МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»

Кафедра інженерії програмного забезпечення

**КУРСОВИЙ РОБОТА**

(ПОЯСНЮВАЛЬНА ЗАПИСКА)

з дисципліни: «Бази даних»

на тему:

**«БАЗА ДАНИХ ПЛАТФОРМИ ДЛЯ ПРОВЕДЕННЯ КУРСІВ»**

студента II курсу групи ІПЗ-20-4

спеціальності 121 «Інженерія програмного забезпечення»

Кормиша Романа Івановича

(прізвище, ім’я та по-батькові)

Керівник: к.т.н., доц. Сугоняк.І.І.

Дата захисту: " \_\_\_ " \_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ р.

Національна шкала \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кількість балів: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Оцінка: ECTS \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Члени комісії \_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_ О.В. Коротун .

(підпис) (прізвище та ініціали) \_\_\_С.М. Кравченко\_ .

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ О.В. Чижмотря .

(підпис) (прізвище та ініціали)

Житомир – 2022

ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»

Факультет інформаційно-комп’ютерних технологій

Кафедра інженерії програмного забезпечення

Освітній рівень: бакалавр

Спеціальність 121 «Інженерія програмного забезпечення»

«ЗАТВЕРДЖУЮ»

В.о.зав. кафедри ІПЗ

А.В.Морозов

“\_\_\_\_” 2022 р.

ЗАВДАННЯ

НА КУРСОВУ РОБОТУ СТУДЕНТУ

Кормишу Роману Івановичу

1. Тема роботи: Розробка бази даних платформи для проведення курсів

керівник роботи: к.т.н., доц. Сугоняк І.І. .

1. Строк подання студентом: “ ” 2022р.
2. Вихідні дані до роботи: Розробити базу даних для платформи для проведення курсів.
3. Зміст розрахунково-пояснювальної записки(перелік питань. Які підлягають розробці)

* Аналіз інформаційних потоків та особливостей предметної області дослідження
* Проектування бази даних за напрямком курсової роботи
* Реалізація підсистеми обробки даних за напрямком курсової роботи
* Адміністрування баз даних

1. Перелік графічного матеріалу(з точним зазначенням обов’язкових креслень)
2. Презентація до КР: .
3. Посилання на репозиторій: <https://github.com/Raimhal/ISkills.git>
4. Консультанти розділів проекту (роботи)

|  |  |  |  |
| --- | --- | --- | --- |
| Розділ | Прізвище, ініціали та посади консультанта | Підпис, дата | |
| Завдання видав | Завдання прийняв |
| 1,2,3 | Сугоняк І.І., к.т.н., доц. |  |  |
|  |  |  |  |

1. Дата видачі завдання “ 10 ” лютого 2022 р.

**КАЛЕНДАРНИЙ ПЛАН**

|  |  |  |  |
| --- | --- | --- | --- |
| № з/п | Назва етапів курсової роботи | Термін виконання етапів роботи | Примітки |
| 1 | Постановка задачі | 10.02.22 | Виконано |
| 2 | Пошук, огляд та аналіз аналогічних розробок | 18.02.22 | Виконано |
| 3 | Формулювання технічного завдання | 19.02.22 | Виконано |
| 4 | Опрацювання літературних джерел | 22.02.22 | Виконано |
| 5 | Проектування структури | 23.02.22 | Виконано |
| 6 | Написання програмного коду | 24.02.22 –  16.06.22 | Виконано |
| 7 | Відлагодження | 16.06.22 –  8.07.22 | Виконано |
| 8 | Написання пояснювальної записки | 10.07.22 –  15.07.22 | Виконано |
| 9 | Захист |  |  |

**Студент***\_\_\_\_\_\_\_\_\_\_\_\_\_\_* Кормиш Р.І

(підпис) (прізвище та ініціали)

**Керівник проекту** \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Сугоняк І.І.

(підпис) (прізвище та ініціали)

**РЕФЕРАТ**

Завданням на курсовой проект (роботу) було розробити базу даних до платформи проведення курсів.

Пояснювальна записка до курсового проекту (роботи) на тему «Розробка бази даних платформи для проведення курсів» складається з вступу, чотирьох розділів, висновків, списку використаної літератури та додатку.

Текстова частина викладена на 69 сторінках друкованого тексту. Пояснювальна записка має 10 сторінку додатків. Список використаних джерел містить 10 найменувань і займає 1 сторінку. В роботі наведено 39 рисунків. Загальний обсяг роботи – 79 сторінок.

Ключові слова: POSTGRESQL, REACT, REDUX, ASP.NET, CLOUDINARY, КЛІЄНТ, АДМІН, АВТОРИЗАЦІЯ, КУРСИ, ІНТЕРНЕТ, РОЛІ, ДАНІ, МОДЕЛЬ ДАНИХ, БАЗА ДАНИХ, КЛЮЧОВЕ ПОЛЕ, ЗАПИТ, AUTH.

**Зміст**

[**ВСТУП** 7](#_Toc92540836)

[РОЗДІЛ 1 АНАЛІЗ ПРОБЛЕМАТИКИ, МЕТОДІВ ТА ЗАСОБІВ ВИРІШЕННЯ ЗАДАЧІ 8](#_Toc92540837)

[**1.1** **Аналіз задачі, засобів та методів її вирішення** 8](#_Toc92540838)

[**1.2** **Аналіз існуючого програмного забезпечення за тематикою курсової роботи.** 8](#_Toc92540839)

[**Висновки до першого розділу :** 14](#_Toc92540840)

[РОЗДІЛ 2 ПРОЕКТУВАННЯ ТА РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 16](#_Toc92540841)

[**2.1 Проектування загального алгоритму роботи веб-додатку** 16](#_Toc92540842)

[Загальна схема роботи програми : 17](#_Toc92540843)

[**2.2 Розробка функціональних алгоритмів роботи програми** 18](#_Toc92540844)

[**2.3 Розробка програмного забезпечення** 19](#_Toc92540845)

[РОЗДІЛ 3 ОПИС РОБОТИ З ПРОГРАМНИМ ДОДАТКОМ ТА ЙОГО ТЕСТУВАННЯ 24](#_Toc92540846)

[**3.1 Опис роботи з програмним додатком (Опис інтерфейсу)** 24](#_Toc92540847)

[**3.2** **Тестування роботи програмного забезпечення** 30](#_Toc92540848)

[**Висновки до 3 розділу :** 31](#_Toc92540849)

[**ВИСНОВКИ** 32](#_Toc92540850)

[**СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ** 33](#_Toc92540851)

[**ДОДАТКИ** 34](#_Toc92540852)

**Перелік умовних скорочень**

Рис – рисунок

БД – база даних

КР – курсова робота

CRUD – create, read, update, delete

ІС – інформаційна система

ТЗ – технічне завдання

СУБД – система управління базою даних

**ВСТУП**

В даній курсовій роботі буде наведено процес побудови клієнтського додатку та розробка бази для платформи для проведення курсів.

**Актуальність теми.** Використання баз даних є однією з характерних рис більшості сучасних інформаційних систем. По своїй суті бази даних є тим, навколо чого і будується інформаційна система будь-якого підприємства. Тому теорії створення та практиці використання баз даних приділяється достатня увага протягом періоду функціонування ІС. Тривалий час основним типом були реляційні бази даних, які на сьогодні вже вважаються класичними. Однак класичність не означає легко та швидко. Хоча на практиці і використовується чимало різноманітних баз даних, але для більшості з них існує велика кількість шаблонних фрагментів коду що, робить створення додатку нудним і призводить до однакових помилок при розобці. Тому для спрощення процесу розробки додатку потрібно знаходити щось зручніше, легше та сучасніше. Цьому завданню і було приділену головну увагу при розробці.

**Метою роботи** є дослідження особливостей проектування та реалізації бази днаих за визначеною темою курсової роботи.

**Завданням на курсову роботу є :**

* аналіз теоретичних засад проектування та реалізації систем на основі баз даних;
* визначення інформаційних потреб предметної області дослідження;
* аналіз напрямку ризиків інформаційних потоків та їх структури;
* проектування бази даних за визначеною предметною областю;
* розробка математичної та алгоритмічної моделі функціонування системи на основі БД;
* реалізація БД та інтерфейсних засобів інформаційної системи.

**Об'єктом дослідження** є методи та засоби проектування баз даних за визначеними предметними областями.

**Предметом дослідження** можливості застосування концепції БД та СУБД для забезпечення інформаційних потреб предметної області.

**РОЗДІЛ 1** **ТЕОРЕТИЧНИЙ АНАЛІЗ ІНФОРМАЦІЙНИХ ПОТОКІВ ТА ОСОБЛИВОСТЕЙ ПРЕДМЕТНОЇ ОБЛАСТІ ДОСЛІДЖЕННЯ**

**1.1 Аналіз інформаційних потреб та визначення предметної області дослідження**

В даному курсовому проекті за предметну область взято базу даних, яка буде зберігати усю необхідну інформацію про платформу курсів. Ця платформа надає можливість отримувати знання з різних спеціальностей за допомогою мережі Internet, тому деякі дання потрібно зберігати саме в мережі. Щоб користувачі мали змогу одержати для себе більш повну інформацію про курс та про платформу, необхідно створити базу даних, яка зберігатиме усі необхідні дані. А для амінів, потрібно зробити можливість контролю даних та отримання статистичної інформації.

Для коректного виконання запитів на отримання даних, дані повинні вноситись та змінюватись у БД. За це будуть відповідати деякий перелік операцій:

* Реєстрація користувача. Додаток повинен розрізняти користувачів, щоб давати йому доступ до певних даних та функцій, тому потрібно, створити таблицю, з користувачами. Для заповнення цієї таблиці потрібне створення нового користувача, за це і відповідатиме дана операція.
* Вхід користувача. Логічний опис використано у попередньому пункті (функція, що дозволить розрізняти користувачів).
* Реєстрація курсу. Для того, щоб система розуміла, що в ній є потрібно це щось, передавати їй і зберігати про нього інформацію. Таким чином, дана функція буде отримувати новий курс та зберігати його в БД.
* Редагування курсу. Може трапитись ситуація, що потрібно буде змінити якусь інформацію про курс, наприклад при некоректному заповненні даних, тому потрібно реалізувати дану функцію.
* Створення категорій та тем курсів, що дасть змого лекше орієнтуватися в тематиці курсів та надасть можливість зручного пошуку за інтересами.
* Зміна теми курсу, що забезпечить розшинення оновлення даних про курс, яке було наведено вище.
* Створення роздів курсу. Курси мають мати деякі ключові розділи, що користувач міг легко відсліджувати сівй прогрес.
* Редагування розділів та додання відео записів, з можливістю подальшого перегляду та редагування. Ця функція надасть можливість змінювати інформації про певний розділ та додавати відео-матеріал курсу, що поліпсить спийняття інформації.
* Створення коментаріїв. Ця опція необхідна, щоб користувач зміг поділитися важеннями про курс та вплинути як на рейтинг самого курсу, та і на рейтинг його автора.
* Купівля курсів. Щоб зробити можливим отринання коштів за певну інформацю та реалівувати платні курси, адже ми всі знаємо, що найцінніша інформація є платною, дана функція необхідна.
* Перегляд курсів. Дана функція потрібна для того, щоб користувач розумів, які курси наявні для вивчення.
* Перегляд профілю потрібен, щоб користувач міг переглянути або відредактувати дані, які про нього будуть бачитми інші користувачі.
* Пошук та відбір курсів по мультикритеріями (ціна, дата створення, рейтинг і т.д). Завдання даної платформи дозволяти користувачам онлайн отримувати знання по певним спеціальностям. Для цього користувачу портібно дати можливість шукати та аналізувати курси, які він хочу придбати або отримати. Для цього потрібний одночасний пошук по темі курсу та по параметру його назви. Також даний функціонал розширюється на сторінці користувача, де він може відслідковувати саме власностворені курси, та ті які він придбав.
* Статистика. Дуже важливим є аналізування інформації про прибуток за певний проміжок часу, саме цим і буде займатись дана функція, про кількість куплених курсів та про рейтинги авторів курсів в вспіввідношенні до їх кількості власностворених курсів. Також дана функція буде гарно візуалізувати дані, для зручного та швидкого аналізу.
* Створення бекапів. Адміністратору варто мати можливість зробити резервну копію бази щоб зберігти дані та потім мати змогу відновити її у випадку взлому, або пошкодження.
* Відновлення бази. Дана функція буде очищати таблиці поточної БД, та заповнювати їх даними з обраного бекапу.

**1.2 Архітектура та засоби реалізації бази даних**

Аналіз та вибір СУБД проведемо з урахуванням того, що число клієнтських місць становить від 30 до 500, а доступ до даних має бути максимально ефективним. Обчислювальна техніка працюватиме під керівництвом ОС Windows і Linux. В таблицях наведено порівняльні характеристики СУБД.

Таблиця 1.

|  |  |  |  |
| --- | --- | --- | --- |
| Характеристика | Microsoft SQL Server | MongoDB | Postgresql |
| Робота під керівництвом ОС | Задовільно | Добре | Добре |
| Складність | Відмінно | Добре | Відмінно |
| Доступ до даних | Відмінно | Відмінно | Відмінно |
| Використання у  великих проектах | Відмінно | Добре | Відмінно |
| Схема даних | Добре | Відмінно | Добре |
| Підтримувані мови  програмування | Відмінно | Відмінно | Відмінно |
| Підключення до  Web | Відмінно | Відмінно | Відмінно |
| Графічні інструменти | Відмінно | Відмінно | Відмінно |
| Тригери | Відмінно | Добре | Добре |
| Паралелізм | Відмінно | Відмінно | Відмінно |
| Одночасний  доступ декількох  користувачів | Добре |  | Відмінно |
| Обробка даних мультимедіа | Слабо | Добре | Відмінно |
| Пошук за регулярними виразами | Добре | Відмінно | Відмінно |
| Підтримка об’єктно-орієнтованої  парадигми | Відмінно | Відмінно | Відмінно |
| Засоби аналізу | Відмінно | Відмінно | Відмінно |

Пояснення таблиці:

Крім того, що Microsoft SQL Server та Postgresql зберігають дані реляційно, а MongoDb використовує сховище документів, дані субд мають інші різні характеристики, більше того, ці характеристики можна оцінити, що і зображено у таблиці номер 1.

Серверними операційними системами для всіх СУБД є Windows та Linux, також Postgresql та MongoDB маю web варіант. Для MongoDb це Atlas, а для Postgresql – Postgresql Web.

Cкладність. Під час аналізування, вивчення та розробки за допомогою СУБД MongoDB виникає труднощі, у той момент, як при використанні SQL Server більшість проблем виникає лише при навчанні та розуміння логіки мови, що дає даній СУБД оцінку відмінно в даному критерії. Якщо звернути увагу на простоту до Postgresql, то її аналізувати тяжко. Але якщо порывнбвати синтаксис, то вона є простішою і інтуїтивно зрозумілішою ніж MongoDB чи MS SQL.

# У всіх СУБД користувачі отримують доступ лише туди, куди йому його надають, що забезпечує безпеку даних.

Хоча і MongoDB це нереляційна база даних, але вона дозволяє отривувати згруповані дані, за допомогою аналогу методу join, так як і MS SQL чи Postgresql, що є великим плюсом для великих проектів, де іноді, потрібно швидко отримати дані відразу з кількох десятків, а то, і сотень таблиць.

Але, для невеликих проектів, потрібно виконати багато роботи та шаблонних налаштувань, щоб коректно працювала база даних, розроблена на SQL Server, також потрібно зробити багато налаштувань безпеки, щоб не виникло різних казусів з витіком даних, у той момент, як Postgresql зробить ці налаштування сам, і не пропустить зловмисника так просто та підключення до БД здійснюється досить просто, порівнянно з іншими.

Схема даних(Схема баз даних — це структура системи баз даних описана мовою, яка підтримується системою керування баз даних і відноситься до організації даних для створення плану побудови бази даних з розподілом на таблиці.). Підтримка фіксованої схеми та розгортання без схеми з можливістю взаємодії між ними реалізована лише не підтримується ні в одній з БД, у той час як в SQL Server, Postgresql та MongoDb потрібно розгортати лише з описаною схемою даних.

По критерію підтримуваних мов програмування не можна виділити когось конкретного, тому що всі представлені СУБД є сучасними і підтримують доволі велику кількість мов програмування, така як: C#, Python, JavaScript, C, Java.

Кожна з перечислених СУБД має додаток, або сайт за допомогою якого можна керувати базою та налаштовувати її(тобто є графічні інтерфейси), кожен з даних інтерфейсів є зручним та простим у використанні(однак іноді тяжко розібратись де знайти певний функціонал та як його заповнити, без допомоги інтернету чи інших джерел інформації).

Звернемо увагу на тригери(події, що дозволяють перехолювати дії при створенні, видаленні чи оновленні даних). У Postgresql дана функціональність виконується та підключається автоматично, відбувається відслідковування змін у БД, та при коректному підлюченні до проекту, зміни буде відразу передаватись та аналізуватись. Тригери також реалізовані у SQL SERVER, даний функціонал призначений для тих самих дій, що і в Postgresql, однак він може виконуватись не лише після певних змін у бд, а й замість них, що є дуже зручним, для валідацій даних. У MongoDB тригери також присутні і мають схожий фцнкціонал до інших БД.

Підтримка одночасного маніпулювання даними (паралелізм), реалізовано у всіх СУБД, що є великим плюсом для цих СУБД.

Кожна з даних СУБД підтримує одночасний доступ кільком користувачам, що є теж неабиким плюсом. Однак кожна з них має різні ліміти та різне розподілення ресурсів для кількох користувачів, в результаті аналізування цих критеріїв і було виставлено оцінки в таблиці 1. Обробка медіаданих реалізована у всіх СУБД, але в SQL SERVER для цього потрібно вручну створити таблицю, яка буде зберігати не лише сам файл, а й ім’я, щоб не втратити цю інформацію.

Щодо пошуку даних за певним значенням, що має зустрітись в тексті (регулярні вирази), то всі СУБД, є на одному рівні, адже дозволяють гнучно змінювати запити та обробляти їх.

Підртимка ООП, за допомогою мов програмування можна групувати отримані дані в об’єкти або класи.

Найкращі засоби аналізу на мою думку має MongoDB, адже на їх сайті можна переглянути повну інформацію про БД, так як в SQL MS та Postgresql потрібно використовувати певні скрипти, що не є завжди зручним.

Отже можна зробити висновок, що MS SQL найкраще підходить для написання доволі склажнтх проєктів, MongoDb – для невеликих та середніх, а PostgreSql можна використовувати в проєкті будь якого розміру.

**Висновки до першого розділу**

В ході виконання першого розділу було проаналізовано три СУБД також було визначено переваги кожної з СУБД. Також було визначено головні функції програми та приблизну логіку доступу до даних. Також, під час виконання розділу 1, було вирішено виконувати курсовий проект за допомогою СУБД Postgresql. Це дозволить реалізувати БД якісно, надійно та швидко, без необхідності ручного налаштування серверу БД.

**РОЗДІЛ 2. ПРОЕКТУВАННЯ БАЗИ ДАНИХ ПЛАТФОРМИ ДЛЯ ПРОВЕДЕННЯ КУРСІВ**

* 1. **Аналіз структури інформаційних процесів адміністраторів**

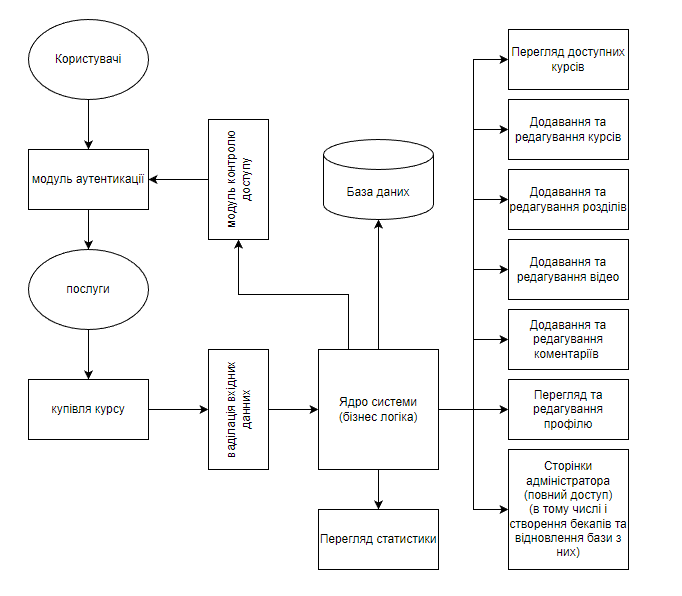


Рис 1. Структура автоматизованої інформаційної системи платформи курсів

Користувачі будуть мати змогу працювати з додатком як після аутентифікації, так і без неї. Наприклад, переглядати деякі дані про курси, коментарі можна буде і без аутентифікації, а ось отримувати якісь більші моливості, наприклад, для звичайного користувача – це буде можливість купівлі курсу, написання коментарію, створення курсу та перегляд куплених чи створених курсів користувача. Адмін матиме повний доступ до зміни будь яких данних, а також до перегляду статистики сайту та створення резурвних копій бази.

Далі наведено приблизні структурні схеми функцій системи.



Рис 2. Структурна схема кількох функцій системи, а саме – реєстрація, логінізація користувача та редагуваня профілю

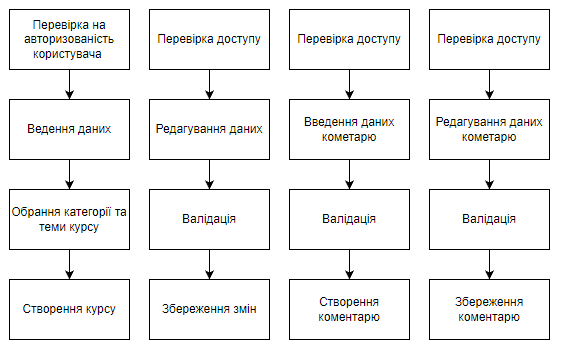


Рис 3. Структурна схема кількох функцій системи, а саме – створення курсу, оновлення курсу, написання коментарію та його редагування

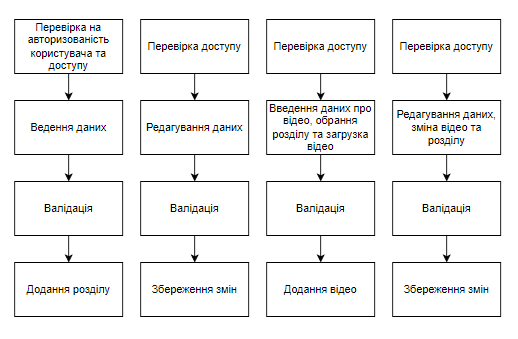


Рис 4. Структурна схема кількох функцій системи, а саме – створення розділу, оновлення розділу, додання відео та його редагування

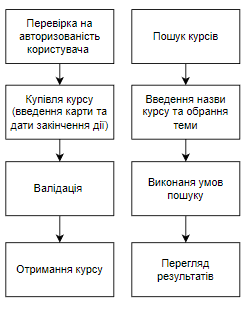


Рис 5. Структурна схема кількох функцій системи, а саме – купівля курсу та пошук курсів

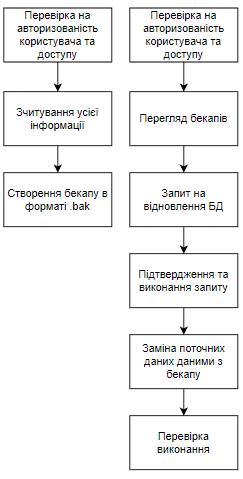


Рис 6. Структурна схема кількох функцій системи, а саме – створення бакапу та відновлення бази

**2.2 Проектування бази даних платформи курсів**

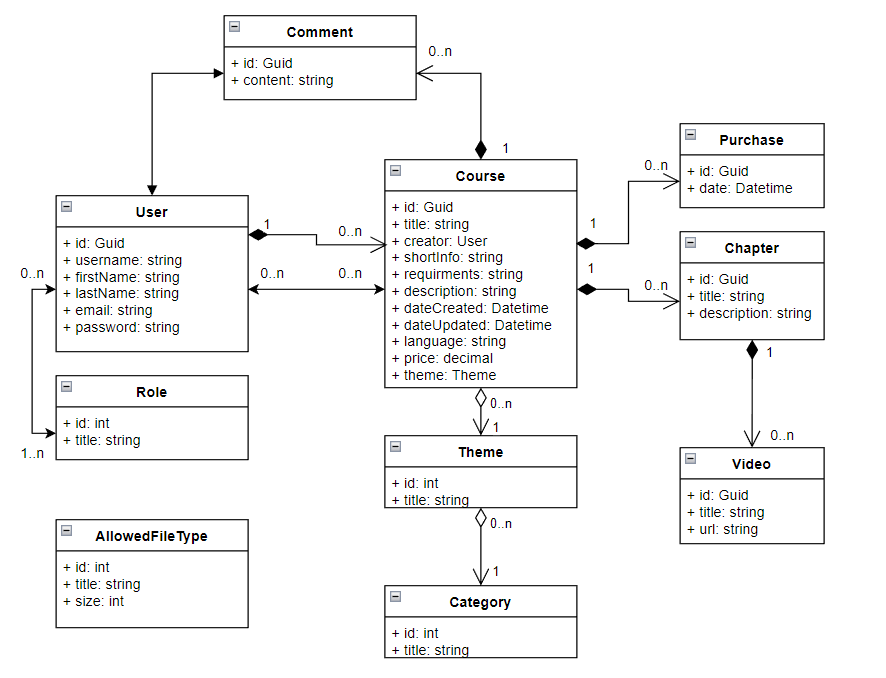


Рис 7. Діаграма класів-сутностей системи

Внаслідок проектування до бази даних включено таблиці:

* Users
* Roles
* Courses
* Themes
* Categories
* Purchases
* Chapters
* Videos
* Comments
* AllowedFileTypes

Для збереження користувачів призначена таблиця "Users ". В дану таблицю вводяться всі необхідні дані користувача. До них належать ім’я, пошта, ім’я, призвіще, псевдонім, рейтинг, хешувальна сіль та хешовавний пароль. Структура таблиці наведена нижче:

Таблиця 2

Структура таблиці “Users”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Назва | Тип даних | ПК | ЗК | Опис поля |
| id | Guid | + | - | Id користувача |
| username | string | - | - | Username користучава |
| firstName | string | - | - | Ім’я користучава |
| lastName | string | - | - | Прізвище користучава |
| email | string | - | - | Email користучава |
| password | string | - | - | Захешований пароль користувача |
| salt | string | - | - | Хешувальна сіль |
| rating | double | - | - | Рейтинг користувача |

Для збереження ролей призначена таблиця "Roles". В дану таблицю вводяться всі необхідні дані ролей. До них належить назва ролі. Структура таблиці наведена нижче:

Таблиця 3

Структура таблиці “Roles”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Назва | Тип даних | ПК | ЗК | Опис поля |
| id | int | + | - | Id ролі |
| title | string | - | - | Назва ролі |

Для збереження курсів призначена таблиця "Courses". В дану таблицю вводяться всі необхідні дані курсів. До них належать назва, посилання на автора, коротка інформація, опис, вимоги, дата створення, дана оновлення, мова курсу, ціна, посилання на тему. Структура таблиці наведена нижче:

Таблиця 4

Структура таблиці “Courses”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Назва | Тип даних | ПК | ЗК | Опис поля |
| id | Guid | + | - | Id курсу |
| title | string | - | - | Назва курсу |
| creatorId | Guid | + | + | Id автора (посилання на таблицю Users) |
| shortInfo | string | - | - | Короткий опис |
| description | string | - | - | Опис |
| requirements | string | - | - | Вимоги |
| dateCreated | DateTime | - | - | Дата створення |
| dateUpdated | DateTime | - | - | Дата оновлення |
| language | string | - | - | Мова курсу |
| price | decimal | - | - | Ціна курсу |
| themeId | int | + | + | Id теми (посилання на таблицю Themes) |
| rating | Double | - | - | Рейтинг курсу |

Для збереження тем призначена таблиця "Themes". В дану таблицю вводяться всі необхідні дані теми. До них належать назва, посилання на категорію. Структура таблиці наведена нижче:

Таблиця 5

Структура таблиці “Themes”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Назва | Тип даних | ПК | ЗК | Опис поля |
| id | Guid | + | - | Id теми |
| title | string | - | - | Назва теми |
| categoryId | int | + | + | Id категорії (посилання на таблицю Categories) |

Для збереження категорій призначена таблиця "Categories". В дану таблицю вводяться всі необхідні дані категорій. До них належать назва. Структура таблиці наведена нижче:

Таблиця 6

Структура таблиці “Categories”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Назва | Тип даних | ПК | ЗК | Опис поля |
| id | Guid | + | - | Id категорії |
| title | string | - | - | Назва категорії |

Для збереження покупок призначена таблиця "Purchases". В дану таблицю вводяться всі необхідні дані покупок. До них належать дата, посилання на курс. Структура таблиці наведена нижче:

Таблиця 7

Структура таблиці “Purchases”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Назва | Тип даних | ПК | ЗК | Опис поля |
| id | Guid | + | - | Id покупки |
| date | DateTime | - | - | Дата покупки |
| courseId | Guid | + | + | Id курсу (посилання на таблицю Courses) |

Для збереження розділів призначена таблиця "Chapters". В дану таблицю вводяться всі необхідні дані розділів. До них належать назва, опис, посилання на курс. Структура таблиці наведена нижче:

Таблиця 8

Структура таблиці “Chapters”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Назва | Тип даних | ПК | ЗК | Опис поля |
| id | Guid | + | - | Id розділу |
| title | string | - | - | Назва розділу |
| description | string | - | - | Опис |
| courseId | Guid | + | + | Id курсу (посилання на таблицю Courses) |

Для збереження відео призначена таблиця "Videos". В дану таблицю вводяться всі необхідні дані про відео. До них належать назва, посилання на відео, посилання на розділ. Структура таблиці наведена нижче:

Таблиця 9

Структура таблиці “Videos”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Назва | Тип даних | ПК | ЗК | Опис поля |
| id | Guid | + | - | Id відео |
| title | string | - | - | Назва відео |
| url | string | - | - | Посилання на выдео |
| chapterId | Guid | + | + | Id розділу (посилання на таблицю Chapters) |

Для збереження коментрарів призначена таблиця "Comments". В дану таблицю вводяться всі необхідні дані коментарів. До них належать коментар , посилання на користувача, посилання на курс. Структура таблиці наведена нижче:

Таблиця 8

Структура таблиці “Comments”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Назва | Тип даних | ПК | ЗК | Опис поля |
| id | Guid | + | - | Id коментаря |
| content | string | - | - | Текс коментаря |
| userId | Guid | - | - | Id користувача (посилання на таблицю Users) |
| courseId | Guid | + | + | Id курсу (посилання на таблицю Courses) |

Для збереження доступних розширень призначена таблиця "AllowedFileTypes". В дану таблицю вводяться всі необхідні дані про дозволені розширення завантажуваних файлів та їх розміри. До них належать назва розширення та розмір в МБ. Структура таблиці наведена нижче:

Таблиця 9

Структура таблиці “AllowedFileTypes”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Назва | Тип даних | ПК | К | Опис поля |
| id | Guid | + | - | Id розширення |
| title | string | - | - | Назва розширення |
| size | int | - | - | Доволений розмір файлу в МБ |

**2.3 Розробка математичної моделі та алгоритмів обробки інформації про платформу курсів**

Для збереження користувача в баз даних, юзеру доведеться заповнити відповідну форму реєстрації, де повнино бути коректно заповнено всі необхідні поля, а сам псевдонім, пошта, прізвище та ім'я (не є обов’язковими) та пароль. Після відправки форму, дані обробляються на бекенді, де додається користувацька роль та перевіряється вільність введеного юзернейму та пошти. Виконання даних операцій користувачу буде доступне операція логіну, де він вводить власні дані, а саме пошту та пароль, і після перевірки корекності даних, надається доступ до користувацького запису. Для оновлення даних потрібно буде перейти до сторінки користувача та змінити дані, там же можна буде змамінити фото профілю.

Після логіну у користувача є 2 шляхи взаємодії з додатком. Перши шлях це переглянути наявні курси та придбати ті, які сподобалися. Для цього користувач має натиснути на відповідну кнопку, та ввести коректні дані карти, а саме номер карти та її срок дії. Це буде забезпечуватися за допомогою інтеграції з платіжною системою, а саме Braintree, яка надає такий функціонал. Для цього потрібно генерувати токен покупки на бекенді та передавати в форму на фронтенді.

Після купівлі курсу, користувач має можливість переглядати матеріал, а саме читати інформацію з розділів та переглядати відео-матеріали. Також матиме можливість залишити відгук про курс, що забезпечить відповідна форма. При написанні коментарю, користувач впливає як на рейтинг самого курсу, так і на рейтинг користувача .Також всі придбані курси користувач матиме змогу переглянути на сторінці профілю.

Іншим шляхом взаємодії буде створення власного курсу. Для цього потрібно заповнити відповідну форму та пройти валідацію. Після чого користувача переадресує на сторінку щойноствореного курсу, де він матиме змогу змнінити дані, за допомогою відповідної форми, додати розділи та відеоматеріали і змінити фото курсу. Всі створені курси користувач також матиме змогу переглянути на сторінці власного профілю.

Всі створенні курси має можливість видалити або їх автор або адмнім, після чого курс буде видалено із всіма розділами, відео та коментарями.

Далі розглянемо функції які доступні тільки адміну, а саме перегляд статистики платформи, додання категорій і тем, а також створення резервних копій бази та відновлення з них.

Для перегляду статистики, адмін повинен буде перейти до відповідної сторінки, де зможе переглянути сптатистику за певний період часу та відношення створених курсів до рейтингу їх авторів. Це досягається виконанням перних групувальних функцій які реалізовані на бекенді.

Для додання категорій, тем та дозволених типів файлів, адмін має перейти на відповідні сторінки та заповнити відповідні форми. Кожна форма має перевірку на унікальність поля назви та валідацію даних.

Також потрібно розглянути те, як буде відбуватипя процес створення бекапів. Для створення бекапу адмін має перейти на сторінку з бекапами та натиснути на відповідну кнопку по створенню бекапу. Далі запит надслиється на бекенд, де він виконується, емулюючи роботу з консоллю та виконуючи відповідні команди для створення бекапу бази даних. Після створення бекапу, адмін побачить його в списку доступних для відновлення.

Для відновлення з резервної копії потрібно обрати бекап та виконати віідповідну команду, запит надсилається на бекенд, де він оброблюється, емулюючи роботу з консоллю та виконуючи відповідні команди для рестору БД. Під час цього процесу база очищається та заповнюється даними з бекапу.

**Висновки до другого розділу**

У результаті виконання даного розділу було проаналізовано інформаційні процеси, спроектовано структуру бази даних, описано приблизну логіку функцій для роботи з базою на клієнті, пояснено деяку логіку обробки, збереження та генерації даних для бази. Також було описано як буде відбуватись побудова та генерація статистичної інформації платформи.

**РОЗДІЛ 3. РЕАЛІЗАЦЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ ПЛАТФОРМИ КУРСІВ**

**3.1. Проектування інтерфейсу системи платформи курсів**

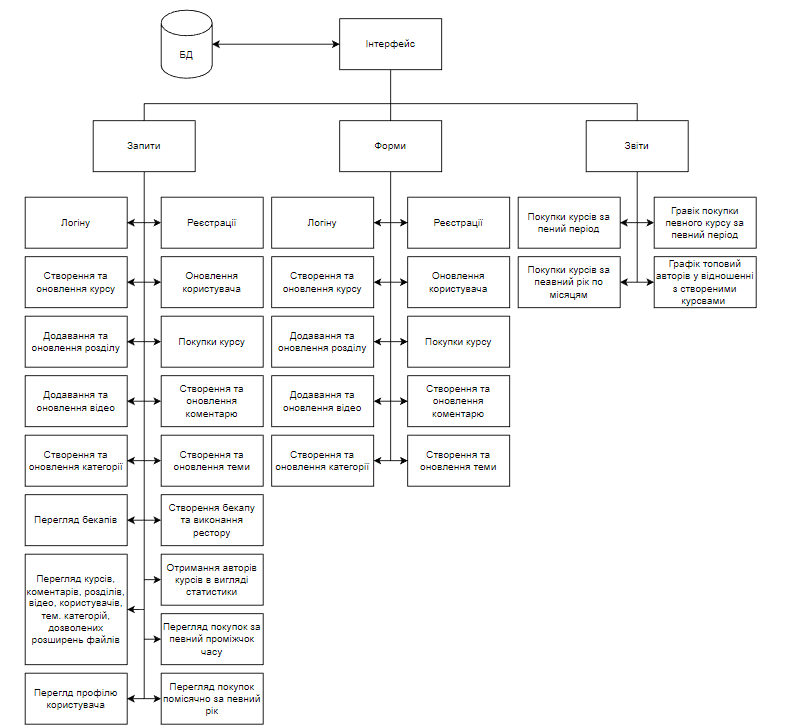
****

Рис 8. Структура інтерфейсу платформи курсів

На рисунку 8 було зображено загальну схему інтерфейсу платформи курсів. Далі будуть зображені форми та сторінки, що показують фінальний результат виконання проектування інтерфейсу додатку. Кожна з форм чи сторінок супроводжується розробкою функцій для роботи з БД.

У більшість випадків побудови форми редагування та додавання були об’єднані з метою зменшення коду. Реакт дозволяє робити умовні побудови первних елементів у компоненті, за рахунок чого і було реалізовано даний підхід розробки інтерфейсу.

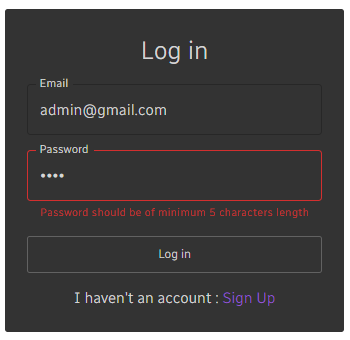


Рис 9. Форма логіну

Форма приймає логін та пароль. При відправці запиту на під’єднання користувача до додатку, відбуається перевірка чи співпадають пароль та логін користувача. Також, пароль має бути більшим ніж 5 символів.

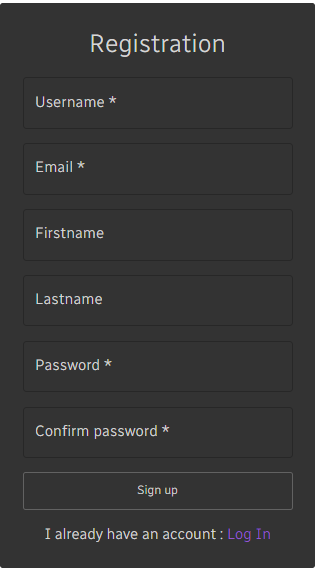


Рис 10. Форма логіну

Форма реєстрації приймає псевдонімі, ім’я, прізвище, пошту, та пароль і повторний пароль. Перераховані поля є обов’язковими, крім імені та прізвища. Перед відправкою форми на сервер відбувається перевірка того, чи співпадають паролі, якщо ж ні, то користувача повідомляють про це і попросять змінити дані, також паролі мають бути довші ніж 5 символів.

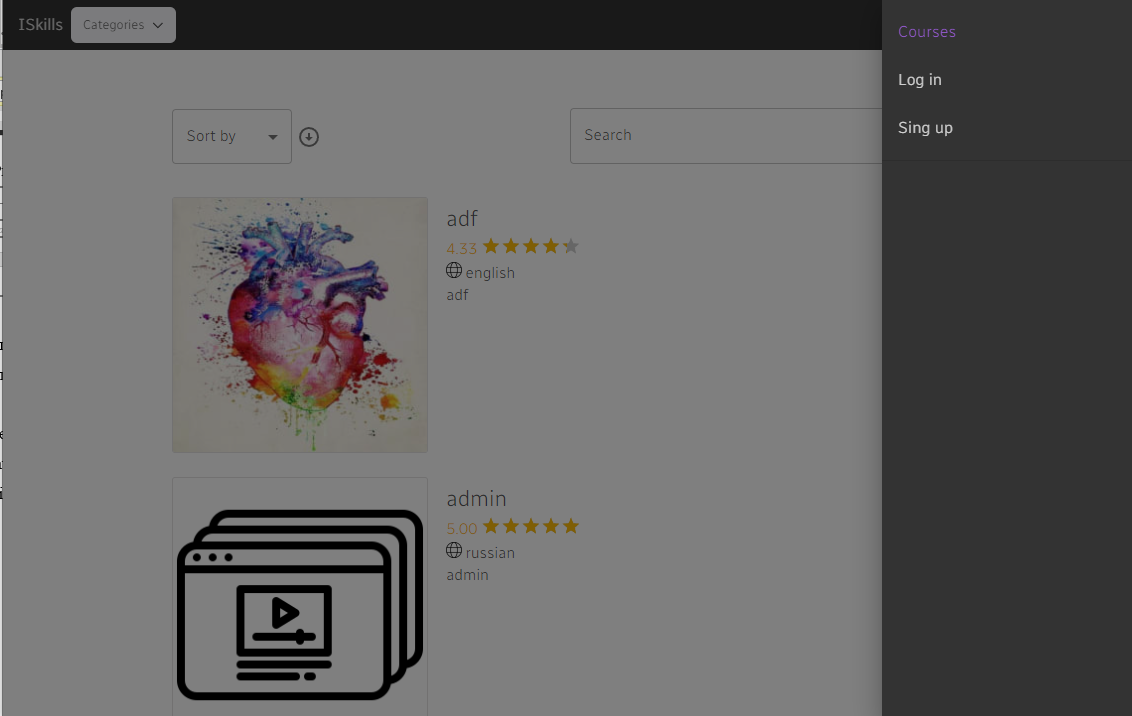


Рис 11. Початкове вікно неавторизованого юзера з боковим меню активованим меню

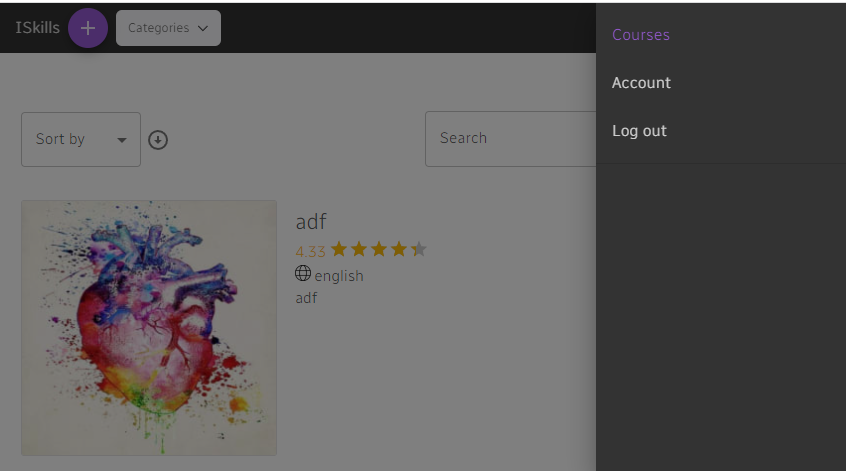


Рис 12. Початкове вікно авторизованого юзера з боковим меню активованим меню

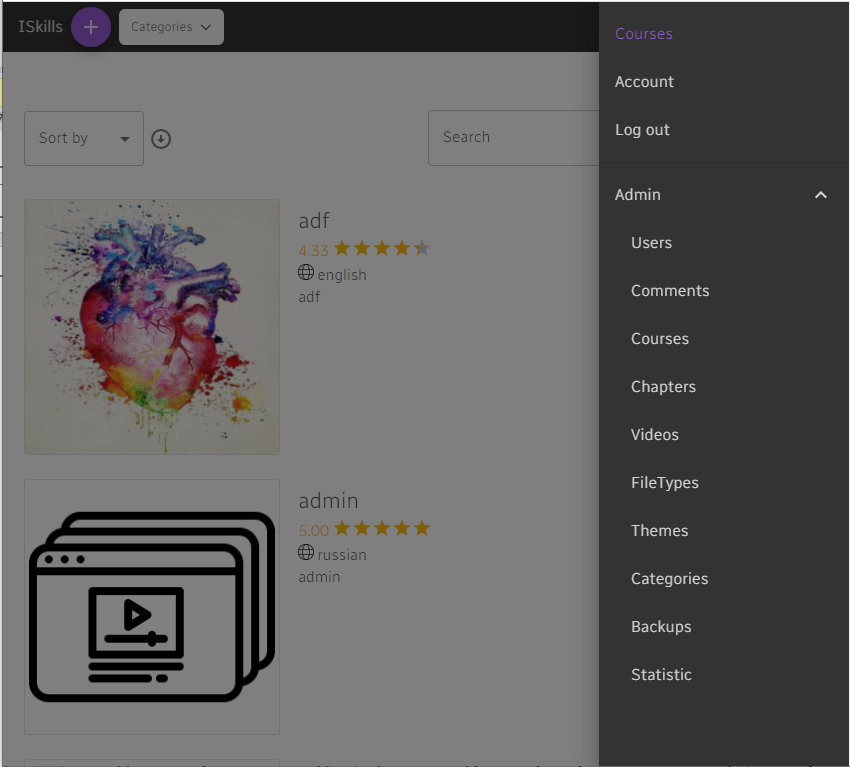
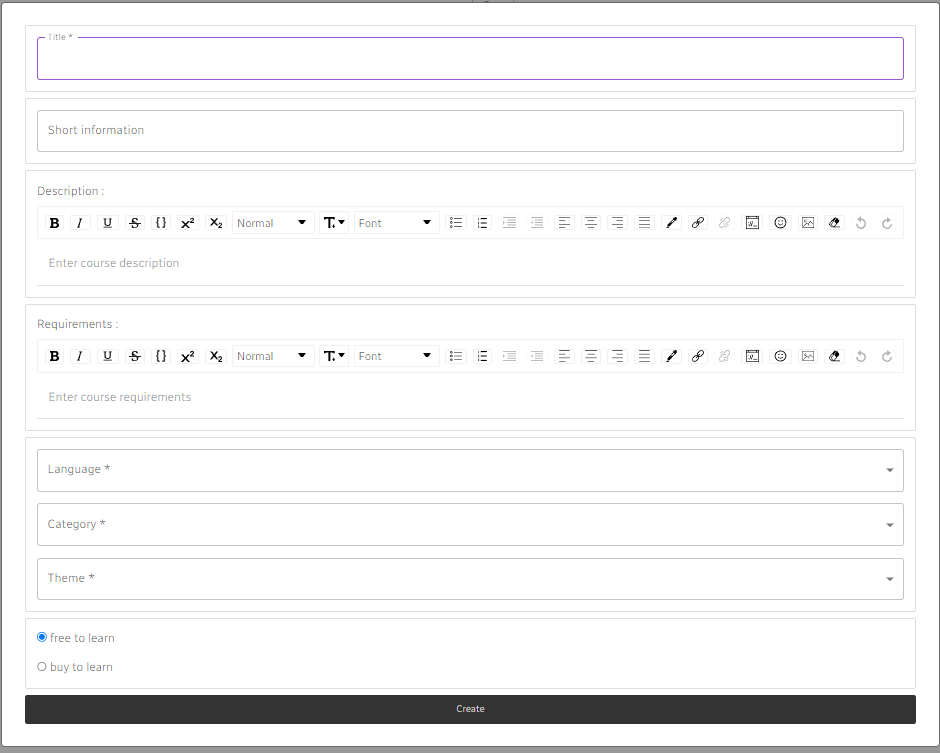


Рис 13. Початкове вікно для адміна з боковим меню активованим меню

З даних рисунків видно, що неавторизований користувач має тільки можливість перегляду курсів та логіну або реєстрації, а уже авторизований може повноцінно взаємодіяти з додатком, тоді як адміністратор має ще додаткові функції, а саме повний контроль над всіми даними в додатку.



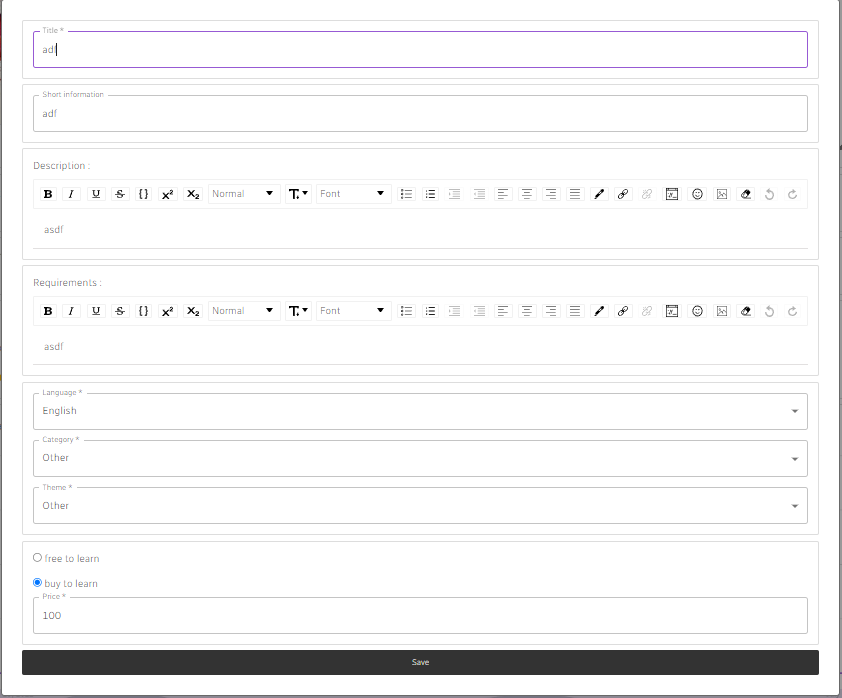


Рис 14-15. Форми створення та оновлення курсу

Дана форма обов’язково приймає назву, мову, категорію, тему та ціну при вказанні, що курс платний. Перед відправкою форми відбуваються перевірки на коректність даних. Також після створення курсу йому додається картинка по замовчування, яку користувач може змінити згодом.

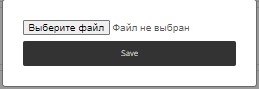


Рис 16. Форми зміни картинки

Дана форма використовується як і на сторінці курсу так і на сторінці користувача для зміни відображуваної картинки відповідного елемента.

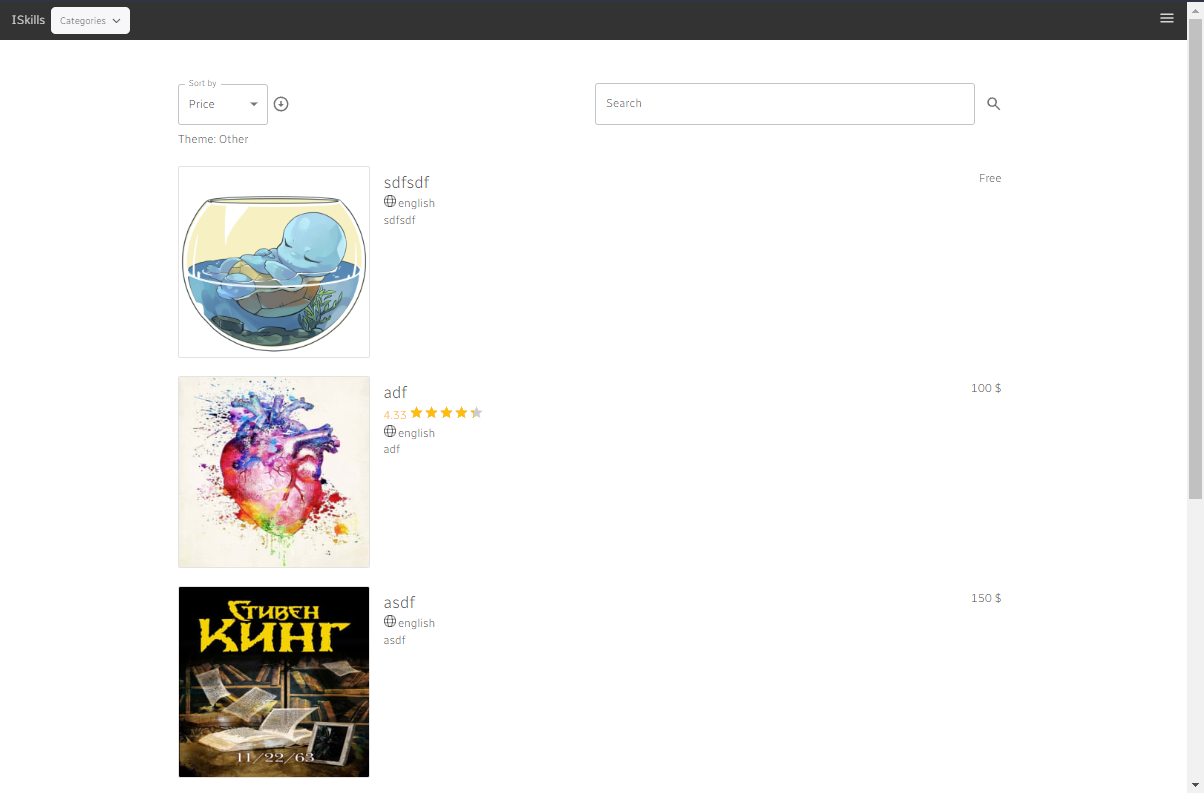
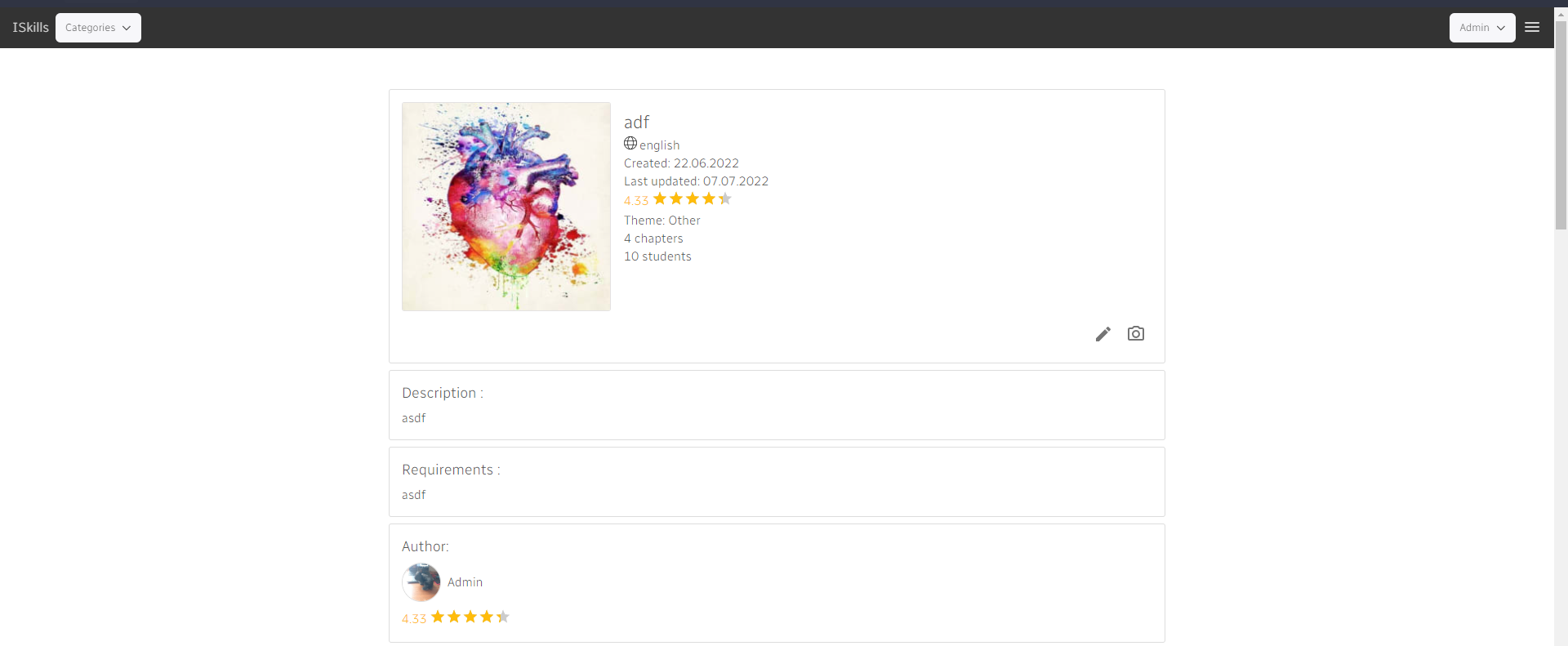
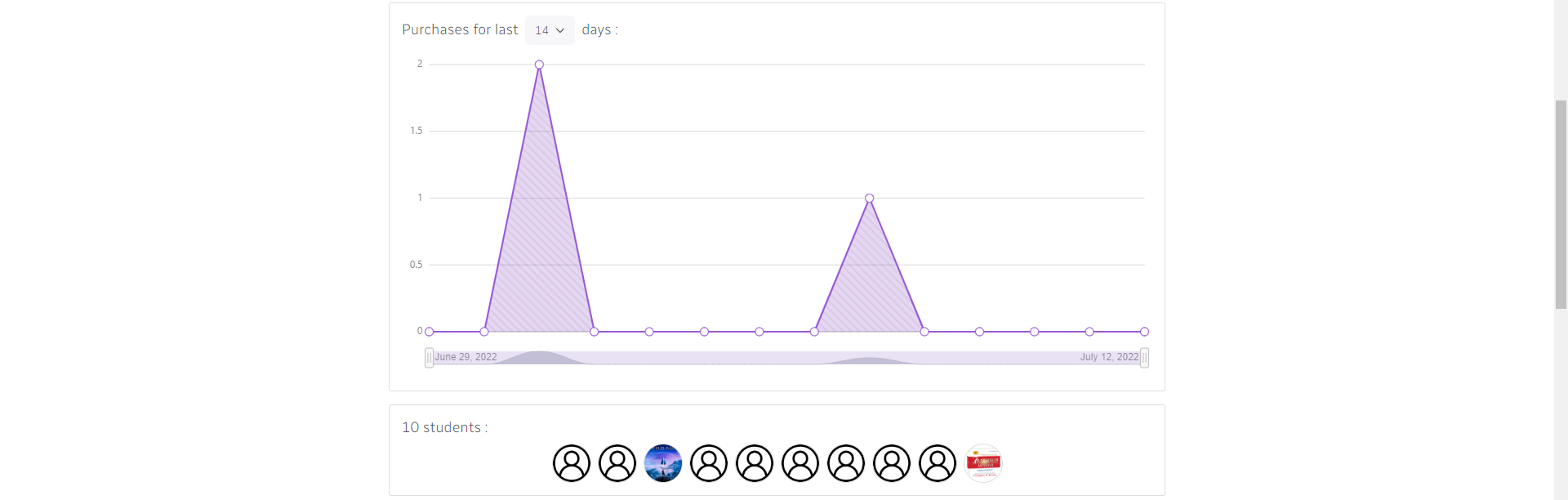


Рис 17. Сторінка для перегляду наявних курсів

На даній сторінці корситувач моде перглядати курси, шукати по їх по назві та по темі одночасно, також має можливість сортувати по деяким параметрам в різному порядку. Також на даній сторінці адмін або автор курсу з може видалити певний курс або курси.





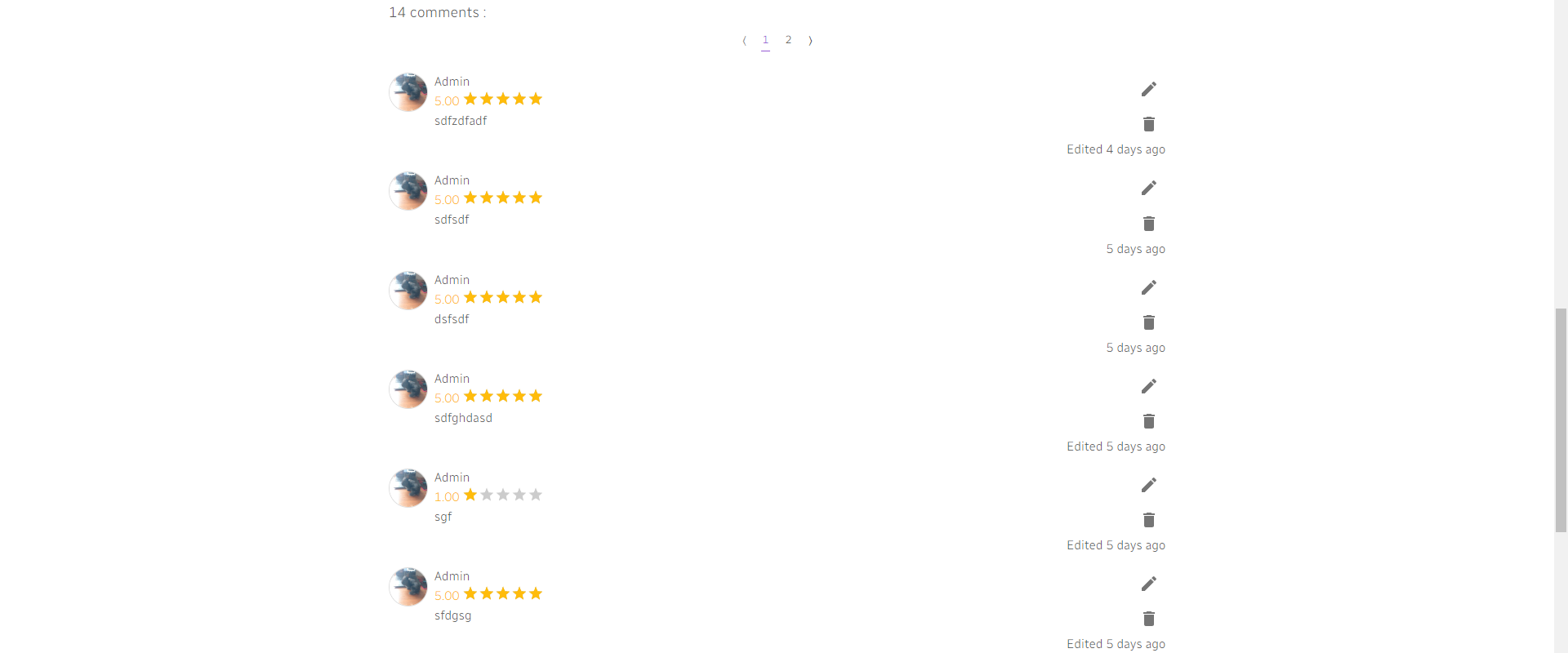
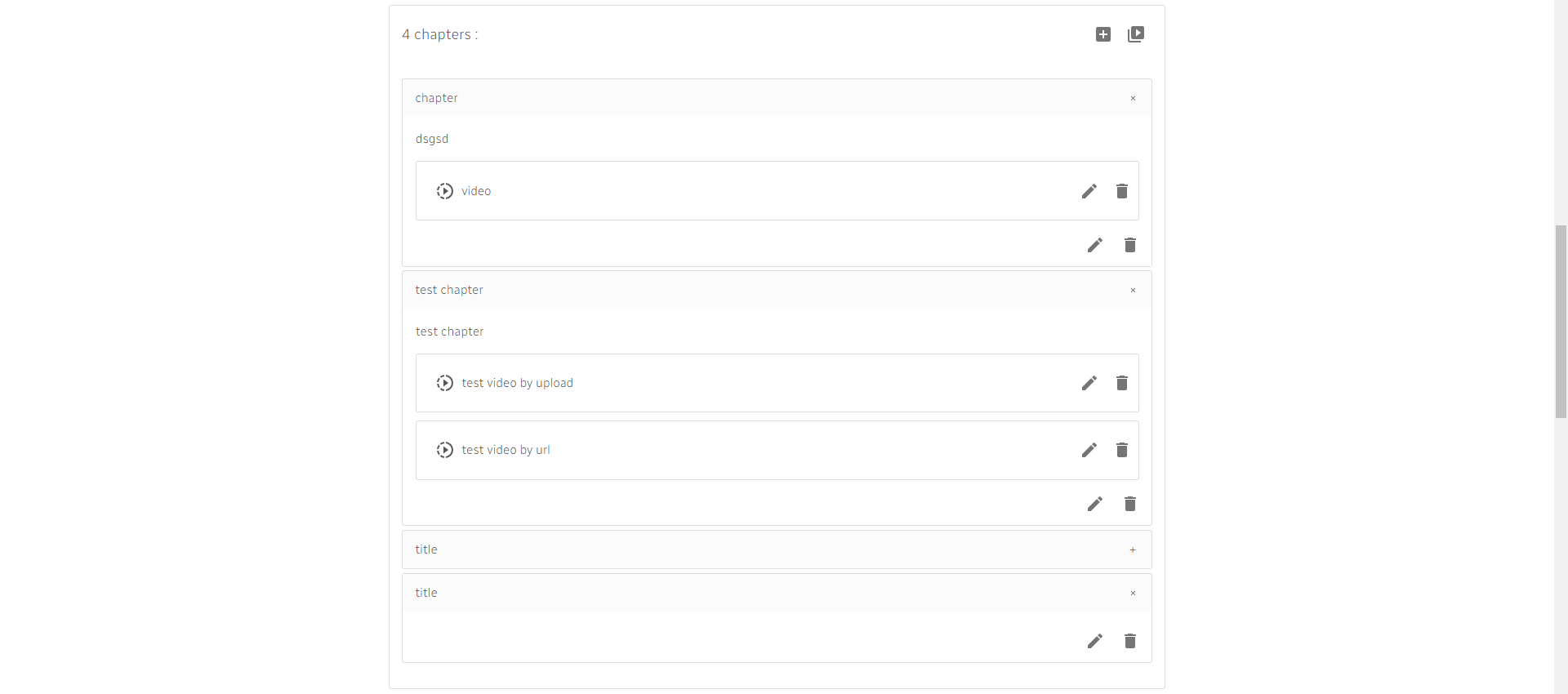
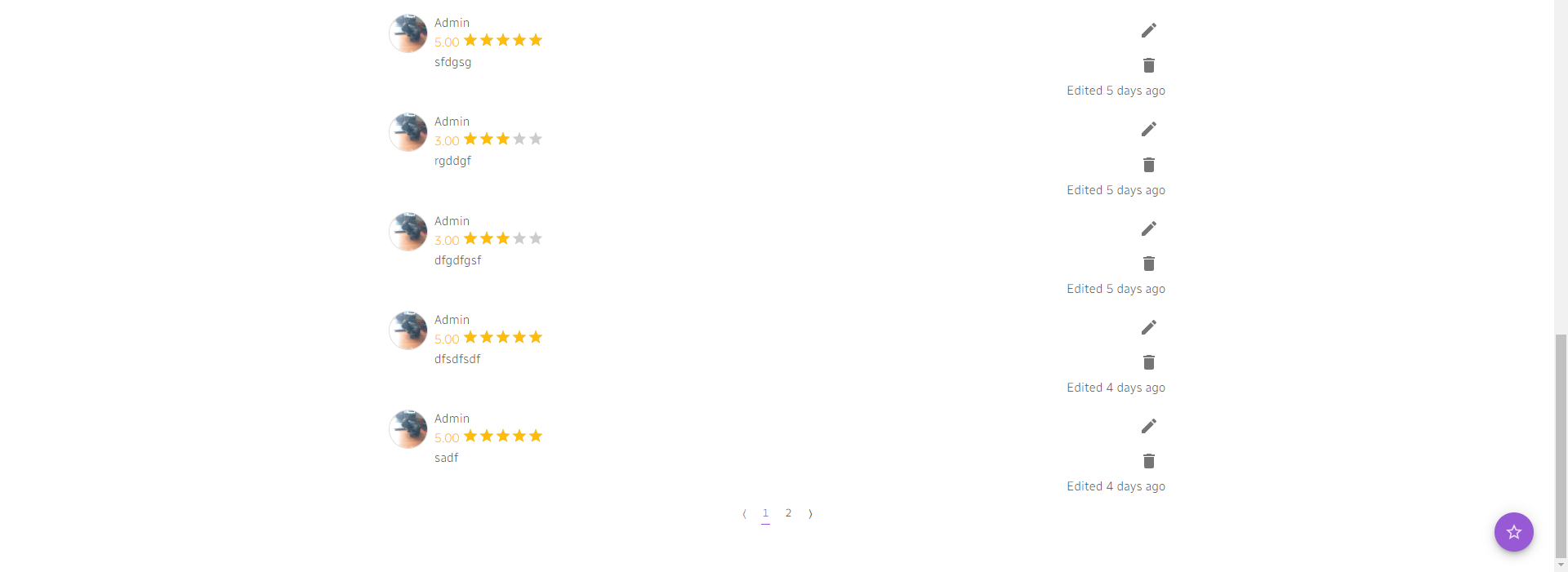
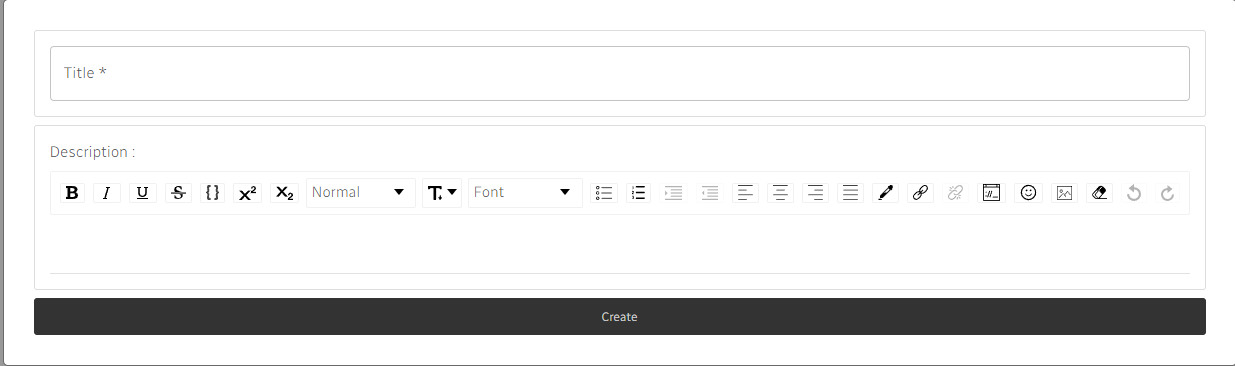
 

Рис (комплексний) 18. Приклад сторінки курсу

На даній сторінці користувач може переглядати матеріали курсу та писти коментарі, а автор курсу може доповнювати новими розділами та відеоматеріалами. адміністратор має такі ж права як і автор тільки ще може видаляти коентарі інших користувачів.



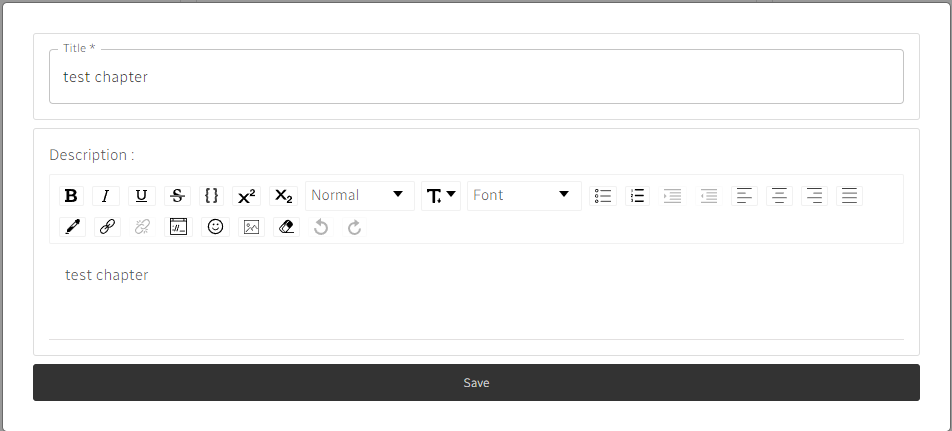


Рис 19, 20. Форми додавання та оновлення розділу

Логіка даних форм схожа до попередньо-розглянутих форм, але в даному випадку обовязковим полем є тільки назва, а опи опціональний.

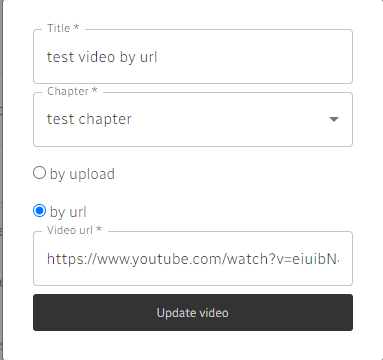
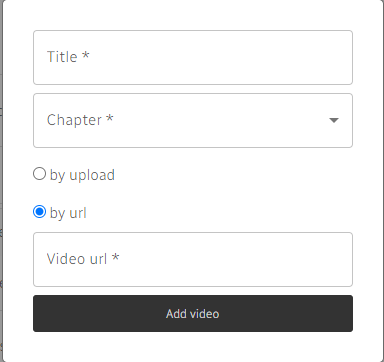


Рис 21, 22. Форми додавання та оновлення відео

Логіка всіх форм схожа, тому вирішено білне не акцентувати на цьому увагу. На даній формі додавання відео автор або адмін повинні ввести коректні дані про назву відео, обрати розділ та або передати url до відео, або загрузити його як файл. Після нажаття на відповідну кнопку виконується додання або оновлення відео. Як бачимо із рисунків, після додання відео, цого можна перемістити в інший розділ.

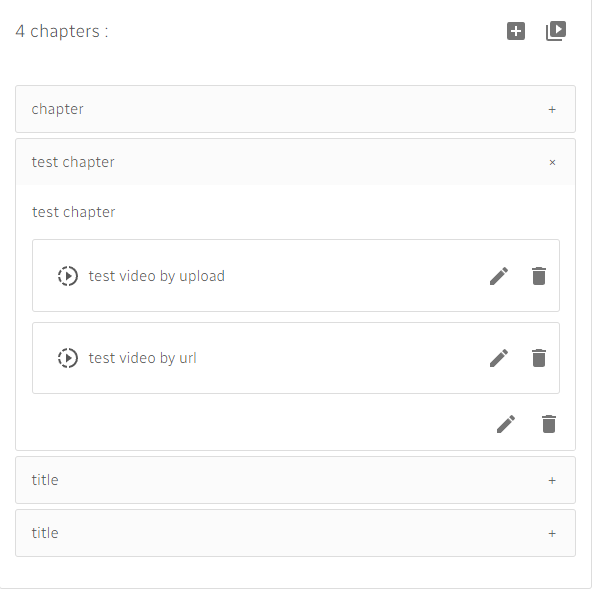


Рис 23. Приклад додавання декількох розділів та відео

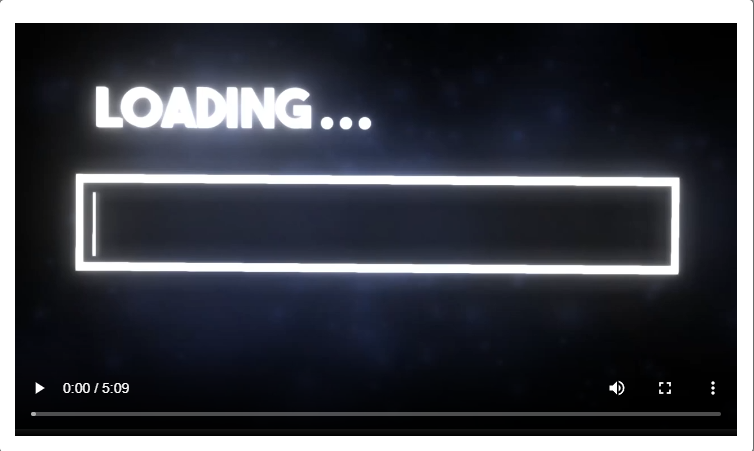
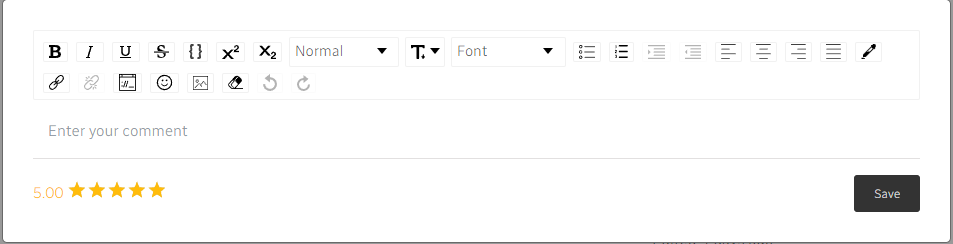


Рис 24. Приклад додавання декількох розділів та відео

На рисунку 23 наведено приклад реальної частини курсу із розділами та відео, які можна переглядати, що і наведено на рисунку 24.



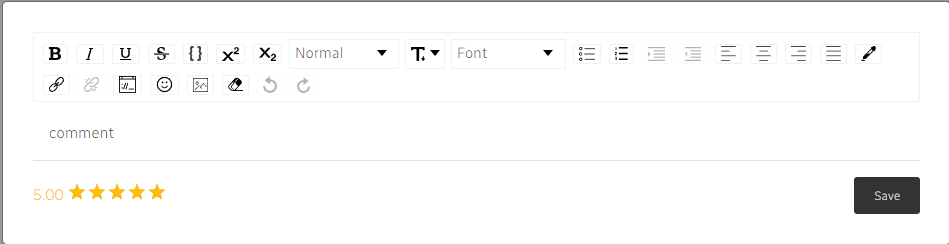
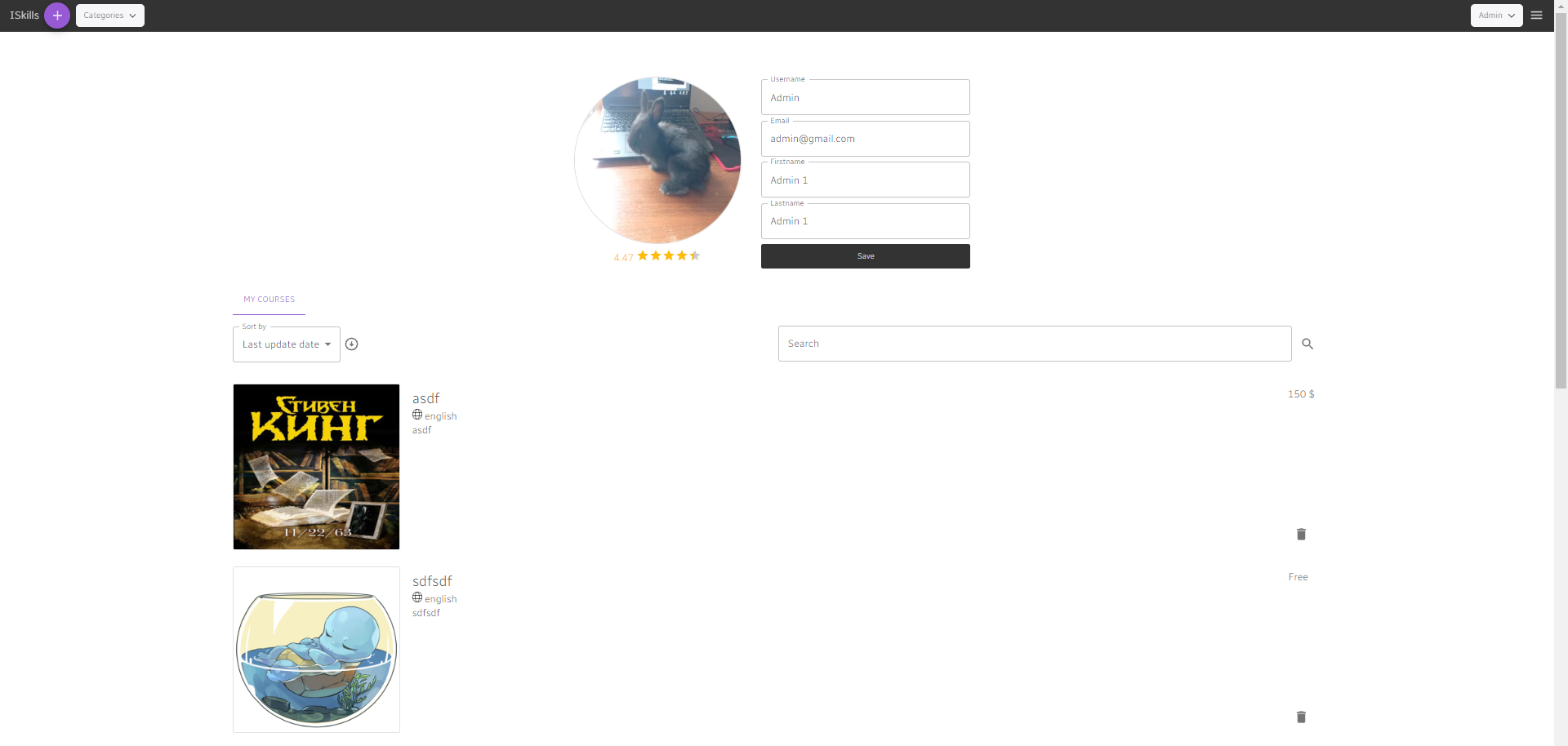


Рис 25, 26. Форми створення та редагування коментарю

В даних формах користувач має заповнити текст коментаря і обрати оцінку курсу, після чого після проходження валідації коментар додається до курсу.



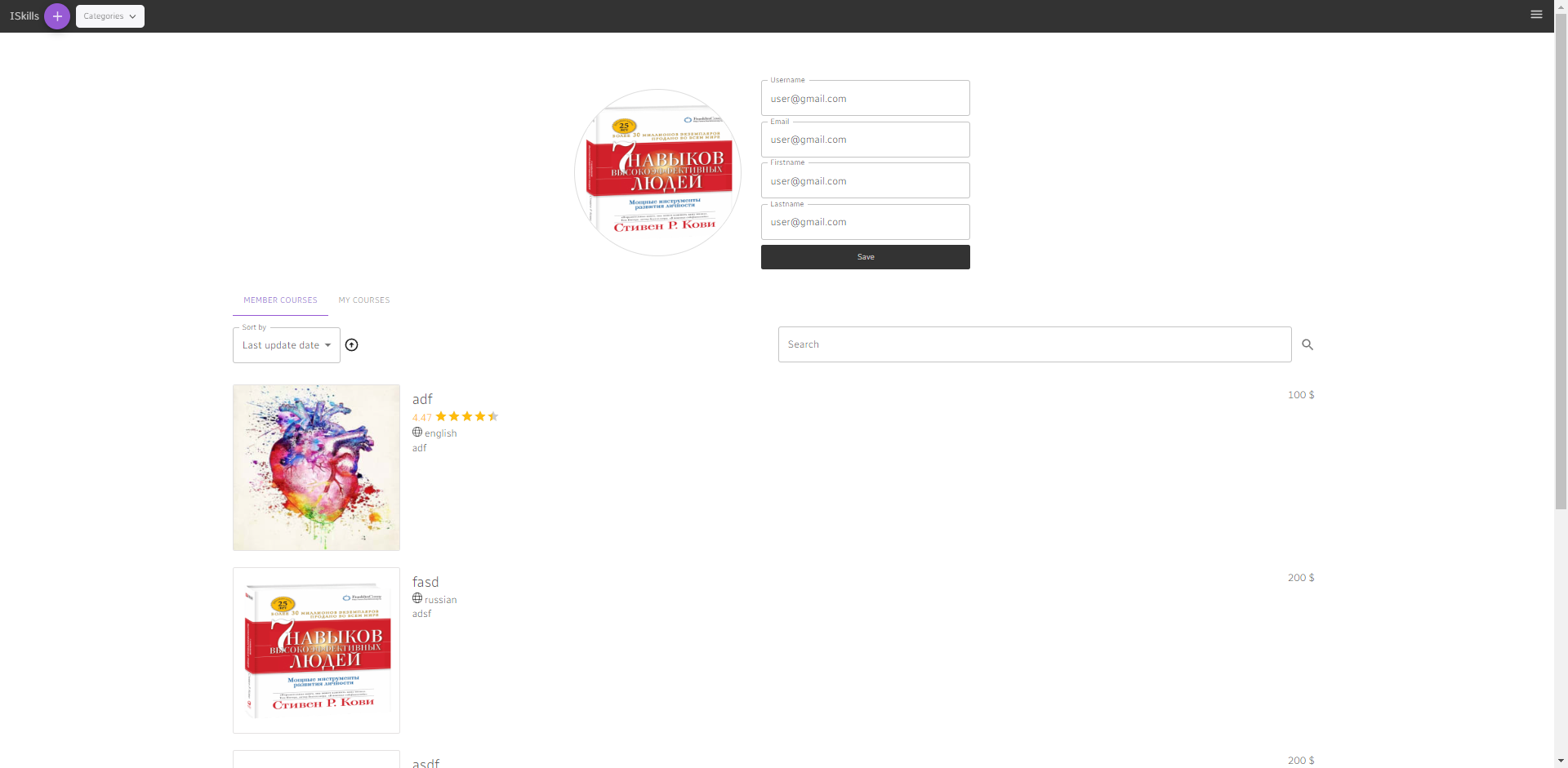
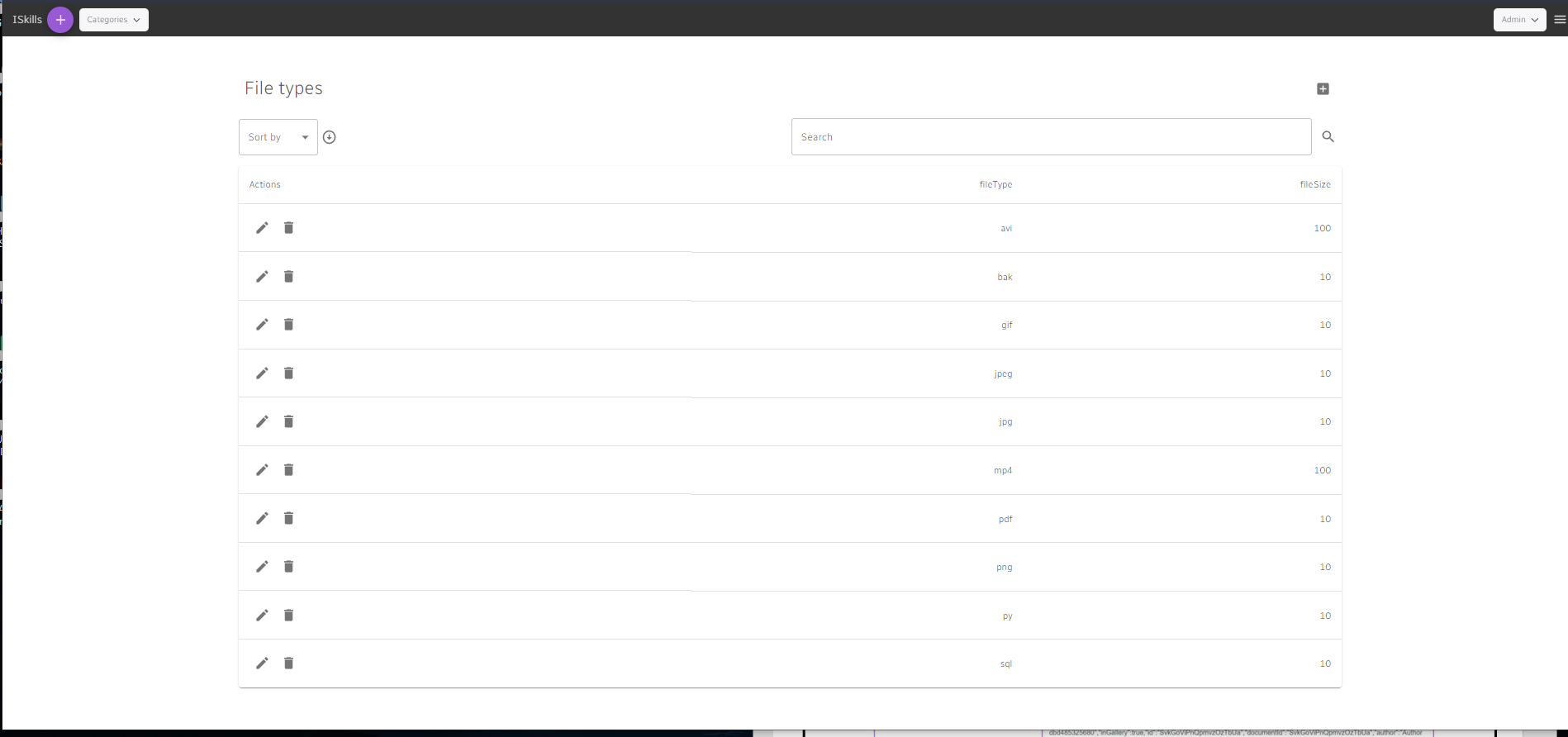


Рис 27, 28. Сторінка профілю адміна та сторінка профілю звичайного користувача

Будь-який корситувач на даній сторінці може змінти інформацію про себе, а також подивитися ті курси які він створив, також можна переглянути ті курси, які були придбані, але адміністратор не має такої можливості, бо він і там має доступ до всіх курсів, тому йому такий функціонал не потрібно, а також адмнім не може рахуватися як струдент курсу. Якщо користувач створить хоча б один курс і хтось його оцінить, то він отримає показник рейтингу в профілі.



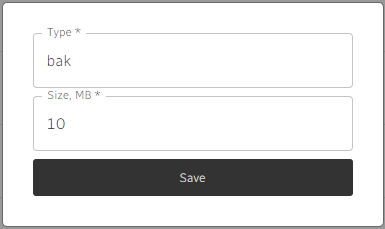
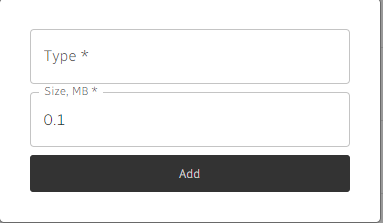


Рис 29, 30, 31. Сторінка дозволених розширень та відповідні форми створення та редагування

На даній сторінці адміністратор може переглянути, які є дозволені для загрузки типи файлів є та їх обмеження на розмір. Також на даній сторінці є відповідні форми, де адміністратор повинен вказати назву розширення та розмір. В даній курсовій роботі використовується хмарний сервіс cloudinary, який дозволяє зберігати медіа-файли, але з пеними обмеженнями, а саме для файлів та картинок це 10 МБ, а для відео – 100 МБ.

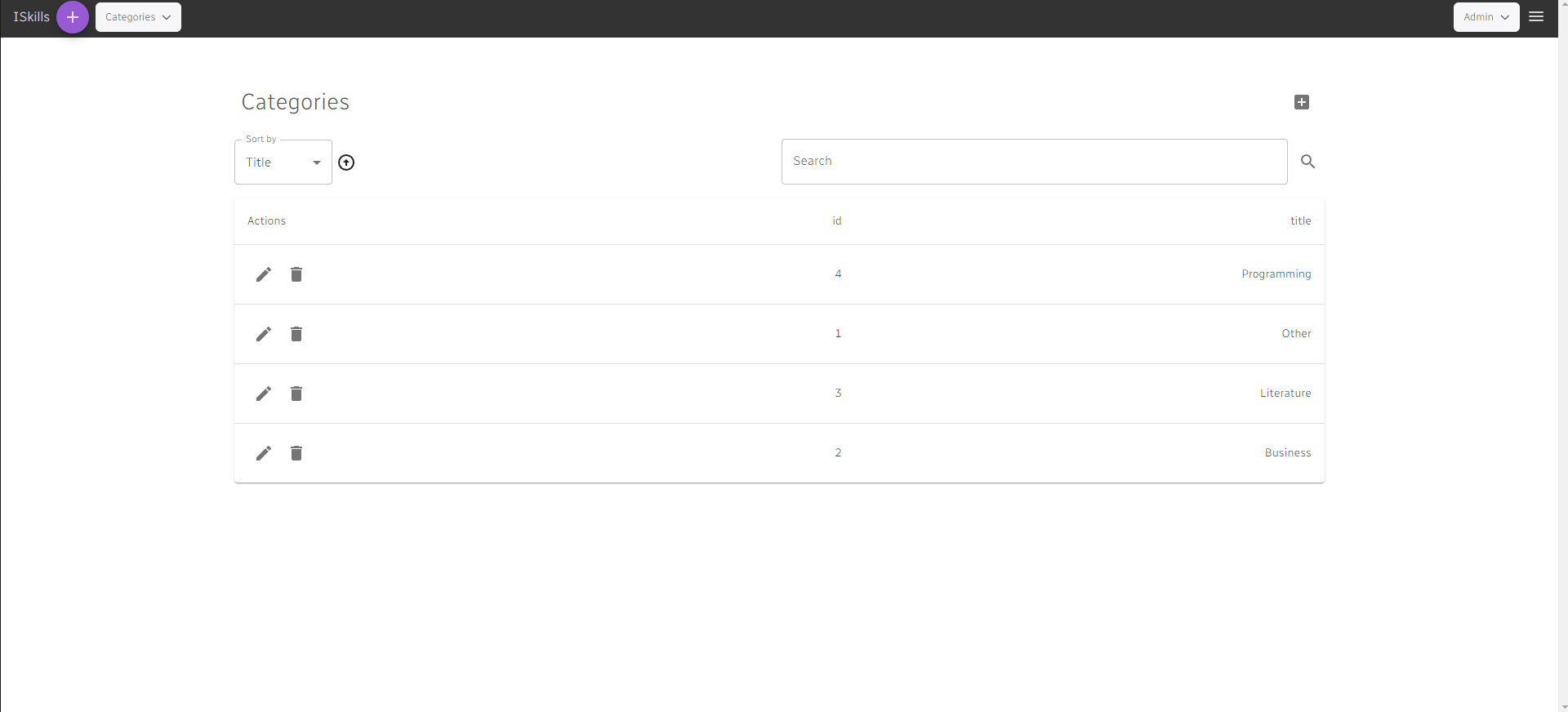


Рис 32. Сторінка категорій

Функціонал даної сторінки аналогічний до попередньої сторінки, за вийнятком контексту про категорії, а у формах потрібно вказувати тільки назву, яка має бути унікальною.

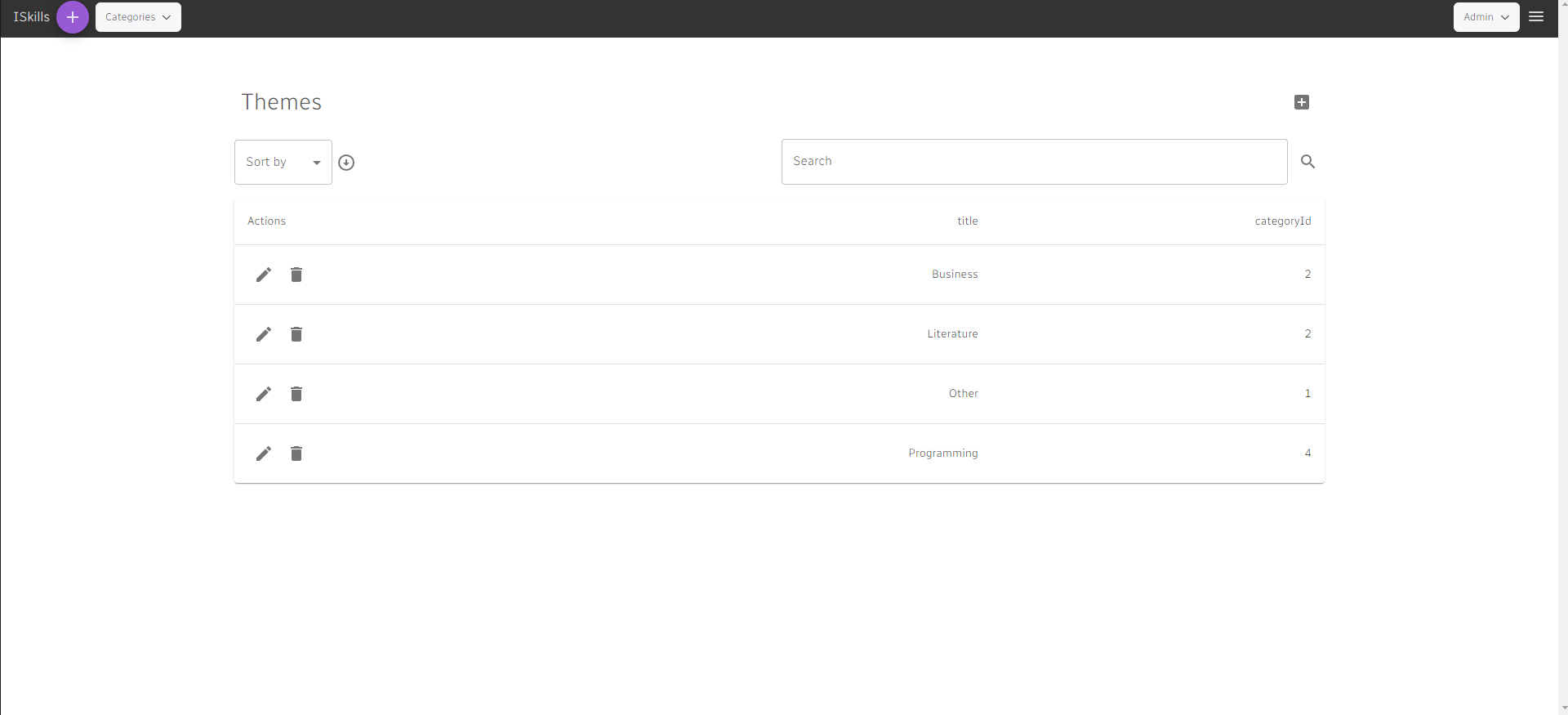


Рис 33. Сторінка тем

Функціонал даної сторінки аналогічний до попередньої сторінки, за вийнятком контексту про теми, а у формах потрібно вказувати назву та обрати категорію, назва також має бути унікальною.

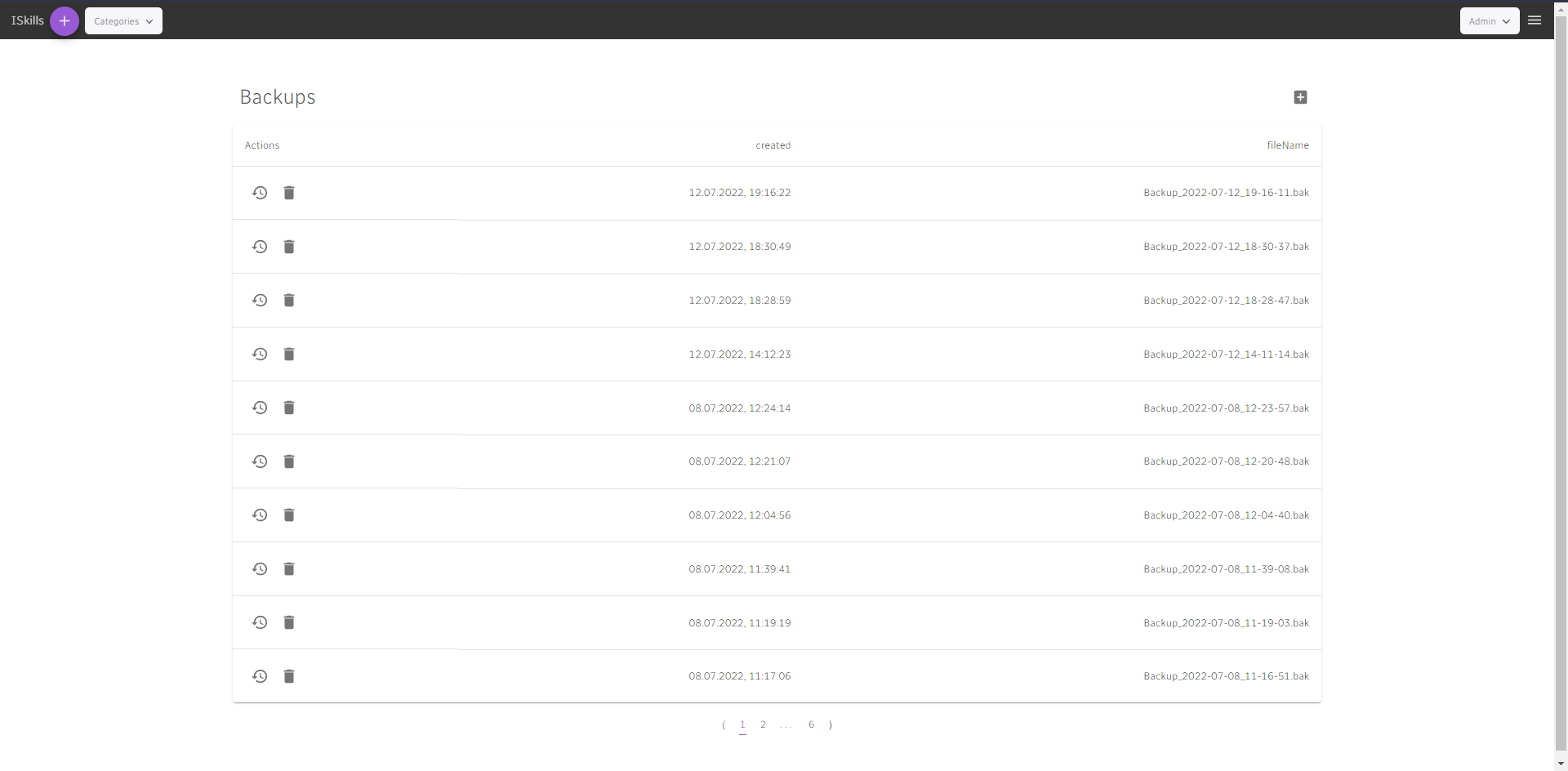
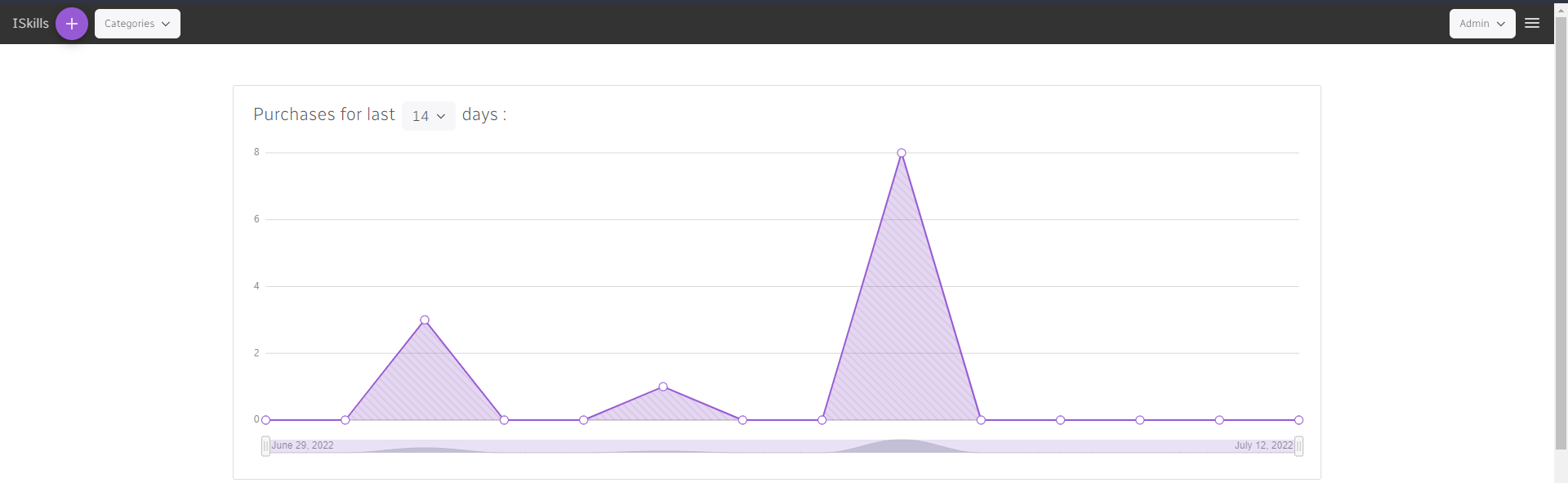
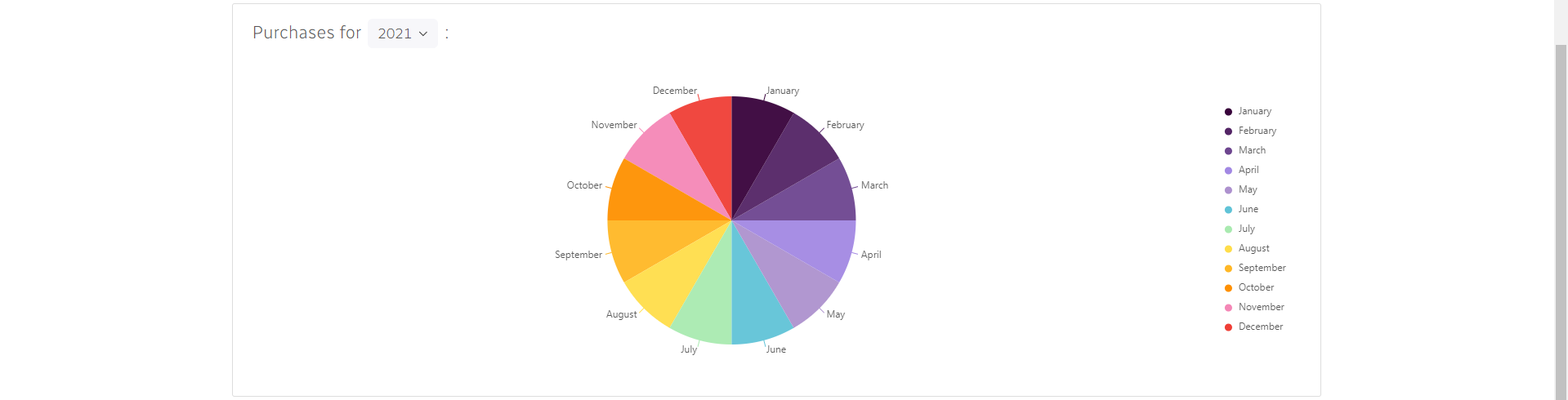


Рис 34. Сторінка бекапів

На даній сторінці адмін може переглянути доступні бекапи, зробити рестор та створити новий бекап.





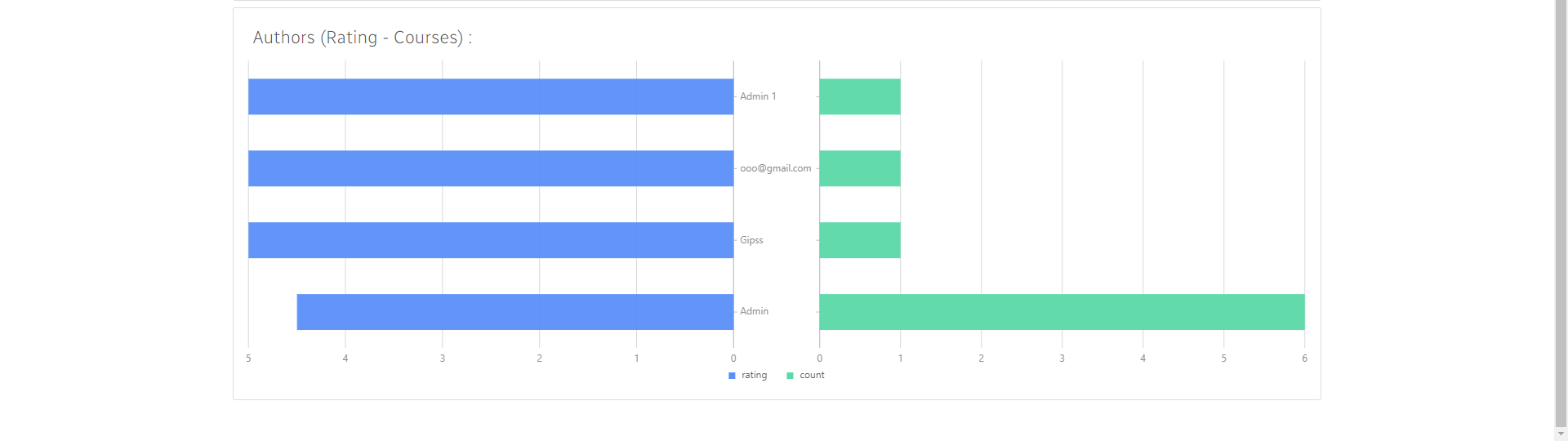


Рис (комплексний) 35. Сторінка статистики

На даній сторінці адміністратор може переглянути покупки за певний період в вигляді графіка, покупки помісячно за певний рік в вигляді кругової діаграми та топових авторів у відношенні до їх курсів у вигдяду дуальної гістограми. Також статистика присутня на сторінці кожного курсу, де будь-який користувач може пеглянути історію покупок за певний період. А також елементом статистики думаю можна вважати рейтинги як курсу, так і користувача.

**3.3. Організація звітності системи**

У даному підрозділі буде розглянуто лише ті звітності, що були розроблені спеціально для додатку, а саме статистика по покупкам за певний період, по покупкам за певний рік по місяцям, по покупка конкретного курсу за певний період і по авторам у відношенні рейтингу та кількості курсів.

Першим методом є метод GetGroupedPurchases. Лістинг наведено нижче :

public async Task<List<DayGroupedItem>> GetGroupedPurchases(string sortOption,

bool reverse, CancellationToken cancellationToken, params object[] dynamics)

{

var courseId = (Guid?)dynamics[0];

var startDate = (DateTime?)dynamics[1];

var endDate = (DateTime?)dynamics[2];

return await \_purchaseDbContext.Purchases.CustomGroupByAsync(

p => (courseId == null || p.CourseId == courseId)

&& (startDate == null || p.Date >= startDate)

&& (endDate == null || p.Date <= endDate),

sortOption,

reverse,

p => p.Date.ToString(),

group => new DayGroupedItem { Day = group.Key.ToString(), Amount = group.Count()},

cancellationToken);

}

Даний метод є асинхронним, він приймає параметер сортування, яким має бути поле таблиці в базі даних, параметер який відповідає за порядок сортування, токен відміни та опціональні параметри, а саме id курсу, щоб групувати тільки конкретного курсу, початкову дату та кінцеву дату, щоб вказати період часу. Даний метод потрібен, щоб отрати статистику за певний період по всім курса чи по конкретному. Сам метод викликає метод який є розширенням для таблиці бази даних, лістинг якого наведено нижче :

public static async Task<List<TResult>> CustomGroupByAsync<T, TResult>(this IQueryable<T> context, Expression<Func<T, bool>> expression, string sortOption, bool reverse,Expression<Func<T, string>> groupByExpression, Expression<Func<IGrouping<string, T>, TResult>> selectExpression,CancellationToken cancellationToken = default)

where T : class

where TResult : class

=> await context

.Where(expression)

.OrderBy(sortOption, reverse)

.GroupBy(groupByExpression)

.Select(selectExpression)

.AsNoTracking()

.ToListAsync(cancellationToken);

Даний метол приймає вираз пошуку, поле сортування, порядок сортування, вираз групування, вираз вибірки після групування та токен відмініми, який є необовязковим. В ході виконання даного методу, спочатку в відповідній таблиці бази даних відбираються записи, які відповідають заданому виразу, потім відбувається сортування по переданому полу та порядку, далі ці записи групуються по виразу групування, а вже потім вони конвертуються в потрібний обєкт. Даний метон написаний так, що його можна було використовувати для будь-якої таблиці БД.

Далі розглянемо метод для отримання статистики за певний рік згрупований по місяцям, а саме GetGroupedYearPurchases, лістинг якого наведено нижче :

public async Task<List<MonthGroupedItem>> GetGroupedYearPurchases(string sortOption,

bool reverse, int year, CancellationToken cancellationToken, params object[] dynamics)

{

var courseId = (Guid?)dynamics[0];

return await \_purchaseDbContext.Purchases.CustomGroupByAsync(

p => (courseId == null || p.CourseId == courseId) && p.Date.Year == year,

sortOption,

reverse,

p => p.Date.Month.ToString(),

group => new MonthGroupedItem { Month = group.Key.ToString(), Amount = group.Count() },

cancellationToken);

}

Як можемо бачити даний метод схожий до попереднього, бо викликає тей самий метод розширення але з іншими параметрами, що є великим плюсом, бо так уникається дублювання коду. Даний метод приймає замість дат номер року і тоді уже відбирає записи, які були зроблені в даному році і групує по місяцям.

Далі розглянемо метод для отримання статистики по авторам, а саме GetTopUsers, лістинг якого наведено нижче :

public async Task<List<UserStatisticModel>> GetTopUsers(CancellationToken cancellationToken)

=> await \_userContext.Users

.Where(u => \_courseDbContext.Courses.Any(c => u.Id == c.CreatorId))

.OrderBy("rating", true)

.Select(x => new UserStatisticModel

{

UserName = x.UserName,

Count = \_courseDbContext.Courses

.Where(c => x.Id == c.CreatorId).Count(),

ImageUrl = x.ImageUrl,

Email = x.Email,

Rating = x.Rating

})

.AsNoTracking()

.ToListAsync(cancellationToken);

Даний метод відбирає користувачів, які створювали курси, та сортує їх за рейтингом та відобрає кількість створених курсів цими корисувачами, що дає змогу зробити дуальну гістограму.

Всі статистичні методи також доповнюються на фронтенді, так як якщо в базі даних немає записів, наприкла за конкретний день, то і даних про цей день не буде, для цього потрібно доповнювати графіки на фронтенді.

**Висновки до третього розділу**

У ході виконання даного розділу відбулось написання повного коду для взаємодії користувача з БД. Було спроектовано інтерфейс системи платформи курсів. Були розроблені форми для створення курсів, користувачів, категорій, тем, коментарів, дозволених типів файлів з обмеженнями розміру, розділів, відео та їх редагувань. Також були створені сторінки з можливістю перегляду, виконання запитів на фільтрації даних, виконання запитів на підгрузку даних. Крім цього, було створено можливість зберігати поточний стан бази, та відновлювати його з резервної копії його.

Також при виконання даного розділу було написано та описано логіку та код виконання звітності про систему за кількома типами, а саме – отримання покупок за певний період або за рік згрупованим по місяцям по всім курсам платформи або по конкретному курсу, а також статистика по авторам курсів, а саме відношення рейтингу до кількості курсів.