

4.1.6. Створення групи та додання туди студенів

Можливість редагування груп та додавання туди учасників є лише у адміна. Для цього, потрібно обрати відповідний ендпоінт та заповнити необхіні дані:

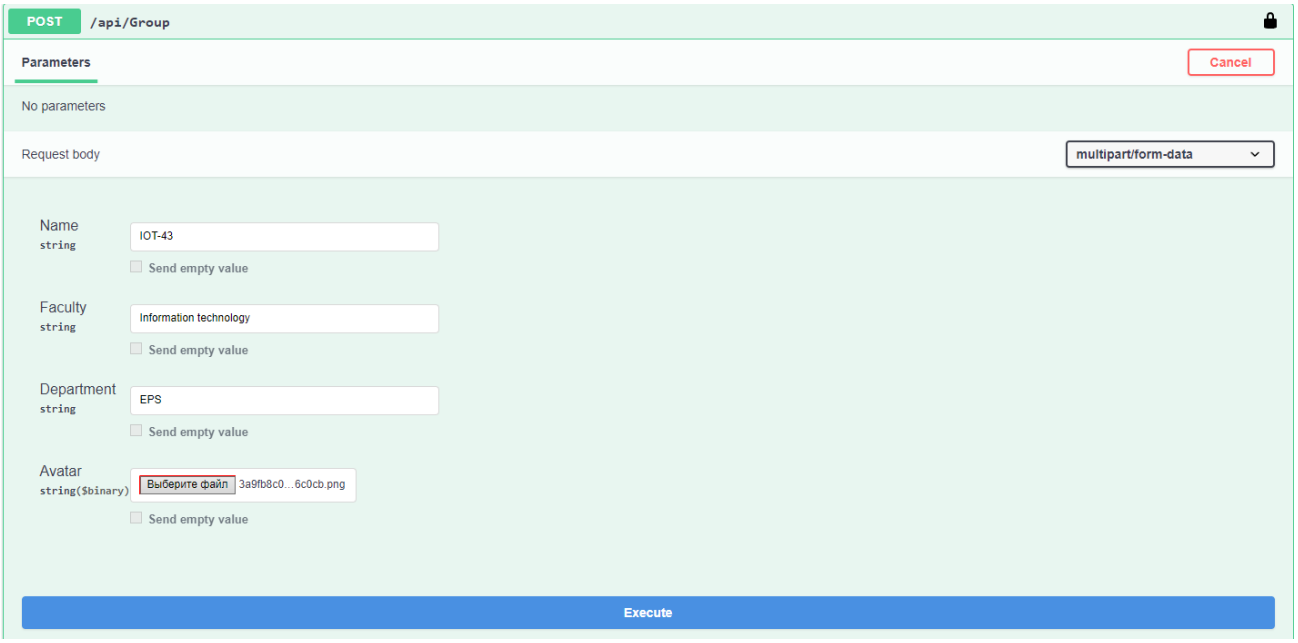


Рис. 53. – Форма відправки для отримання всіх досягнень студента

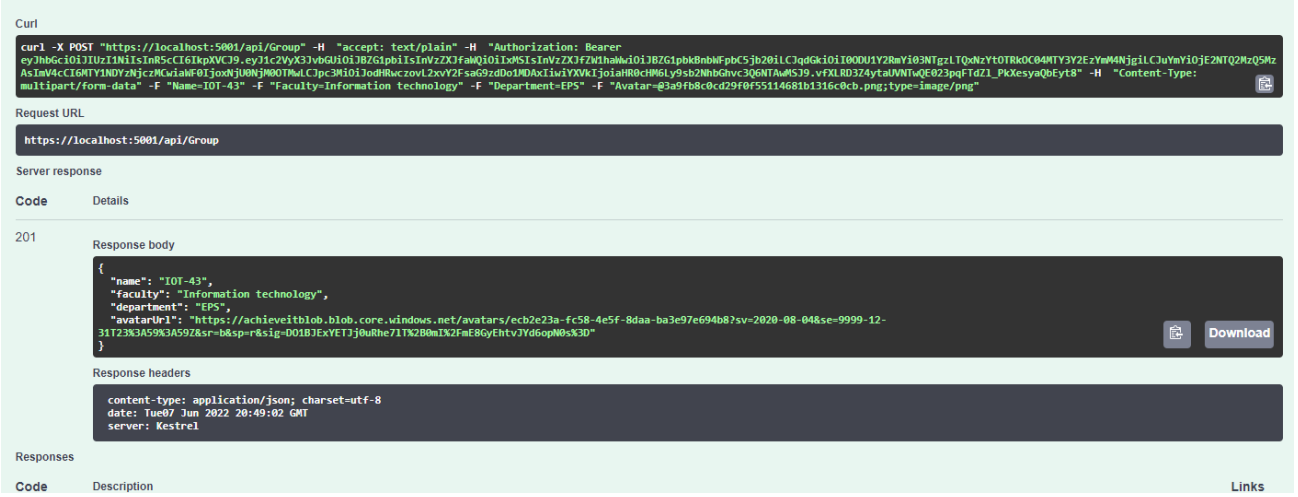
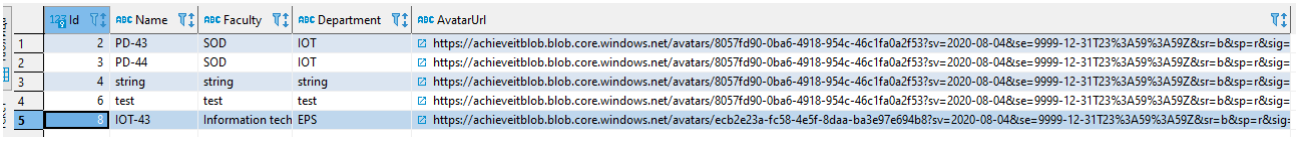


Рис. 54. – Відповідь серверу після відправки запиту на додавання нової групи



	Id	Name	Faculty	Department	AvatarUrl
1	2	PD-43	SOD	IOT	https://achieveitblob.blob.core.windows.net/avatars/8057fd90-0ba6-4918-954c-46c1fa0a2f53?sv=2020-08-04&se=9999-12-31T23%3A59%3A59Z&sr=b&sp=r&sig=
2	3	PD-44	SOD	IOT	https://achieveitblob.blob.core.windows.net/avatars/8057fd90-0ba6-4918-954c-46c1fa0a2f53?sv=2020-08-04&se=9999-12-31T23%3A59%3A59Z&sr=b&sp=r&sig=
3	4	string	string	string	https://achieveitblob.blob.core.windows.net/avatars/8057fd90-0ba6-4918-954c-46c1fa0a2f53?sv=2020-08-04&se=9999-12-31T23%3A59%3A59Z&sr=b&sp=r&sig=
4	6	test	test	test	https://achieveitblob.blob.core.windows.net/avatars/8057fd90-0ba6-4918-954c-46c1fa0a2f53?sv=2020-08-04&se=9999-12-31T23%3A59%3A59Z&sr=b&sp=r&sig=
5	7	IOT-43	Information tech	EPS	https://achieveitblob.blob.core.windows.net/avatars/ecb2e23a-fc58-4e5f-8daa-ba3e97e694b8?sv=2020-08-04&se=9999-12-31T23%3A59%3A59Z&sr=b&sp=r&sig=

Рис. 55. – Додання запису в БД

Для того, щоб додати студента до групи, нам потрібно знати його пошту та Id групи:

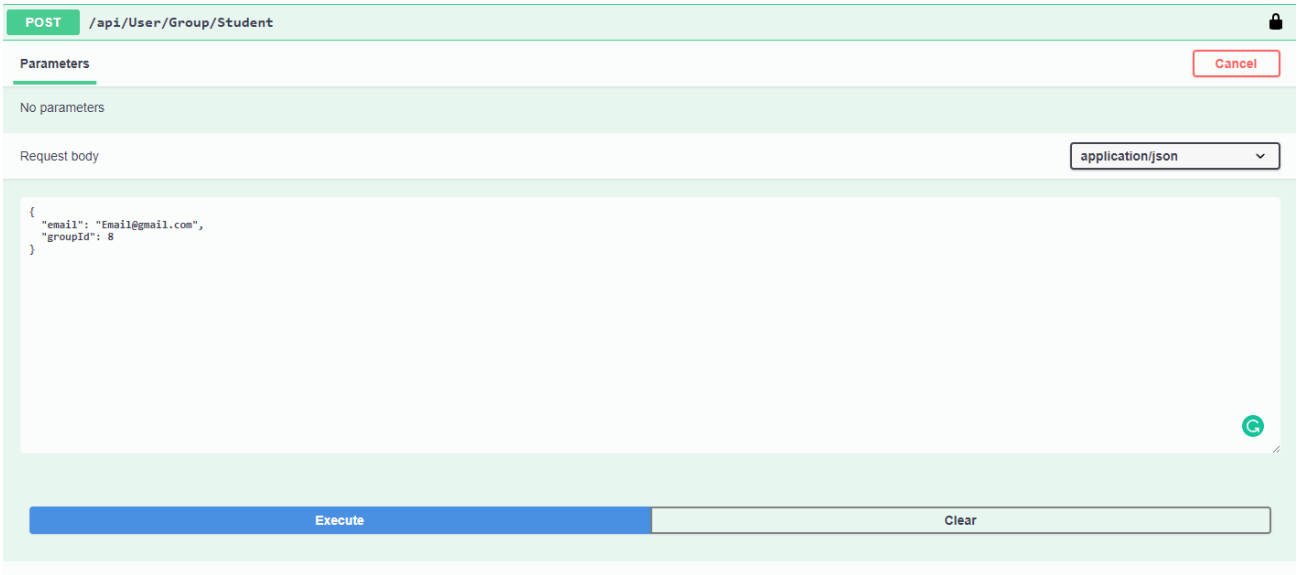
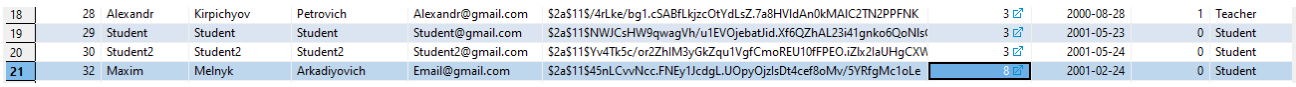


Рис. 56. – Запис студента до групи



18	28	Alexandr	Kirpichyov	Petrovich	Alexandr@gmail.com	$2a$11$/4rLke/bg1.cSABfLkjzcOtYdLsZ.7a8HVldAn0kMAIC2TN2PPFNK	3	2000-08-28	1	Teacher
19	29	Student	Student	Student	Student@gmail.com	$2a$11$NWJCsHW9qwagVh/u1EVOjebatJid.Xf6QZhAL23i41gnko6QoNIs(	3	2001-05-23	0	Student
20	30	Student2	Student2	Student2	Student2@gmail.com	$2a$11$Yv4Tk5c/or2ZhlM3yGkZqu1VgfCmoREU10fFPEO.iZIx2IaUHgCXV	3	2001-05-24	0	Student
21	32	Maxim	Melnyk	Arkadiyovich	Email@gmail.com	$2a$11$45nLCvvNcc.FNEy1JcdgL.UOpyOjzlsDt4cef8oMv/5YRfgMc1oLe	3	2001-02-24	0	Student

Рис. 57. – Додання запису в Б

ВИСНОВКИ

Дана робота була спрямована на підвищення мотивації студентів при процесі дистанційного навчання завдяки впровадженню елементів гейміфікації.

1. Було обґрунтовано актуальність розробленої системи та її наукову новизну шляхом аналізу схожих за функціоналом продуктів. Було досліджено типові проблеми, з якими зустрічаються студенти під час дистанційного навчання, проблеми існуючих додатків та їх переваги. Було сформовано та описано технічні завдання для системи.

2. Було спроектовано архітектуру системи та розроблено веб-додаток з використанням фреймворку ASP.NET, бази даних MySQL та сховище файлів Azure blob storage, системи контролю версій Git, середовищем розробки JetBrains Rider та наведені головні переваги цих інструментів.

3. Було описано архітектуру, ключові особливості та алгоритми роботи системи у вигляді схем та діаграм.

4. Було досліджено особливості завантаження, збереження, обробки та отримання доступу до файлів при розробці веб-додатку за допомогою технології Azure blob storage.

5. В ході розробки системи було використано шаблони проектування та сучасні принципи ООП для створення гнучкого та швидкого програмного забезпечення із слабкою зв’язністю компонентів.

6. Було досліджено різні методи автентифікації користувачів та застосовано Bearer токен.

7. Було досліджено особливості роботи баз даних MySQL та Azure blob storage. В якості збереження основних даних було обрано MySQL, для зберігання файлів Azure blob storage.

Результати дослідження бакалаврської роботи апробовані на всеукраїнських науково-технічних конференціях: “Застосування програмного забезпечення в інфокомунікаційних технологіях” та “IV Міжнародна студентська конференція Наука сьогодення: від досліджень до стратегічних рішень”.

ПЕРЕЛІК ПОСИЛАНЬ

1. About Moodle – MoodleDocs - [Електронний ресурс] – Режим доступу: https://docs.moodle.org/400/en/About_Moodle
2. Introduction to Azure Storage - Cloud storage on Azure | Microsoft Docs - [Електронний ресурс] – Режим доступу: https://docs.microsoft.com/en-us/azure/storage/common/storage-introduction
3. NZ Інструкція для учнів і батьків - [Електронний ресурс] – Режим доступу: https://nz.ua/upload/NZ%20Інструкція%20для%20учнів%20і%20батьків.pdf
4. 1.3. Основні переваги субд MySql - [Електронний ресурс] – Режим доступу: https://studfile.net/preview/5607354/page:3/
5. JetBrains: Essential tools for software developers and teams - [Електронний ресурс] – Режим доступу: https://www.jetbrains.com
6. DBeaver Community | Free Universal Database Tool - [Електронний ресурс] – Режим доступу: https://dbeaver.io
7. Postman API Platform | Sign Up for Free - [Електронний ресурс] – Режим доступу: https://www.postman.com
8. Git - [Електронний ресурс] – Режим доступу: https://git-scm.com
9. Застосування UML (частина 3). Діаграма класів - Class Diagram :: Державний університет телекомунікацій - [Електронний ресурс] – Режим доступу: https://dut.edu.ua/ua/news-1-626-8002-zastosuvannya-uml-chastina-3-diagrama-klasiv----class-diagram_kafedra-kompyuternih-nauk-ta-informaciynih-tehnologiy
10. MySQL - [Електронний ресурс] – Режим доступу: https://www.mysql.com
11. Horsdal C. Microservices in .NET Core / Christian Horsdal., 2017. – 55-248 с.

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Розробка web-додатку « Веб-порталу Університету » мовою C#

Виконав студент 4 курсу
Групи ПД-43
Мельник Максим Аркадійович
Керівник роботи
к.т.н., доц. кафедри ІПЗ Поперешняк Світлана Володимирівна

Київ – 2022

АНАЛОГИ

АНАЛІЗ АНАЛОГІВ

Особливості	“Achievelt”	Moodle	Google classroom	Нові знання
Відкрите API	+	-	-	-
Досягнення	+	-	-	-
Форум для обговорення	+	+	-	-
Журнал оцінок	+	+	-	+
Перегляд аккаунту	+	+	-	+
Швидкість розгортання	+	-	+	-
Сортування завдань	+	+	-	-

МЕТА, ОБ’ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

Мета роботи – підвищення мотивації студентів в процесі дистанційного навчання завдяки впровадженню елементів гейміфікації.

Об’єкт дослідження – процес дистанційного навчання.

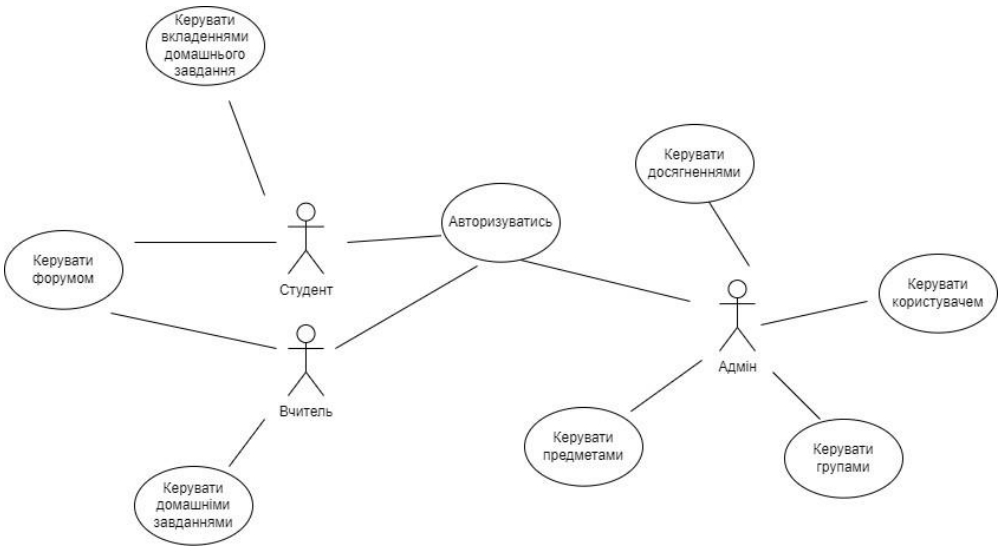
Предмет дослідження - web-додаток веб-порталу Університету для підтримки процесу дистанційного навчання.

ТЕХНІЧНІ ЗАВДАННЯ

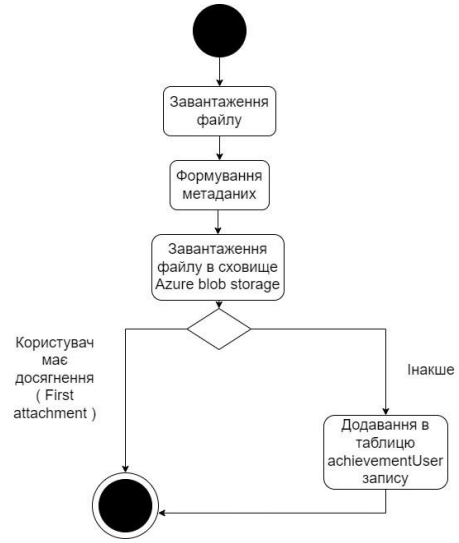
1. Розділити розміщення файлів і даних користувачів по окремим базам даних
2. Реалізувати автентифікацію через Bearer-токен
3. Спроектувати взаємозв’язок ролей в системі
4. Реалізувати взаємодію з базами через шаблон Unit of Work
5. Імпортувати та налаштувати систему логування
6. Інтегрувати гейміфікацію за допомогою реалізації алгоритму отримання досягнень.

ПРОГРАМНІ ЗАСОБИ РЕАЛІЗАЦІЇ

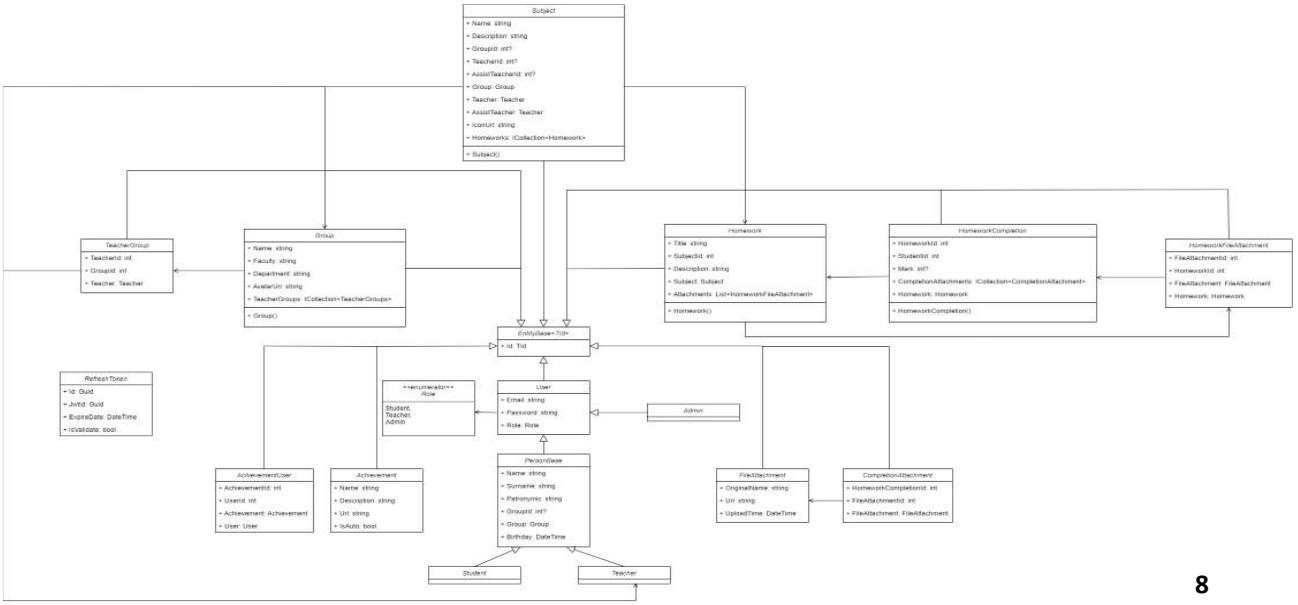
ДІАГРАМА ПРЕЦЕДЕНТІВ



ДІАГРАМА ДІЯЛЬНОСТІ АЛГОРИТМУ ДОДАВАННЯ ДОСЯГНЕННЯ

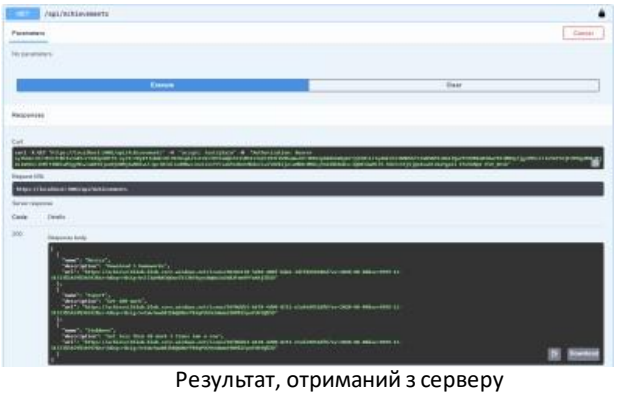


ДІАГРАМА КЛАСІВ



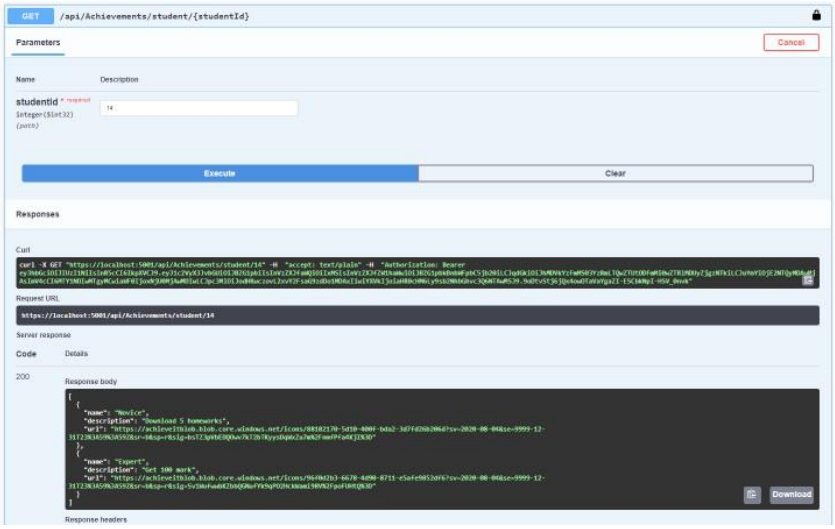
ЕКРАННІ ФОРМИ

Endpoint для отримання всіх досягнень в базі даних



Результат, отриманий з серверу

ЕКРАННІ ФОРМИ



Приклад відповіді, отриманої від серверу за запитом виведення всіх досягнень користувача

АПРОБАЦІЯ РЕЗУЛЬТАТІВ ДОСЛІДЖЕННЯ

1. Мельник М.А. Розробка веб-додатка як системи управління навчанням / С.В. Поперешняк, М.А. Мельник // Науково-технічна конференція «Застосування програмного забезпечення в інфокомунікаційних технологіях». Збірник тез. – К.: ДУТ, 2022. – С. 43

2. Мельник М.А. Використання веб-токенів / С.В. Поперешняк, М.А. Мельник // IV Міжнародна студентська конференція Наука сьогодення: від досліджень до стратегічних рішень м. Івано-Франківськ, Україна. 13.06.2022. Прийнято до друку.

ВИСНОВКИ

1. Проаналізовано сучасні аналоги та було виявлено основні недоліки, що дало змогу створити проєкт з їх урахуванням
2. Досліджено предметну область та реалізовано систему досягнень, щоб збільшити мотивацію студентів.
3. Досліджено особливості завантаження, збереження, обробки та отримання доступу до файлів при розробці веб-додатку за допомогою технології Azure blob storage.
4. Спроектовано архітектуру системи та розроблено веб-додаток з використанням фреймворку ASP.NET, бази даних MySQL та сховище файлів Azure blob storage, системи контролю версій Git, середовищем розробки JetBrains Rider та наведені головні переваги цих інструментів.
5. Розроблено алгоритм отримання досягнень, який в подальшому сприяє процесу гейміфікації дистанційного навчання.
6. Розроблено програмний інтерфейс веб-додатку веб-порталу Університету для підтримки процесу дистанційного навчання.

ДЯКУЮ ЗА УВАГУ!