

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Інститут інформатики та радіoeлектроніки,
Факультет комп'ютерних наук і технологій
(повне найменування інституту, назва факультету)

Кафедра програмних засобів
(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

бакалавра

(ступінь вищої освіти)

на тему ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ПІДТРИМКИ ВИВЧЕННЯ
ПЕРСОНАЛІЗОВАНИХ КУРСІВ

SOFTWARE FOR SUPPORTING THE STUDY IN PERSONALIZED COURSES

Виконав: студент(ка) 4 курсу, групи КНТ-217

Спеціальності 122 Комп'ютерні науки

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Комп'ютерні науки

Мертвіщев Д.В.

(прізвище та ініціали)

Керівник Скрупський С.Ю.

(прізвище та ініціали)

Рецензент Зеленьова І.Я.

(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
 (повне найменування закладу вищої освіти)

Інститут, факультет ІРЕ, ФКНТ
 Кафедра програмних засобів
 Ступінь вищої освіти бакалавр
 Спеціальність 122 Комп'ютерні науки
(код і найменування)
 Освітня програма (спеціалізація) Комп'ютерні науки
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
С.О. Субботін
 “ ” 2021 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Мертвіщева Дениса Валерійовича
(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Програмне забезпечення підтримки вивчення
персоналізованих курсів
Software for Supporting the Study in Personalized Courses

керівник проєкту (роботи) Скрупський Степан Юрійович, к.т.н., доцент,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від “30” березня 2021 року № 103

2. Строк подання студентом проєкту (роботи) травень 2021 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне
завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Розробка архітектури програми.
3. Розробка програми. 4. Експлуатація та тестування програми.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	Скрупський С.Ю., доцент		
Нормоконтролер	Андрєєв М.О., асистент		

7. Дата видачі завдання “9” березня 2021 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області	2-3 тижні	Розділ 1
3	Розробка архітектури програми	4 тиждень	Розділ 2
4	Розробка програми	5-6 тижні	Розділ 3
5	Тестування та експериментальне дослідження програми	7 тиждень	Розділ 4
6	Оформлення пояснювальної записки та документів до неї. Нормоконтроль та рецензування	8 тиждень	Додатки
7	Захист роботи	9 тиждень	

Студент(ка)

(підпис)

Мертвіщев Д.В.
(прізвище та ініціали)

Керівник проєкту (роботи)

(підпис)

Скрупський С.Ю.
(прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 86 с., 30 рис., 1 табл., 3 дод., 17 джерел.

ОНЛАЙН КУРСИ, ОНЛАЙН НАВЧАННЯ, ПЕРСОНАЛІЗАЦІЯ, ВЕБЗАСТОСУНОК, ВЕБФРЕЙМВОРК.

Об'єкт роботи – процес вивчення онлайн курсів.

Предмет роботи – програмне забезпечення підтримки вивчення персоналізованих курсів.

Мета роботи – розроблення програмного забезпечення підтримки вивчення персоналізованих курсів для забезпечення індивідуального способу роботи з матеріалами з курсами, що має підвищити результативність навчання.

Матеріали, методи та технічні засоби: мова програмування Python, об'єктно-орієнтоване програмування, структурне програмування, вебфреймворк Django.

Результати. Описано та обгрунтовано рішення, прийняті під час аналізу предметної області, проєктування та реалізації програмного забезпечення.

Висновки. Розроблене програмне забезпечення підтримує процес онлайн навчання з можливістю створення персоналізованої подачі відеоматеріалів і текстових матеріалів, визначаючи персоналізацію за результатами проведених тестів та заповнених учнем анкет за допомогою логічних правил. Залишаючи відгуки на курси після завершення навчання, учні впливають на вибір наступних користувачів, що підбирають собі курси для вивчення.

Галузь використання – онлайн навчання без прив'язування до закладу освіти.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 86 pages, 30 figures, 1 table, 3 appendixes, 17 sources.

ONLINE COURSES, ONLINE TRAINING, PERSONALIZATION, WEB APPLICATION, WEBFRAMEWORK.

The object of work is the process of studying online courses.

Subject of work is software for support the study of personalized courses.

The purpose of the work is to develop software to support the study of personalized courses to provide an individual way of working with materials with courses, which should increase the effectiveness of training.

Materials, methods and technical means: Python programming language, object-oriented programming, structural programming, Django web framework.

Results. The decisions made during the analysis of the subject area, design and implementation of software are described and grounded.

Conclusions. The developed software supports the process of online learning with the possibility of creating personalized presentation of videos and text materials, determining personalization based on the results of tests and questionnaires completed by the student using logical rules. Leaving feedback on courses after graduation, students influences the choice of subsequent users who choose courses to study.

Field of usage is online learning without being tied to an educational institution.

ЗМІСТ

	С.
Перелік скорочень та умовних познач.....	7
Вступ.....	8
1 Аналіз предметної області.....	10
1.1 Онлайн курси.....	10
1.2 Огляд програмних аналогів.....	11
1.3 Висновки за розділом 1	18
2 Розробка архітектури програми.....	20
2.1 Вибір мови програмування	20
2.2 Вимоги до програмного забезпечення	21
2.3 Проектування структури даних	25
2.4 Висновки за розділом 2	31
3 Розробка програми	32
3.1 Послідовність персоналізації курсів	32
3.2 Розроблені класи	35
3.3 Висновки за розділом 3	43
4 Експлуатація та тестування програми	44
4.1 Призначення програми	44
4.2 Умови виконання програми	44
4.3 Робота з програмою	44
Висновки	54
Перелік джерел посилання	55
Додаток А Технічне завдання	57
Додаток Б Текст програми	62
Додаток В Слайди презентації.....	79

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

БД – база даних;

ПЗ – програмне забезпечення.

ВСТУП

Актуальність будь-яких центральних питань, пов'язаних з освітою, не викликає сумніву, адже з питаннями освіти людина зіштовхується майже з початку свого життя: позашкільні заклади, заклади середньої освіти та вищої освіти. На даний момент одним з найважливіших питань освіти є організація онлайн навчання. Онлайн навчання отримало додатковий поштовх до розвитку за рахунок кризи, пов'язаної з пандемією коронавірусу. Дистанційне навчання стало основою сучасного навчання, відповідно засоби організації онлайн курсів є одним з таких напрямків.

Необхідно розуміти, що онлайн курси – це не лише засіб організації вищої або середньої освіти, онлайн курси можуть існувати і без даної форми, адже питання навчання перед сучасною людиною виникають постійно. Отримання додаткової освіти, розширення власних знань, актуалізація знань – все це потребує навчання у зручній формі для людини. Саме такою формою у сучасному світі і є онлайн курси.

Важливим для організації онлайн курсів є програмне забезпечення (ПЗ), яке має дозволяти якісно і зручно виконувати цю діяльність: створювати курси, проходити їх. Але окрім того важливим також питанням у сучасному світі є питання персоналізації курсів, адже кожна людина потребує певних особливих матеріалів, адже у кожної людини є свої проблеми під час навчання, відповідно використання єдиного підходу може призвести до того, що результативність освіти може бути зниженою. З іншого боку можна запропонувати учню (студенту) особливі матеріали, яких він потребує, якщо він виявив неуспішні результати під час вивчення деяких питань, з іншого боку учню, який таких проблем не має, цими матеріалами можна не перевантажувати процес навчання. Наявність широкого колу матеріалів, у тому числі відеоматеріалів, текстових матеріалів, дозволить використовувати базову модель і додавати до неї особливі матеріали з

додаткового пулу, в залежності від того, які результати навчання продемонстрував учень (студент) раніше.

Зважаючи на актуальність проблеми, метою роботи є розроблення ПЗ підтримки вивчення персоналізованих курсів для забезпечення індивідуального способу роботи з матеріалами з курсами, що має підвищити результативність навчання.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Онлайн курси

Є безліч причин, чому люди навчаються: бажання зробити наступний крок у кар'єрі або готовність до зміни кар'єри, здобуття наукового ступеня, при цьому якою б не була причина, навчання наближає людину на крок до реалізації її цілей, раніше потрібно було далі тільки вирішити де і чому навчатися, сьогодні студентам також потрібно вирішити як, тобто у якій формі, навчатися, адже онлайн навчання – одна з найбільших революцій у сучасній освіті [1].

На сьогодні онлайн школа може означати декілька речей, починаючи від інтегрованих у технологічні класи аудиторій і закінчуючи набором декількох кредитів в Інтернеті або взагалі віртуальним студентом, при цьому важливо відзначити, що доступ до денної онлайн освіти став чудовою можливістю для незліченної кількості студентів та сімей:

- сім'ї військових можуть дати своїм дітям якісну освіту, незважаючи на необхідність постійних переїздів;

- діти з особливими потребами можуть працювати у своєму власному темпі, не боючись відсутності задоволення власних фізичних потреб або знущань;

- спортсмени, які часто змушені подорожувати, легше можуть переслідувати свої пристрасті;

- навіть дітей без додаткових вимог інтернет-школи приваблюють особливо в невеликих містах чи сільських шкільних округах, де кількість курсів обмежена, у такому випадку студенти можуть вивчати ті напрямки, які були б неможливими для них за звичайних форм навчання [2].

Популярність онлайн навчання викликана наступним:

- миттєвість зворотного зв'язку всередині платформ, що надаються іншими онлайн студентами або платформою, покращують криву навчання, окрім того це також пропонує сором'язливим або більш стриманим людям

можливість брати участь у дискусіях у класі легше, ніж це відбувається в очних заняттях;

– ще однією перевагою є час: люди можуть упорядкувати свій графік відповідно до своєї зручності, і це дозволяє їм масштабуватися самостійно, це представляє величезну можливість для підприємців швидко покращити свої знання;

– структура витрат на навчання в Інтернеті є ще одним фактором швидкого зростання ринку: онлайн курси є більш доступним варіантом порівняно з традиційними, часто як щодо вартості навчання загалом, так і щодо інших витрат, адже витрат на поїздки додому немає, а іноді необхідні навчальні матеріали, такі як підручники, доступні в Інтернеті безкоштовно;

– онлайн навчання є інтерактивним та захоплюючим: навчання в Інтернеті не означає прослуховування лекцій та читання матеріалів разом із лекційними слайдами, навчальні матеріали можуть містити щотижневі заходи, вікторини, дошки обговорень, відео чи аудіо презентації, якщо виникають які-небудь запитання, консультанти з онлайн навчання є єдиним пунктом для академічної підтримки [1], [3].

1.2 Огляд програмних аналогів

Основною програмною платформою, яка використовується для вивчення онлайн-курсів, звичайно є Coursera [4].

Загалом Coursera надає засоби роботи не тільки з онлайн-курсами, її навчальні програми розбиті на 6 основних типів:

– курси: Coursera пропонує понад 3900 онлайн-курсів із сотнями безкоштовних курсів, які викладають найкращі викладачі університетів та компаній світового класу;

– керовані проекти: дозволяють отримати відповідні професійні навички та галузеві інструменти менш ніж за 2 години, інтерактивне

навчання з покроковими візуальними інструкціями та цінами від 9,99 доларів;

- спеціалізації: серія курсів, що підходить для тих, хто хоче оволодіти професійною майстерністю, можна вирішити пройти лише один курс, пов'язаний з кар'єрною майстерністю, або серію курсів, які охоплюватимуть усі аспекти цієї майстерності;

- професійний сертифікат: навчальні програми, які готують до роботи або певної галузі, деякі професійні сертифікати визнаються організаціями або школами;

- сертифікат MasterTrack: семестрові курси, які доступні лише як цілісна одиниця, можна проходити індивідуальні курси в рамках MasterTrack, а сертифікати видаються лише після завершення всього MasterTrack;

- онлайн-ступінь: Coursera співпрацює з провідними університетами, щоб надати учням доступні та акредитовані університетські ступені [5].

Важливими особливостями платформи є наступні:

- сертифікат про закінчення навчання: отримати сертифікат можна сплативши за курс, окрім того звичайно потрібно подавати домашні завдання та скласти тести/іспити;

- шляхи кар'єрного навчання: кожен навчальний шлях для кар'єри Coursera показує просування робочих ролей – від початкового рівня до досвідченого професіонала – у певній галузі кар'єри (наприклад, науковець даних, менеджер проектів, мережний інженер), потім навички, необхідні для навчання для прогресу в цій кар'єрі, відображаються у відповідних навчальних програмах на Coursera;

- фінансова допомога: якщо користувач не може дозволити собі оплатити курс, він може подати заявку на отримання фінансової допомоги;

- форуми курсів: можна здійснити пошук, щоб дізнатись, чи вже було задано питання, яка виникає цікавість користувача, або задати власне запитання, щоб отримати необхідну підтримку;

- контроль лекції: можна змінювати швидкість відтворення лекцій;

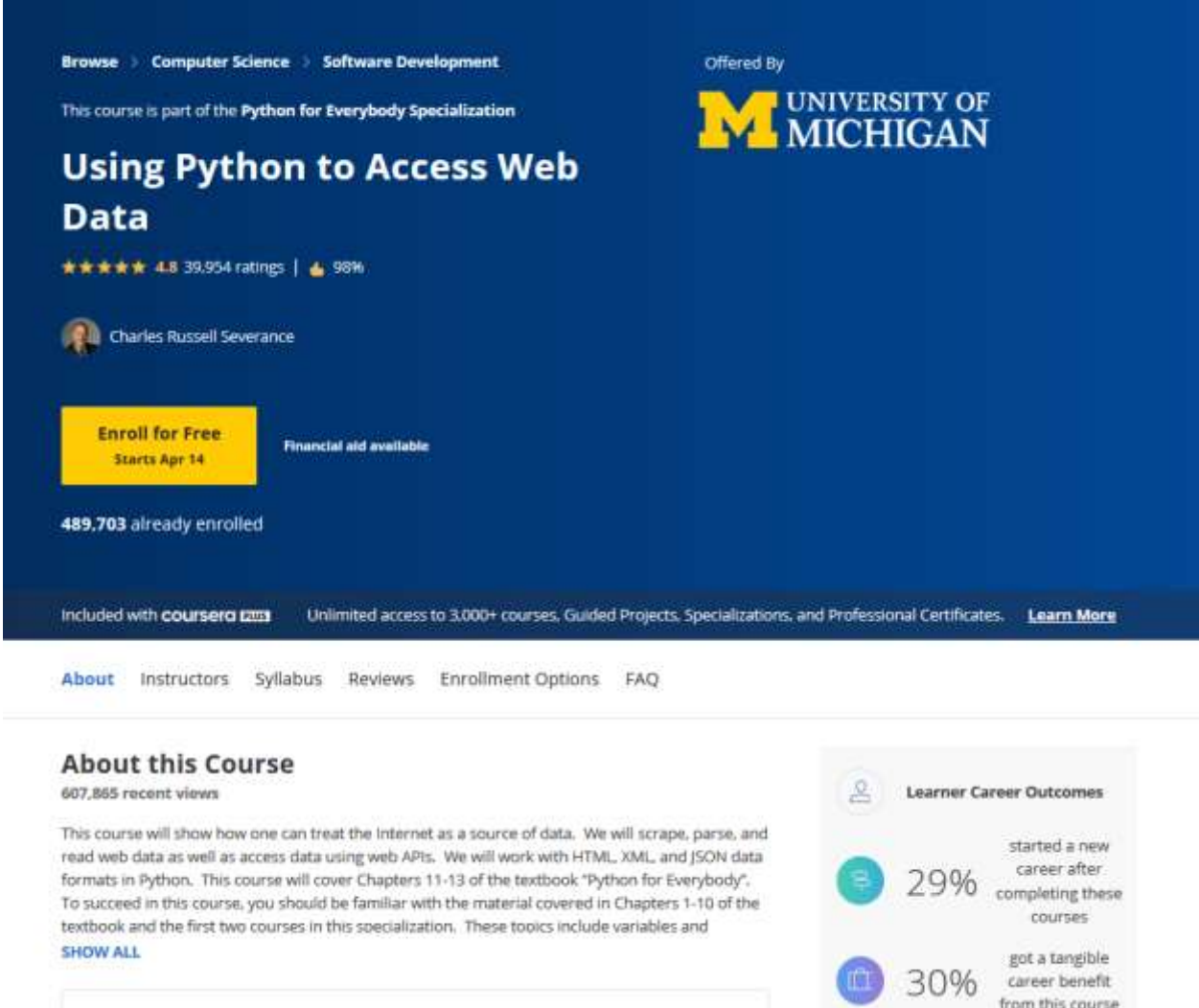
- підтримка між платформами: курси доступні на будь-якому настільному та мобільному пристрої, у режимі офлайн у тому числі;
- завантажувані відеофайли: можна дивитися відео з курсу в автономному режимі;
- вікторини, домашні завдання та іспити: Coursera пропонує як автоматично оцінені, так і рецензовані завдання, щоб перевірити свої знання в курсі;
- відео субтитри: усі відео Coursera мають субтитри основною мовою курсу, деякі курси мають перекладені субтитри;
- персоналізовані рекомендації: побудований на моделях машинного навчання, механізм рекомендацій Coursera дає персоналізовані рекомендації щодо курсів, які слід вивчати далі;
- направлення до штучного інтелекту, щоб допомогти не відставати: інструменти, які допомагають створювати навчальні цілі та відстежувати прогрес, а також цілеспрямовані рекомендації щодо огляду матеріалів, коли користувач не проходить тест [6].

Щодо організації безпосередньо роботи над курсами в Coursera (рис. 1.1), можна виділити, що кожна частина курсу складається з елементів від 30 хвилин до кількох годин відео, а також тесту для перевірки розуміння, рецензованих есе, а також подальшого читання матеріалів та посилань на інші інструменти в Інтернеті, які можна спробувати, у кінці кожного модуля є різні завдання, залежно від курсу [7].

Більшість фактичного вмісту на Coursera безкоштовна, так якщо обрати опцію «перевірити» курс, то можна побачити весь матеріал, але не доведеться нічого платити, однак безкоштовні курси завжди мають певні функції, за які доведеться заплатити, якщо знадобиться отримати до них доступ, до них належать:

- подача завдання для зворотного зв'язку;
- отримання наставництва;

- деякі курси мають додаткові відео та списки читання для платних учасників;
- необхідність отримання сертифікату [7].



Using Python to Access Web Data

Offered By **UNIVERSITY OF MICHIGAN**

4.8 39,954 ratings | 98%

Charles Russell Severance

Enroll for Free
Starts Apr 14

Financial aid available

489,703 already enrolled

Included with **Coursera Plus** | Unlimited access to 3,000+ courses, Guided Projects, Specializations, and Professional Certificates. [Learn More](#)

[About](#) [Instructors](#) [Syllabus](#) [Reviews](#) [Enrollment Options](#) [FAQ](#)

About this Course

607,865 recent views

This course will show how one can treat the Internet as a source of data. We will scrape, parse, and read web data as well as access data using web APIs. We will work with HTML, XML, and JSON data formats in Python. This course will cover Chapters 11-13 of the textbook "Python for Everybody". To succeed in this course, you should be familiar with the material covered in Chapters 1-10 of the textbook and the first two courses in this specialization. These topics include variables and

[SHOW ALL](#)

Learner Career Outcomes

- 29% started a new career after completing these courses
- 30% got a tangible career benefit from this course

Рисунок 1.1 – Coursera

В основному, якщо це просто навчання з цікавості, то можна записатися на більшість курсів на Coursera безкоштовно, якщо потрібно отримати належну кваліфікацію (сертифікат), доведеться заплатити, до того ж обмежена кількість курсів не доступна безкоштовно, в основному це довші професійні курси та дипломні програми:

- індивідуальні курси, за які не буде отримано сертифікат, стартують від 49 доларів США (фіксована плата);

– спеціалізації та професійні сертифікати коштують від 39 до 89 доларів на місяць;

– сертифікати MasterTrack коштують більше 2000 доларів;

– онлайн ступінь починається приблизно з 10 000 доларів [5], [7].

До недоліків платформи можна віднести:

– значні бар'єри для вступу (час, гроші та підтримка), які означають, що молодші викладачі та малі заклади недостатньо представлені в каталозі;

– якщо учень (студент) потребує отримання сертифікату за результатами навчання, то це потребує сплати за курси;

– курси на вимогу є поки що незавершеною роботою [8].

Наступним інструментом з даного пулу є Udeyу [9] (рис. 1.2).

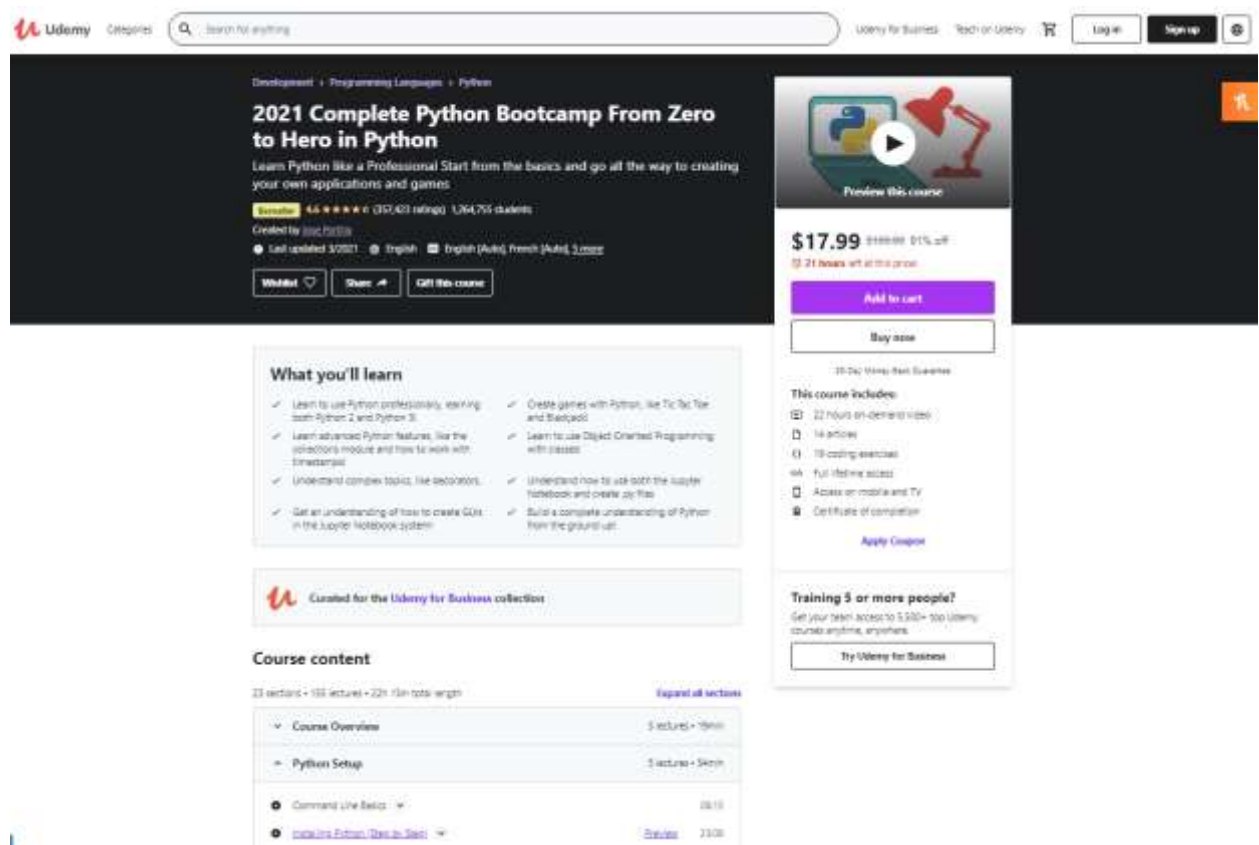


Рисунок 1.2 – Udeyу

Udeyу – це сайт онлайн навчання, який продає відеокурси на широкий спектр тем, якісно представленими напрямками є особистий та професійний розвиток, тренінги з управління, підручники з ПЗ, курси

програмування тощо, при цьому можна платити за курси Udemu по одному або отримувати доступ до понад 4000 в будь-який час за умови підписки на бізнес-варіант, при цьому варто зазначити, що ціна дещо висока, і потрібно мінімум п'ять людей для ведення бізнес-підписки, що складається з двох рівнів: Team і Enterprise, підписка на команду коштує 360 доларів на людину на рік і підходить для команд від 5 до 20 осіб, у корпоративних облікових записах використовується спеціальне ціноутворення залежно від розміру та потреб організації [10].

Щодо безпосередньо організації курсів у програмі варто відзначити наступне:

- основою навчання є відео: вони можуть бути виконані у стилі лекцій або навчальних посібників, відео, як правило, триває не більше 10 хвилин кожне, і вони згруповані в розділи, курс може тривати три години, але кожне відео не триватиме більше 10 хвилин, макет допомагає викладачам переконатися, що їх курси чітко структуровані, маючи розділи та короткі відео, викладачі повинні розбити їх вміст на конкретні, засвоювані фрагменти, налаштування допомагають студентам побачити загальний обсяг курсу заздалегідь, це також полегшує студентам паузу та перерву в курсі, коли це потрібно, оскільки перерва ніколи не триває більше ніж дев'ять чи десять хвилин, наявність коротких відео також дозволяє учням легко повторити те, що вони хочуть оновити;

- коли урок включає додаткові ресурси, такі як робочі аркуші, можна завантажити їх, зазвичай викладачі посилаються на ці матеріали та обговорюють їх на курсі;

- під відеопрогравачем є декілька вкладок, на яких студенти можуть взаємодіяти: існує «Огляд», який дуже схожий на детальний опис курсу, який демонструється перед реєстрацією, «Запитання та відповіді» – це місце, де учні можуть задавати питання, люди можуть голосувати за запитання, якщо хочуть надати їм більшої популярності, «Нотатки» – це місце, де можна робити нотатки, які ви можете прив'язати до певної точки в будь-якому з

відео, «Анонси» – дозволяє викладачеві ділитися новою інформацією, наприклад, новим відповідним курсом або інтерактивною подією для студентів;

– обсяг взаємодії спільноти залежить від курсу: деякі викладачі дають доручення розміщувати запитання та відповіді, хоча це завжди необов'язково, деякі викладачі отримують декілька запитань на день, сайт підтримує можливість завантаження зображень із запитаннями, тому студенти можуть візуально ділитися своєю роботою, коли їх запитують [10].

EdX [11] (рис. 1.3) – платформа, заснована провідними університетами США Гарвардом та Массачусетським технологічним університетом, на відміну від інших постачальників курсів EdX є некомерційною організацією, на платформі представлено багато безкоштовних курсів з широкого кола тем, при цьому можна виділити наступні особливості:

– навігація курсами є дещо складнішою, ніж у прямого конкурента Coursera, особливо на дискусійних форумах;

– відеолекції більшості курсів доступні через YouTube і доступні для завантаження для навчання в автономному режимі;

– багато курсів доступні безкоштовно без сертифіката;

– широкий набір програм навчання: MicroMasters, професійний сертифікат, XSeries, онлайн ступінь магістра та Global Freshman Academy, однак більшість відмінностей пов'язані з маркетингом, MicroMasters та ступінь магістра також дозволяють студентам отримувати академічні кредити в університетах США;

– багато курсів EdX з часом будуть заархівовані – це означає, що курс все ще доступний, але він більше не контролюється відповідним університетом (тобто відсутні іспити чи сертифікати);

– деякі курси включають оцінку іспиту вручну командою курсу [12].

The screenshot shows the EdX website interface. At the top, there's a navigation bar with 'edX' logo, links for 'Courses', 'Programs & Degrees', and 'Schools & Partners', a search bar, and links for 'edX for Business', 'Sign in', and 'Register'. Below this, a breadcrumb trail reads 'Catalog > Computer Science Courses > IBM's Cloud Application Development Foundations'. The main header features the IBM logo and the course title 'Introduction to Cloud Development with HTML5, CSS3, and JavaScript'. A description states: 'Learn the key tools and programming languages to bootstrap your career as a Cloud Developer. Get started with web development using HTML5, CSS3, JavaScript and GitHub.' To the right is an illustration of a hand interacting with a cloud and gears. A dark green banner below the header lists course details: 'Estimated 4 weeks' (1-2 hours per week), 'Self-paced' (Progress at your own speed), and 'Free to start' (Add a verified certificate for \$49 USD). The section 'There is one session available:' indicates the course is archived. A box shows the session dates: 'Starts Apr 15' and 'Ends Jun 30'. A checkbox for email notifications is present. A red 'Enroll now' button is shown with '15,017 already enrolled'. At the bottom, a navigation bar includes links for 'About', 'What you'll learn', 'Syllabus', 'Instructors', and 'Ways to enroll'.

edX Courses Programs & Degrees Schools & Partners Search online courses Sign in Register

Catalog > Computer Science Courses > IBM's Cloud Application Development Foundations

IBM

Introduction to Cloud Development with HTML5, CSS3, and JavaScript

Learn the key tools and programming languages to bootstrap your career as a Cloud Developer. Get started with web development using HTML5, CSS3, JavaScript and GitHub.

Estimated 4 weeks
1-2 hours per week

Self-paced
Progress at your own speed

Free to start
Add a verified certificate for \$49 USD

There is one session available:
After a course session ends, it will be [archived](#).

Starts Apr 15
Ends Jun 30

☐ I would like to receive email from IBM and learn about other offerings related to Introduction to Cloud Development with HTML5, CSS3, and JavaScript.

Enroll now 15,017 already enrolled

About What you'll learn Syllabus Instructors Ways to enroll

About this course

Want to take the first steps to become a Cloud Application Developer? This self-paced online course will lead you through the programming languages and tools you will need to develop your own Cloud and Web Applications.

[Show more](#)

At a glance

- Institution:** IBM
- Subject:** [Computer Science](#)
- Level:** Introductory
- Prerequisites:**
 - Basic computer literacy.
 - Understanding of Cloud Computing concepts and
- Language:** English
- Associated programs:**
 - Professional Certificate in [Cloud Application Development Foundations](#)
 - Professional Certificate in [Full Stack Cloud Developer](#)

Рисунок 1.3 – EdX

1.3 Висновки за розділом 1

У першому розділі проаналізовано особливості та причини популярності онлайн курсів, виконано огляд програмних аналогів,

включаючи інструмент більш традиційної роботи з онлайн курсами Coursera, інструмент продажу відеоматеріалів за курсами Udemy та платформи переважно безкоштовних курсів EdX.

Завдання дипломної кваліфікаційної роботи бакалавра:

- виділення вимог до ПЗ підтримки вивчення персоналізованих курсів;
- вибір мови і засобів проєктування і розробки;
- проєктування ПЗ;
- реалізація і налагодження ПЗ.

2 РОЗРОБКА АРХІТЕКТУРИ ПРОГРАМИ

2.1 Вибір мови програмування

Логічним рішенням розробки програми є створення вебзастосунку, адже в такому випадку до програми можна буде звертатися і з персонального стаціонарного комп'ютера, і з ноутбука, і зі смартфонів, і з планшетів. За рахунок використання серверу можна буде організувати доступ одразу зі всіх пристроїв, що дозволить в межах дипломної роботи створити рішення, яке одразу можна практично впровадити і використовувати незалежно від того, які технічні можливості є у кінцевого користувача.

Одним з найпопулярніших рішень проблеми розробки серверної сторони на даний момент є мова програмування Java [13]. Тому її було визначено першим можливим варіантом розробки програми.

Мова програмування Python дозволяє створювати вебзастосунку, виконувати машинне навчання, забезпечувати автоматизацію [14]. Враховуючи також наявність вебфреймворків для неї (Django, Flask), її було обрано другим можливим варіантом.

У табл. 2.1 представлено результати виконаного порівняння.

Таблиця 2.1 – Порівняння мов програмування для реалізації програми

Показник	Python	Java
Синтаксис	Простий	Складніший
Керування пам'яттю	Автоматичне	Автоматичне
Типізація	Динамічна	Статична
Швидкість виконання	Середня (за рахунок використання інтерпретатора)	Менша
Вивчення нових можливостей	Легке	Достатньо складне (тривале)
Вебфреймворки	Spring	Django, Flask

У результаті вибір було зроблено на користь мови програмування Python, адже вона дозволяє створювати застосунки, що працюють швидше, окрім того спрощуючи і пришвидшуючи сам процес розробки.

2.2 Вимоги до програмного забезпечення

Для досягнення цілі і завдань роботи програма повинна виконувати наступні функції:

- реєстрація облікових записів користувача;
- авторизація;
- пошук курсів;
- створення курсів;
- редагування даних курсів;
- внесення і редагування відеоматеріалів за курсами;
- створення тестів за курсом;
- проходження тестів за курсом;
- обчислення результату тесту;
- внесення відгуків після проходження курсу;
- визначення результату проходження курсу;
- формулювання запитань до викладача під час проходження курсу;
- створення текстових матеріалів за курсом;
- перегляд відеоматеріалів за курсом;
- читання текстового матеріалу за курсом;
- перегляд персоналізованих відеоматеріалів за курсом;
- читання персоналізованих тестових матеріалів за курсом;
- оформлення підписки на проходження курсу;
- перегляд кількості підписок на курси;
- перегляд власних результатів проходження курсу;
- перегляд переліку пройдених курсів;
- перегляд результатів навчання студентів за курсом;

- перегляд переліку курсів, навчання за якими відбувається;
- перегляд переліку створених курсів;
- перегляд запитань за курсом;
- перегляд відповідей на власні запитання;
- перегляд відгуків за курсом;
- перегляд доступних матеріалів за курсом;
- створення анкети за навчанням з курсу;
- заповнення анкети за навчанням з курсу.

У програмі має бути забезпечено чіткий розподіл між авторами курсів та учнями (студентами). Дана функція перш за все має виконуватися для того, щоб будь-який користувач не міг стати автором курсу, адже це може призвести до зниження якості матеріалів на вебсайті, а в результаті – до зниження інтересу, відповідно втрати користувачів. У той же час студентам не має створюватися ніяких додаткових бар'єрів: кожен бажаючий повинен мати можливість пройти курс, що є однією з важливих особливостей вебзастосунку порівняно з існуючими програмами, адже для привертання уваги до рішення поріг входження має бути знижений.

Головною особливістю розроблюваного ПЗ є надання персоналізованих курсів, що виражається у можливості отримання спеціальних відеоматеріалів та текстових матеріалів в залежності від результатів проходження тестів за курсом та заповнення анкети щодо навчання за курсом, де користувач може акцентувати увагу на тих питаннях, які у нього викликають найбільшу складність або найбільшу цікавість.

Взаємодія учня з програмою повинна виконуватися на основі реалізації наступних сценаріїв (рис. 2.1).

По-перше, авторизація, а в якості альтернативного сценарія – реєстрація облікових записів користувача.

По-друге, робота над новими курсами:

- пошук курсів;
- перегляд описів курсів;

- оформлення підписки на проходження курсу;
- перегляд відгуків за курсом.

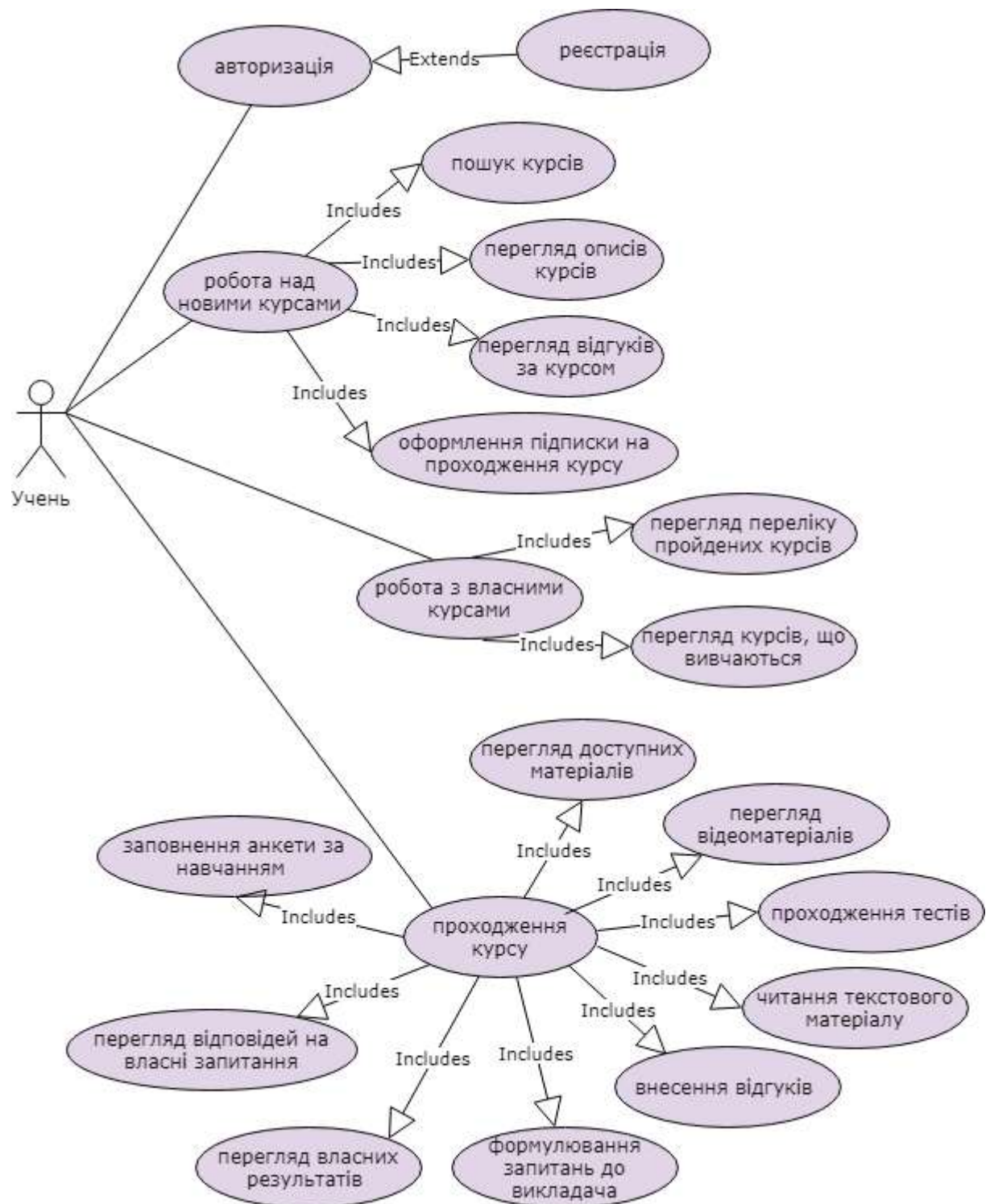


Рисунок 2.1 – Діаграма варіантів використання системи учнем

По-третє, робота з власними курсами:

- перегляд переліку пройдених курсів;

– перегляд переліку курсів, навчання за якими відбувається.

Сценарій проходження курсу охоплює:

- перегляд доступних матеріалів за курсом;
- перегляд відеоматеріалів за курсом, включаючи перегляд персоналізованих відеоматеріалів;
- читання текстового матеріалу за курсом, включаючи читання персоналізованих тестових матеріалів;
- проходження тестів за курсом, включаючи автоматизоване обчислення результату тесту;
- внесення відгуків після проходження курсу;
- формулювання запитань до викладача під час проходження курсу;
- перегляд власних результатів проходження курсу;
- перегляд відповідей на власні запитання;
- заповнення анкети за навчанням з курсу.

На рис. 2.2 представлена відповідна діаграма для викладача.

Взаємодія викладача з програмою повинна виконуватися на основі реалізації наступних сценаріїв:

- авторизація;
- створення курсів;
- перегляд переліку створених курсів, включаючи перегляд кількості підписок на курси;
- редагування даних курсів;
- внесення і редагування відеоматеріалів за курсами;
- створення і редагування тестів за курсом;
- створення і редагування текстових матеріалів за курсом;
- перегляд результатів навчання студентів за курсом;
- перегляд запитань за курсом;
- створення відповідей на запитання;
- перегляд відгуків за курсом;
- створення анкети за навчанням з курсу.

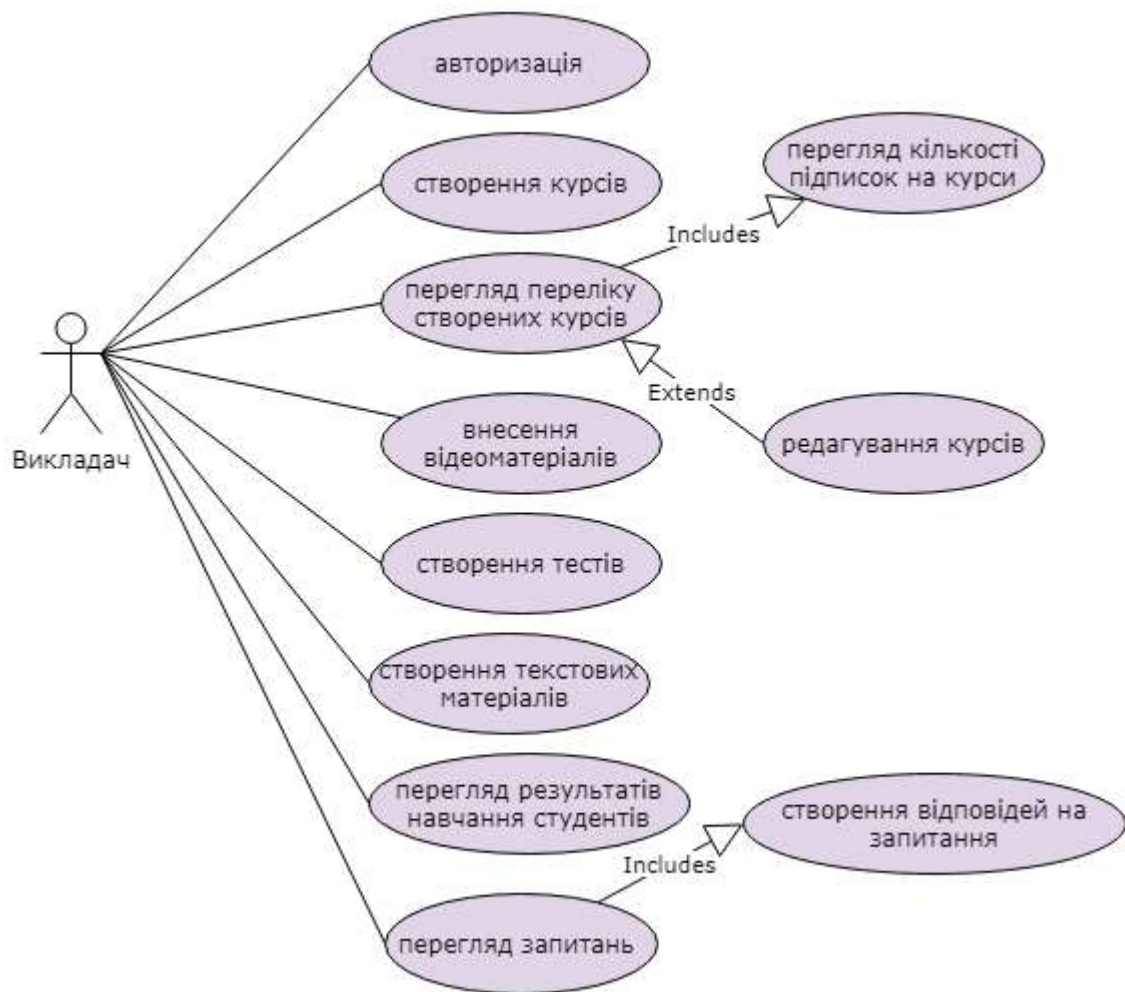


Рисунок 2.2 – Діаграма варіантів використання системи викладачем

2.3 Проєктування структури даних

Організація роботи з даними виконана на основі створення бази даних (БД). Для керування БД використано систему MySQL [15], оскільки вона, на відміну від SQLite, надає можливість створення рішень рівня виробництва, відповідно проблем з подальшим розгортанням створеного програмного продукту бути не повинно.

Поля таблиці Account, що представляє обліковий запис користувача:

- id – ідентифікатор;
- email – електронна пошта користувача;
- password – пароль користувача;
- surname – прізвище користувача;

- name – ім'я користувача;
- phone – телефон користувача;
- birth – дата народження користувача;
- profile – картинка;
- info – опис користувача (може вказуватися місце роботи, додаткові регалії за необхідності);
- asstype – тип облікового запису (учень або учитель).

Поля таблиці Course, що представляє головну сутність предметної області – курс:

- id – ідентифікатор;
- title – назва;
- info – опис курсу;
- startlearn – дата початку найближчого вивчення курсу;
- endlearn – дата завершення найближчого вивчення курсу;
- account – викладач (автор курсу);
- created – дата і час створення курсу;
- subscriptions – кількість підписів на курс;
- score – середня оцінка курсу;
- image – шлях до файлу з зображенням для курсу.

Поля таблиці Material, що представляє матеріали за курсом (об'єднані відеоматеріали та текстові матеріали):

- id – ідентифікатор;
- title – назва матеріалу;
- course – курс, до якого належить матеріал;
- order – порядок, в якому має з'явитися матеріал в курсі (матеріали не відображаються всі одночасно, а з'являються порціями) серед інших матеріалів, тестів та анкет;
- type – тип матеріалу (відеоматеріал або текстовий матеріал);
- file – шлях до файлу матеріалу;
- info – опис матеріалу;

– group – група матеріалів за курсом, до якої належить даний (основні або допоміжні);

– startlearn – дата початку доступності матеріалу;

– endlearn – дата завершення доступу до матеріалу.

Поля таблиці Subscription, що визначає підписку учня на вивчення курсу:

– id – ідентифікатор;

– account – обліковий запис учня;

– course – курс, підписка на який оформлена;

– startlearn – дата, коли користувач підписався на вивчення курсу;

– endlearn – дата, коли користувач завершив вивчення курсу;

– result – результат (отримана оцінка);

– feedback – зворотній зв'язок від користувача (відгук за курсом);

– feedpoints – оцінка відгука;

– feeddate – дата і час оформлення відгуку;

– lastorder – остання позиція матеріалів, яку переглянув користувач.

Поля таблиці Communication, що визначає комунікацію між учнем та викладачем:

– id – ідентифікатор;

– subscription – підписка на курс;

– question – запитання;

– qudate – дата і час запитання;

– answer – відповідь;

– ansdate – дата і час відповіді.

Поля таблиці TestQuestion, що представляє запитання тесту:

– id – ідентифікатор;

– qtext – текст запитання;

– test – тест, до якого належить запитання;

– order – порядок запитання у тесті.

Поля таблиці TestQuestionAnswer, що представляє варіанти відповідей на запитання тесту:

- id – ідентифікатор;
- atext – текст відповіді;
- testquestion – запитання, до якого належить відповідь;
- order – порядок відповіді у запитанні;
- right – правильність відповіді.

Поля таблиці Que, що представляє анкету щодо навчання за курсом:

- id – ідентифікатор;
- course – курс, до якого належить анкета;
- order – порядок, в якому має з'явитися анкета в курсі серед інших матеріалів, тестів та анкет;

- start – дата початку доступності анкети;
- end – кінцева дата доступності анкети.

Поля таблиці QueQuestions, що представляє запитання анкети:

- id – ідентифікатор;
- que – анкета, до якої належить запитання;
- qtext – текст запитання;
- order – порядок запитання в анкеті.

Поля таблиці SubscriptionMaterial, що визначає роботу підписника з матеріалами курсу:

- id – ідентифікатор;
- subscription – підписка користувача на курс;
- material – матеріал;
- watched – дата і час, коли було переглянуто матеріал востаннє;
- order – порядок відображення матеріала для користувача в курсі;
- show – визначає, потрібно відображати матеріал користувачу або ні.

Поля таблиці QueQuestionAnswers, що представляє варіанти відповідей на запитання анкети:

- id – ідентифікатор;

- qtext – текст відповіді;
- quequestion – запитання з анкети, до якого належить поточна відповідь.

Поля таблиці SubscriptionQueQuestionAnswers, що представляє фактичні відповіді користувача на запитання анкети:

- id – ідентифікатор;
- account – користувач;
- answer – обраний варіант відповіді;
- adate – дата і час відповіді.

Поля таблиці AccountTest, що представляє тест, виконаний користувачем:

- id – ідентифікатор;
- tdate – дата і час проходження тесту;
- points – кількість балів, набрану за тест;
- subscription – підписка, за якою відбулось тестування;
- test – тест.

Поля таблиці Test, що визначає тест за курсом:

- id – ідентифікатор;
- title – назва тесту;
- course – курс, до якого належить тест;
- order – порядок, в якому має з'явитися тест в курсі серед інших матеріалів, тестів та анкет;

- starttest – дата, з якої доступний тест;
- endtest – дата, до якої доступний тест;
- typetest – тип тесту (враховується оцінка у загальній оцінці курсу або ні);

- points – кількість балів, яку можна отримати за тест за всі правильні відповіді.

На рис. 2.3 представлена структура БД.

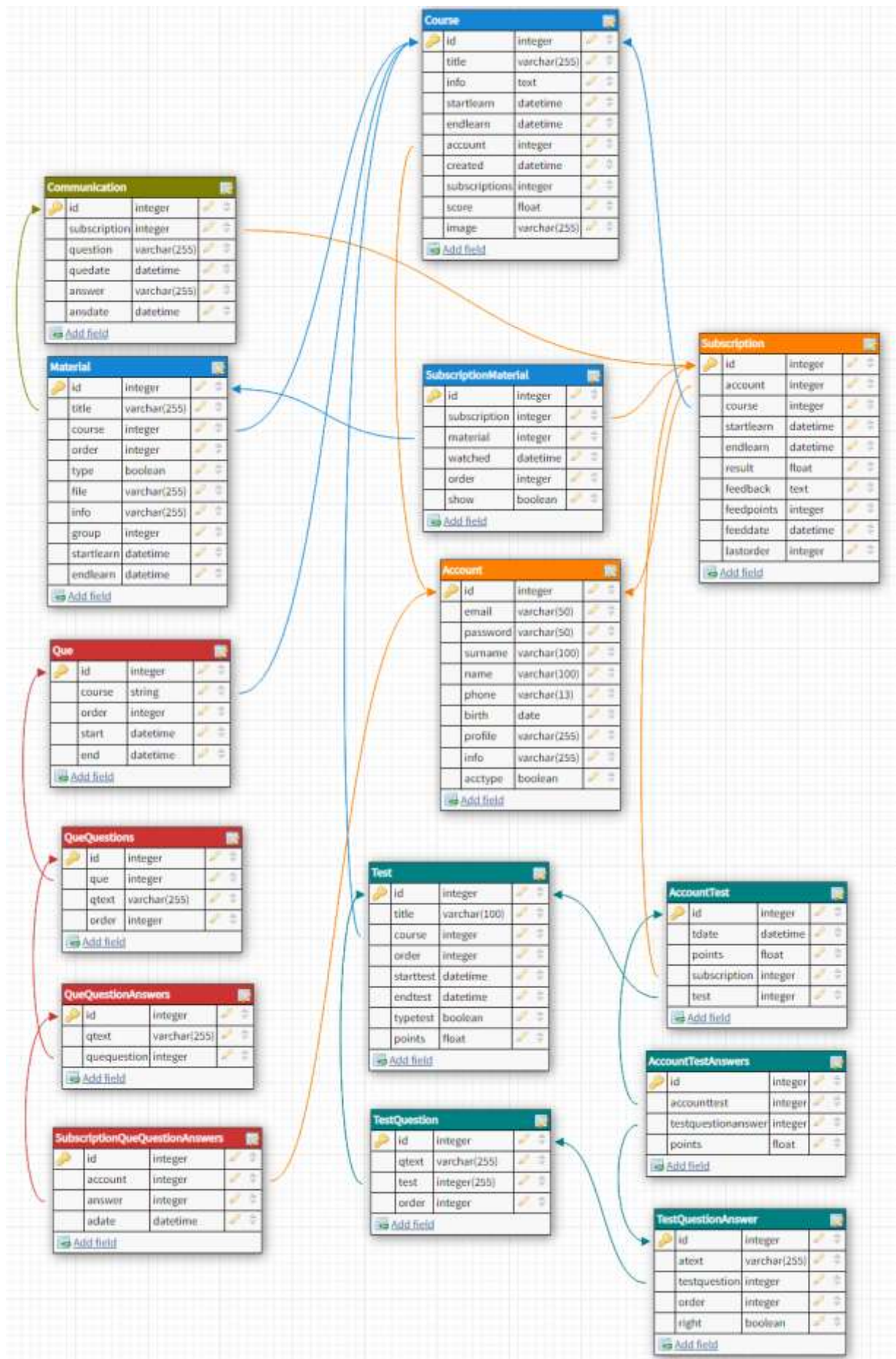


Рисунок 2.3 – Структура БД

Поля таблиці AccountTestAnswers, що представляє відповіді на запитання тесту, отримані від користувача:

- id – ідентифікатор;
- accounttest – тест користувача;
- testquestionanswer – обрана відповідь на запитання;
- points – кількість балів за відповідь.

2.4 Висновки за розділом 2

За результатами порівняння мов програмування обрано мову Python для реалізації програми.

Визначено функціональні вимоги до програми з моделюванням основних сценаріїв роботи учня та викладача з програмою у вигляді діаграм варіантів використання. Визначено спосіб роботи з даними на основі БД з представленням структури цієї БД.

3 РОЗРОБКА ПРОГРАМИ

3.1 Послідовність персоналізації курсів

Послідовність, яка використовується для реалізації процесу навчання за взаємодією з користувачем (учнем), представлена на діаграмі діяльності на рис. 3.1.

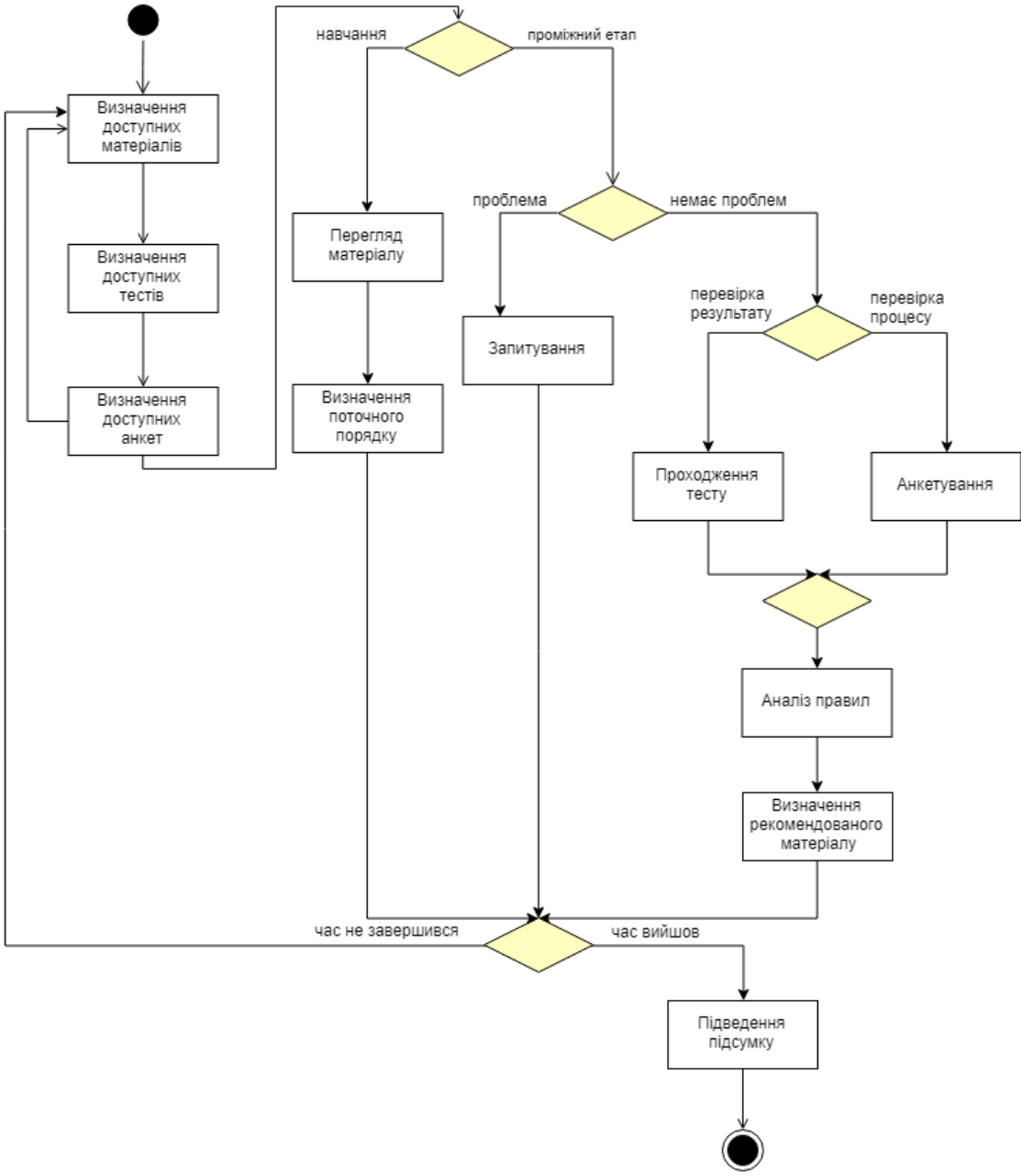


Рисунок 3.1 – Діаграма діяльності навчання учня

Для реалізації персоналізації курсів було використано ідею персоналізації матеріалів в залежності від того, як учень під час вивчення курсу проявив себе під час складання тестів та позицій, які він висловив в анкеті. Під час вивчення курсу може реалізовуватися декілька тестів, тобто окрім підсумкового тесту можуть передбачатися і інші. Під час побудови БД було використано ідею, що для теста зазначається, чи входить він в оцінку. Таким чином можна створювати тести, які дозволяють зрозуміти поточний рівень знань учня, які можуть і не впливати на підсумкову оцінку, але дозволяють встановити проблемні для учня питання і навпаки – питання, які проблем не викликають.

Усі матеріали в свою чергу поділені на підгрупи: основні і додаткові. Персоналізація відбувається саме на основі додаткових матеріалів. За результатами необхідно визначити, який саме додатковий матеріал потрібно використати для навчання учня.

Також на персоналізацію впливають відповіді учня на запитання анкети. За допомогою цього учень може виявити, що саме для нього є першочерговим у проблемах з навчанням.

В залежності від відповідей на окремі запитання, можна визначати, який саме додатковий матеріал показати або навпаки не показувати, а також визначати, в якому порядку показати цей матеріал. У БД для кожного матеріала вказано стандартний порядок відображення. Тоді можна використати одну з двох схем:

- визначити, що в залежності від відповідей на запитання, потрібно пропустити або відтворити певний матеріал;
- визначити, що в залежності від відповідей на запитання, потрібно відтворити певний матеріал в іншому порядку, тобто замінити між собою певні матеріали.

Для визначення системи персоналізації використовуються правила. Правила накопичуються у текстовому файлі.

Кожне правило може мати наступну форму: $t_1 (2\ 4\ 7) = 1$, $t_1 (9\ 11\ 12) = 0 \rightarrow m_{10+5}$, m_{12+6} або $a_3 (21\ 2, 22\ 6, 23\ 8) \rightarrow m_1$.

У випадку першого правила: якщо при проходженні тесту (вказує літера «t» на це) з ідентифікатором 1 за відповіді на запитання під номерами 2, 4, 7 було отримано в сумі максимум 1 бал, а за відповіді на запитання 9, 11, 12 було отримано 0 балів, то матеріал з ідентифікатором 10 відобразити (на це вказує символ «+») п'ятим, а матеріал з ідентифікатором 12 відобразити шостим.

У випадку другого правила: якщо при заповненні анкети (вказує літера «a» на це) з ідентифікатором 3 при відповіді на запитання 21 було обрано відповідь 2, при відповіді на запитання 22 було обрано відповідь 6, а при відповіді на запитання 23 – відповідь 8, то матеріал з ідентифікатором 1 не відображати (на це вказує мінус в кінці).

Будь-яке правило розділено на ліву та праву частини, відділені символом «->».

У лівій частині може бути перелічена довільна кількість комбінацій, що включають відповіді на тести або анкети. Кожна така група має включати літеру «t» або «a» для визначення тесту або анкети. Після цієї літери вказується їх ідентифікатор у відповідній таблиці. Далі для кожної такої групи в дужках має бути перелічено:

– для теста: через пробіл ідентифікатори запитань теста, а після закриття групи через знак «=» кількість балів, яку сумарно за ці запитання має набрати користувач, включаючи також випадки, коли набрано було менше (якщо потрібно навпаки визначити, що за цю групу запитань було отримано вказану кількість балів або більшу, то потрібно після кількості балів вказати символ «+» (наприклад, «= 15+»);

– для анкети: через кому пари, що включають через пробіл ідентифікатор запитання та варіант відповіді.

У правій частині має бути вказано через кому матеріали, на відображення яких впливає це правило. Для визначення матеріалу

використовується літера «m», після якої має бути вказано ідентифікатор матеріала. Після ідентифікатора вказується знак «+» або «-», що позначає відобразити або навпаки приховати матеріал відповідно. За замовчуванням кожен матеріал з БД відображається. Після цього знаку може також бути вказано порядковий номер. Цей порядковий номер для вказаних випадків, визначених правилом, змінює порядок відображення матеріалу з того, який зазначено в таблиці БД в полі order, на той, який зазначено в правилі.

3.2 Розроблені класи

Для реалізації вебзастосунку було використано вебфреймворк Django [16], який дозволяє спростити процес розробки за рахунок забезпечення взаємодії між компонентами програмної системи.

Забезпечення роботи з даними було виконано на основі створення відповідного набору моделей програмного застосунку, що відповідають таблицям БД. При цьому за рахунок об'єктно-реляційного відображення ці моделі були автоматично перетворені на таблиці БД, що є одним з факторів, який позитивно вплинув на швидкість розробки проєкту.

Процес створення моделей вебзастосунку виглядає наступним чином. Кожна модель створена наслідуванням класу `models.Model` пакету `django.db` це дозволяє уникати необхідності визначення запитів до БД зокрема.

Для кожної моделі визначено набір полів. Наприклад, для моделі курсу `Course` її поля наступні:

- `title = models.CharField(max_length=255)` – назва курсу може складатися максимально з 255 символів і визначена як символічне поле;
- `info = models.TextField()` – опис курсу визначений як текстове поле;
- `startlearn = models.DateTimeField()` – дата початку вивчення курсу визначена як поле дати і часу;
- `endlearn = models.DateTimeField()` – дата завершення вивчення курсу визначена як поле дати і часу;

– `created = models.DateTimeField()` – дата створення курсу визначена як поле дати і часу;

– `subscriptions = models.IntegerField()` – кількість підписок на курс визначена цілочисельним полем, значення якого змінюється безпосередньо через представлення вебзастосунку при натисканні на підписку курсу користувачем, тоді окрім створення об'єкту підписки на основі відповідної моделі змінюється також дане поле шляхом збільшення на 1;

– `account = ForeignKey (Account, on_delete=models.CASCADE)` – автор курсу визначається як зовнішній ключ, значення якого відповідає обліковому запису користувача з таблиці `Account`;

– `score = models.FloatField()` – дійсне поле, яке визначає середню оцінку за курс;

– `image = models.CharField(max_length=255)` – шлях до файлу з зображенням курсу може складатися максимально з 255 символів і визначений як символне поле.

Окрім того для даного класу, як і для деяких інших, біло визначено метод `__str__(self)`, який дозволяє опрацьовувати даний об'єкт як рядок, повертаючи при цьому назву курсу.

Дане опрацювання для моделі `Material`, яка відповідає як відеоматеріалам, так і текстовим матеріалам, включає повернення окрім назви ще відповідного типу матеріалу:

```
def __str__(self):
    m = ""
    if self.type == 0:
        m = 'Відеолекція: '
    else m = 'Текстова лекція: '
    return m + self.title
```

Для класу `Course` було визначено також підклас `Meta`, в якому визначено спосіб упорядкування курсів між собою – на основі кількості підписників на курс:

```
class Meta:
    ordering = ['-subscriptions']
```

Клас `Course` має також метод `getfirstparagraph (self)`, який розділяє опис дисципліни за входженням тегу переходу на новий абзац (після введення користувачем опису всі позиції переносу каретки замінюються у відповідному представленні створення курсу на теги `<br>`) і повертає перший абзац:

```
self.info.split('<br>')[0]
```

Взаємодія користувача з Django розпочинається з оброблення його запиту, для цього використовується URL-менеджер. Він побудований за наступним принципом. Це перелік відповідностей між адресою та представленням, яке його має обробляти. Кожен елемент організований подібним чином:

```
path('courses', CoursesView.as_view(), name='coursesview')
```

Відповідно, якщо після назви вебсервера буде вказано розділ `«/courses»`, то відкриється сторінка, сформована на основі шаблону, що відповідає представленню `CoursesView`, яке відповідає за завантаження потрібних для заповнення шаблону даних.

Представлення `CoursesView` є основним представленням для роботи учня з пошуку курсів (рис. 3.2). Дане представлення побудовано наслідування класів `ListView`, що визначає засоби роботи з переліком

об'єктів, та `LoginRequiredMixin`, що визначає засоби авторизації користувачів.

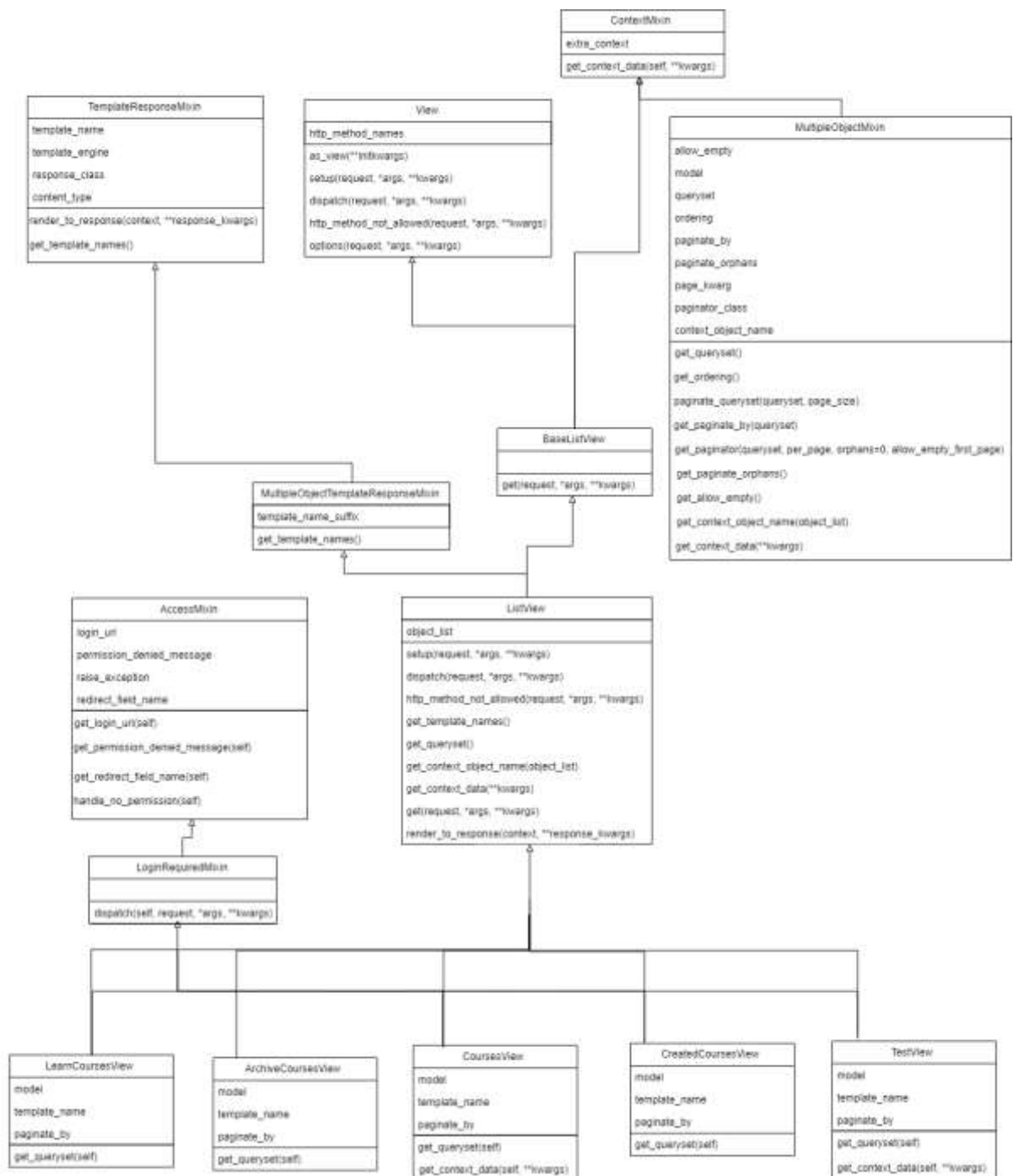


Рисунок 3.2 – Діаграма класів представлень переліків курсів і тестів

Представлення `CoursesView` відповідальне за представлення переліку курсів. Цей перелік може бути відсортований або за кількістю підписників на курс, або за середньою оцінкою курсу. Окрім того на отриманий результат

впливає розташоване на даній сторінці поле пошуку. В залежності від того, що було введено в дане поле, має змінюватися і перелік курсів, що відображаються.

На перелік впливає два параметри, значення яких передаються з форми пошуку даної сторінки через параметри GET-запиту:

```
searchparam = self.request.GET.get('searchby')  
ordertype = self.request.GET.get('ordertype')
```

На основі витягнутих значень параметрів відбувається формування переліку курсів, назви яких мають включати значення, вказане в полі пошуку, а порядок сортування має відповідати обраному:

```
createdcontext = Course.objects.filter(  
    title__contains = searchparam  
)order_by(ordertype)
```

Порядок сортування визначається шляхом встановлення поля моделі, яке відповідає конкретному варіанту, заданому в GET-запиті:

```
if ordertype == 'pop':  
    ordertype = 'subscriptions'  
elif ordertype == 'feedb':  
    ordertype = 'score'  
else:  
    ordertype = 'subscriptions'
```

Перший параметр визначає поле пошуку, а другий параметр визначає варіант сортування переліку (за кількістю підписників або за середньої оцінкою курсу).

Дану функціональність визначено в методі `get_queryset(self)`.

Окрім того визначено метод `get_context_data(self, **kwargs)` для створення базового контексту представлення, з яким взаємодіє шаблон.

Для відображення сформованого переліку курсів по 9 на сторінці встановлено атрибут класу `CoursesView`:

```
paginate_by = 9
```

Шаблоном даного представлення є «student/courses.html», відповідно всі шаблони були розподілені на ті, що відповідають роботі учня (знаходяться в теці «student»), і ті, що відповідають роботі викладача (знаходяться в теці «teacher»). Шаблони створено на основі використання безкоштовного HTML-шаблону Video Catalog [17].

Для звернення до даних, переданих з представлення, в шаблоні виконується звернення до об'єкту `object_list`, елементи якого переглядаються циклічно і для кожного з них визначається розміщення у відповідних елементах розмітки:

```
{% for course in object_list %}
<div class="col-lg-4 col-md-6 col-sm-12 tm-catalog-item">
  <div class="tm-bg-gray p-4">
    
    <h3 class="tm-text-primary mb-3"> <a href= "{% url 'studcourseview'
course.id %}">{{ course.getfirstparagraph }}</a></h3>
    <p>{{ course.info }}</p>
    <p>Завершується: {{ course.endlearn }}</p>
  </div>
</div>
{% endfor %}
```

У даному шаблоні використовується не весь опис курсу, а тільки перший абзац. Для цього в моделі курсу було створено окремий метод `getfirstparagraph`, описаний вище, який повертає перший абзац, він викликається у шаблоні.

Усі представлення в свою чергу розділено за базовими засобами представлення, покладеними в їх основу. Так засоби роботи з переліками представлено в файлі `lists.py`.

Засоби роботи з класами, побудованими на деталізації об'єкту представлено в файлі `details.py`.

Клас `StudCourseView` визначає представлення відображення деталізованої інформації про курс учню. Представлення відповідає шаблону `course.html`.

Метод `get_context_data(self, **kwargs)` перевизначений для формування даних для відповідного шаблону. Оскільки ці дані належать не тільки одній моделі, тобто не тільки курсу, але і матеріалам, тестам, анкетам тощо, а базовий клас `DetailView` передбачає роботу з однією моделлю.

У результаті клас повертає окрім даних про сам курс додатково через контекст:

- активні відеолекції курсу через ключ `videos`;
- активні текстові лекції курсу через ключ `texts`;
- активні тести курсу через ключ `tests`;
- незаповнені анкети курсу через ключ `ques`;
- результати учня за курсом через ключ;
- наступний матеріал курсу через ключ `nextmaterial`;
- перелік запитань учня і відповідей на них за ключем `messages`.

Оскільки учень має тільки раз заповнювати кожну анкету, а в БД відбувається збереження тільки відповідей на запитання, то формування конкретного переліку анкет для відображення учню на сторінці курсу відбувається наступним чином.

Спочатку виділяються анкети, які належать до даного курсу та які доступні в проміжок часу, до якого входить поточний час:

```
ques = Que.objects.all()
ques = ques.filter(course = currcourse)
ques = ques.filter(start__lte = nowtime, end__gte = nowtime)
ques = ques.order_by("-order")
```

Після цього реалізується виділення відповідей авторизованого користувача на запитання анкет та всіх відповідей на запитання:

```
allsubques = SubscriptionQueQuestionAnswers.objects.all()
allsubques = allsubques.filter(account = studaccount)
allqueanswers = QueQuestionAnswers.objects.all()
```

Після цього відбувається перегляд всіх активних анкет, виділених вище, і з них до остаточного переліку анкет включаються тільки ті, відповіді на запитання яких не входять (визначає змінна mode) до переліку відповідей користувача:

```
mode = False
for q in ques:
    allqueanswers2 = allqueanswers.filter(quequestion__que__id = q)
    for qq in allqueanswers2:
        if qq in allsubques:
            mode = True
    if not mode:
        readyques.append(qq)
context['ques'] = readyques
```

Засоби внесення нових даних в БД на основі класу `CreateView` представлено в файлі `create.py`. Засоби змінювання даних на основі класу `UpdateView` представлено в файлі `update.py`. Усі ці файли знаходяться в теці `views`, тобто формують відповідний підпакет. Серед цих засобів зокрема класи представлення створення тесту `CreateTestView`, представлення створення анкети `CreateQueView`, представлення оновлення відповідей користувача на тест `UpdateAccountTestAnswers`, представлення оновлення відповідей користувача на запитання анкети `UpdateSubscriptionQueQuestionAnswers`.

3.3 Висновки за розділом 3

Описано процедуру, яку використано для забезпечення персоналізації курсів на основі результатів проходження тестів учнем та його анкетування.

Для реалізації програми використано вебфреймворк `Django`. Представлено основні створені класи під час реалізації ПЗ та логіку їх створення.

4 ЕКСПЛУАТАЦІЯ ТА ТЕСТУВАННЯ ПРОГРАМИ

4.1 Призначення програми

Програма виконує персоналізацію процесу вивчення курсів на основі проблем, які виникають в учня, та побажань щодо питань, які його цікавлять або викликають проблеми. Програма призначена для організації вивчення онлайн курсів користувачами будь-якого віку з метою отримання додаткових знань, розширення компетенцій.

4.2 Умови виконання програми

Мінімальні системні вимоги до вебсервера включають:

- процесор: 4 ядра, тактова частота – 2,9 ГГц;
- оперативна пам'ять – 10 Гб;
- платформа – 32-розрядна або 64-розрядна;
- жорсткий диск – 1 Гб вільного простору.

4.3 Робота з програмою

Робота з програмою розпочинається з запуску вебсервера Apache на виконання. Тільки після цього можна ввести адресу, за якою на сервері розташовано вебсайт, в браузер, тоді відобразиться стартова сторінка вебзастосунку. Стартовою сторінкою є сторінка авторизації, тобто програма передбачає роботу тільки авторизованих користувачів.

Після авторизації учня, для нього відображається сторінка пошуку нових курсів (рис. 4.1). Ця сторінка містить зверху поле для введення назви курсу або її частини, а нижче – вибору варіанту сортування результатів (за оцінкою або за кількістю підписників на курс). Після вибору цих параметрів і натискання на кнопку «Шукати», нижче буде відображено результати пошуку. Результати пошуку відображаються рядками по 3 курси,

максимально 9 курсів на сторінку, для переходу між сторінками використовуються номери сторінок, розташовані внизу сторінки. Для переходу на потрібний курс потрібно натиснути на його назву, тоді відкриється сторінка курсу для користувача.

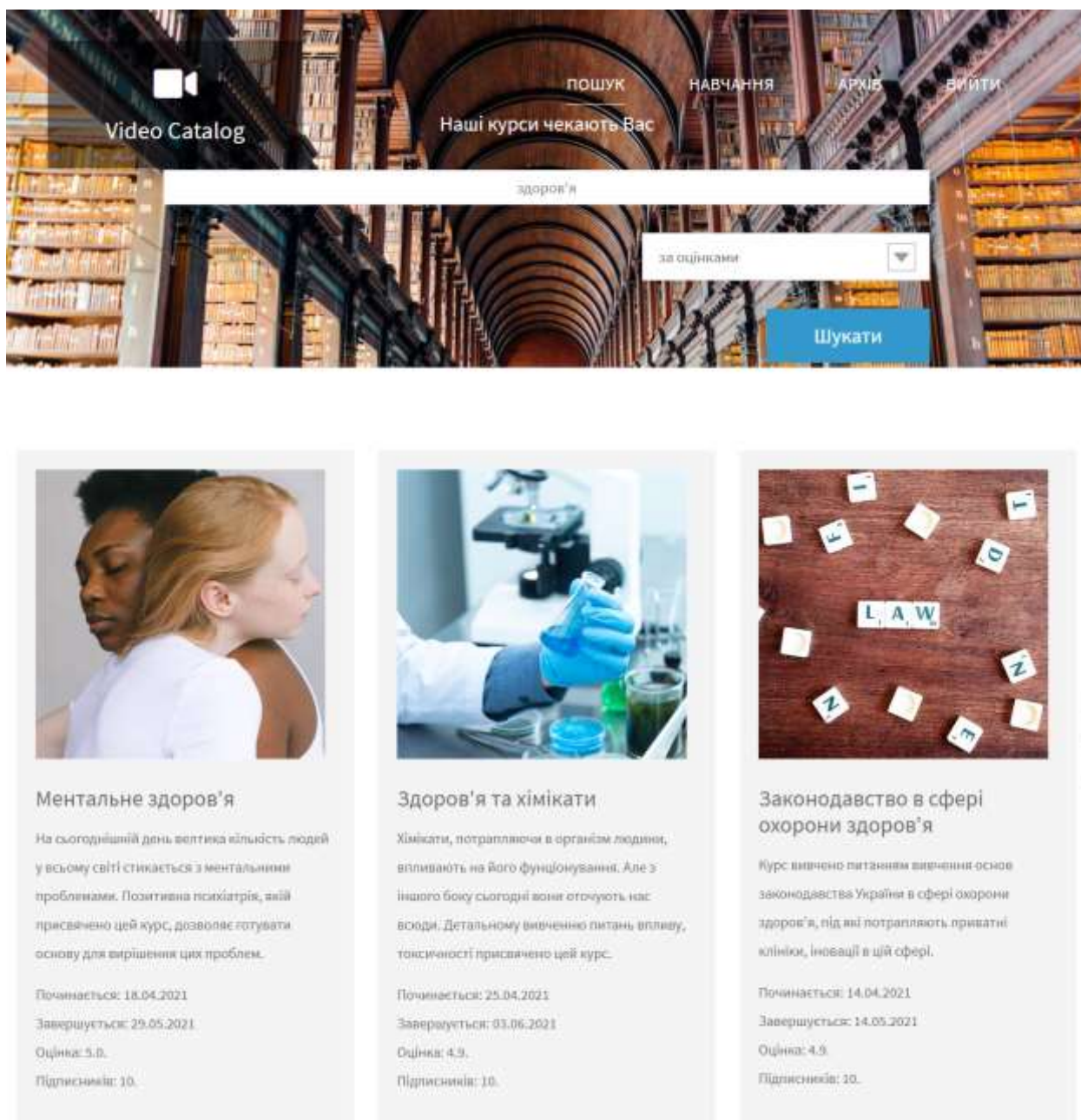


Рисунок 4.1 – Пошук нових курсів

Для кожного курсу відображається його зображення, назва, опис, дата початку, завершення, середня оцінка та кількість підписників за весь час.

Розділи меню включають «Пошук», який відображає описану вище сторінку, «Навчання», який відображає курси, на які підписаний користувач і які ще не завершені (рис. 4.2), «Архів», який відображає курси, на які підписаний користувач і які вже завершені, та «Вийти» для повернення до сторінки авторизації.

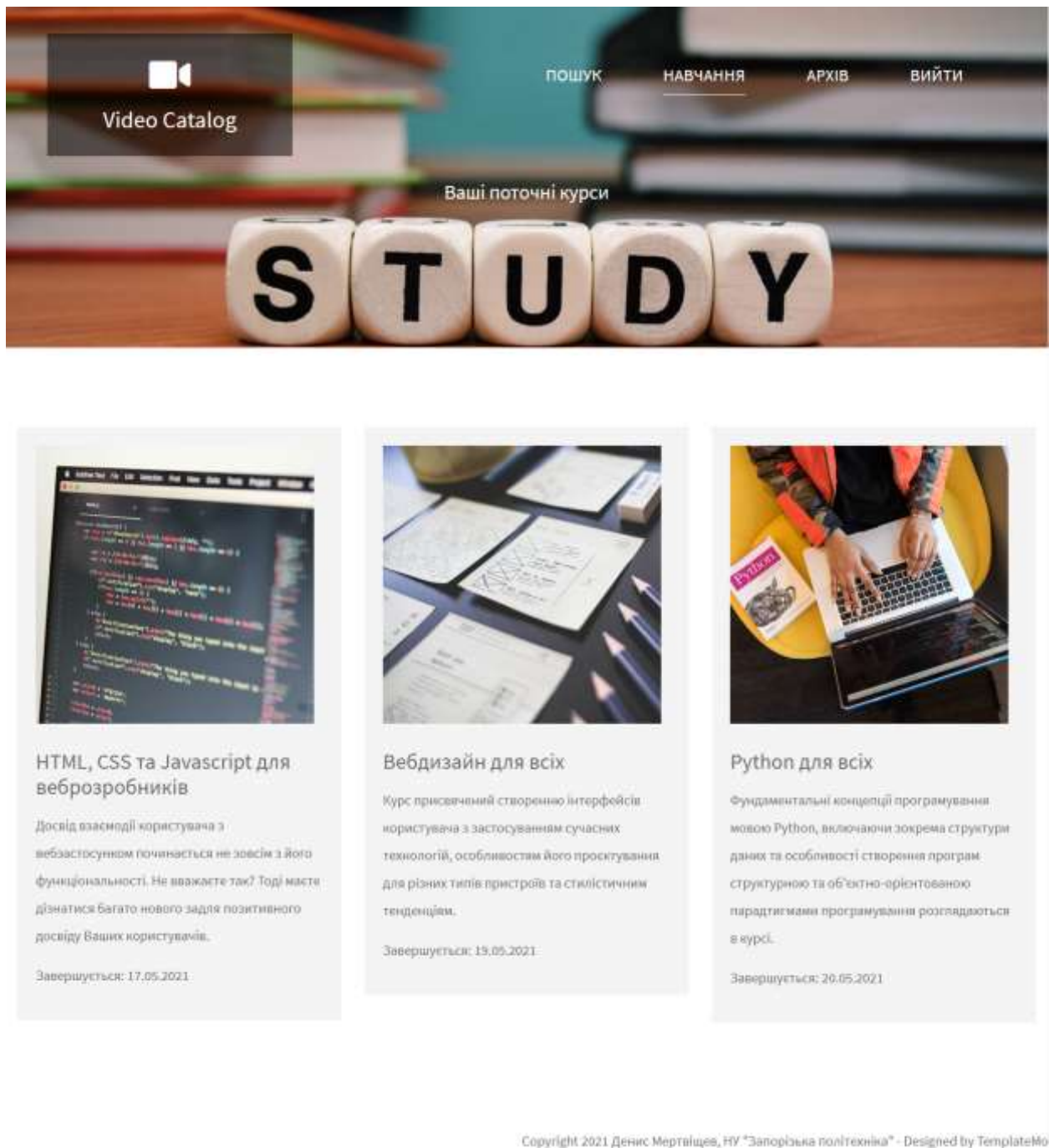


Рисунок 4.2 – Поточні курси, що вивчає авторизований користувач

Якщо користувач вже підписався на курс, то йому відкривається сторінка курсу у вигляді, представленому на рис. 4.3.

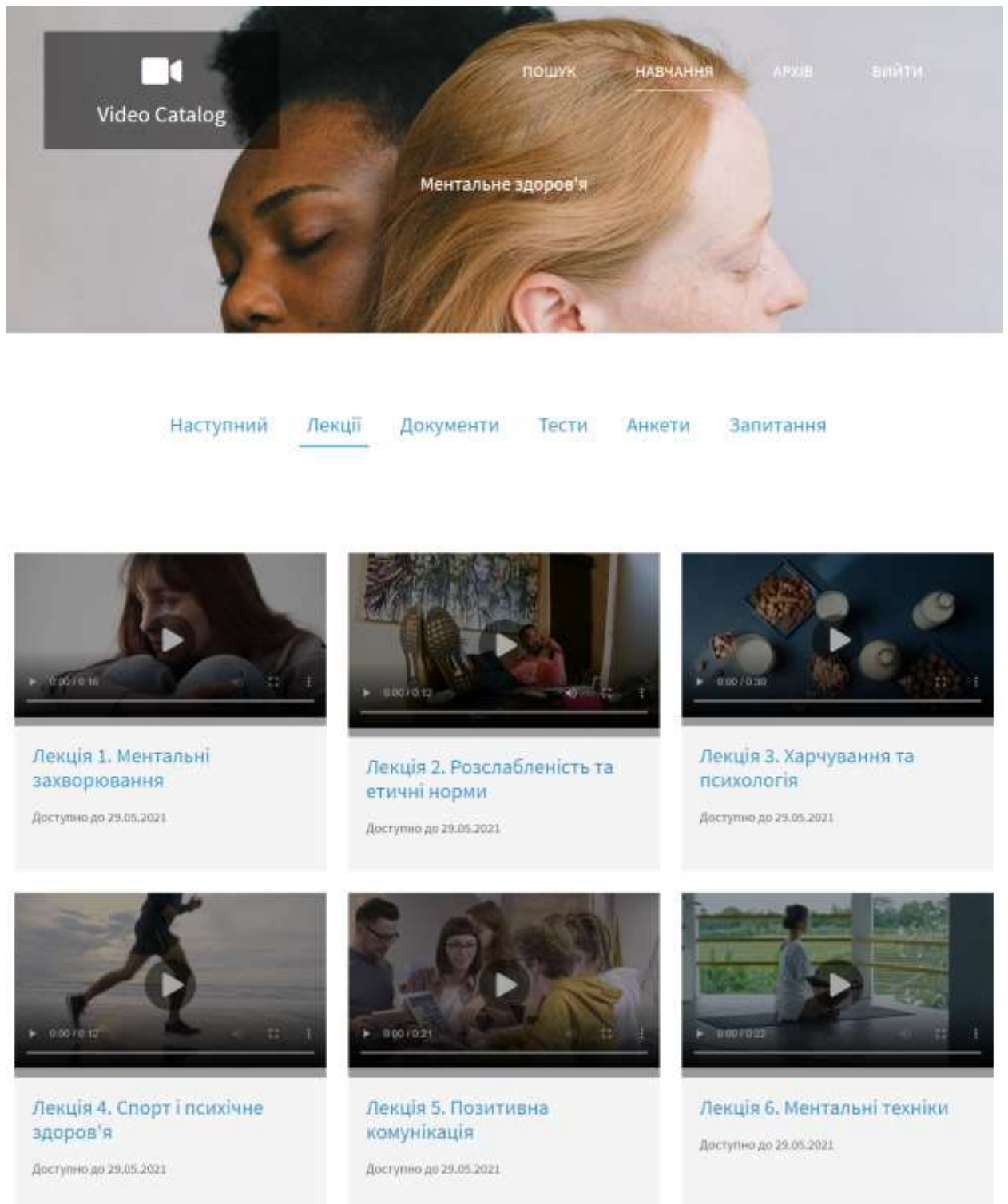


Рисунок 4.3 – Перегляд курсу, на який підписаний учень

При цьому усі матеріали розділені на групи за допомогою підменю: «Наступні», де відображається рекомендація для перегляду наступного матеріалу, «Лекції», де відображаються всі відеолекції, доступні учню на даний момент, за курсом (рис. 4.3), «Документи», де відображаються текстові лекції, «Тести», де відображаються тести, доступні користувачу за курсом, «Анкети», де відображаються анкети, які може заповнити учень, при чому кожна анкета відображається тільки до першого заповнення (як і тест), заповнити анкету повторно не можна, вона зникає з таблиці, та «Запитання», де учень може сформулювати запитання до викладача та переглянути відповідь від нього.

Якщо користувач ще не підписався на курс, то при переході з пошуку на сторінку курсу йому відображається назва курсу, його викладач, опис курсу, дата початку і завершення курсу, середня оцінка, кількість підписників та відгуки учнів з окремими оцінками (рис. 4.4).

Для того щоб підписатися на курс, потрібно натиснути на кнопку «Підписатися». Після цього курсу з'явиться на сторінці поточних курсів (рис. 4.2). Якщо курс ще не розпочався, то при цьому матеріали на сторінці курсу (рис. 4.3) відображені не будуть.

Залишити відгук за курсом користувач може тільки після завершення курсу або після отримання успішної оцінки раніше часу завершення вивчення курсу, тільки тоді в нього з'являється пункт підменю «Відгук», де на відповідній сторінці можна написати власний відгук та поставити оцінку. При цьому курс переміщується до архіву, адже час вивчення його завершився.

Під час проходження тесту учнем йому доступні одразу всі запитання, він може довільно між ними переміщуватися, натискаючи на відповідні кнопки з номерами запитань. Окрім того за кожним запитанням учень має обрати варіант відповіді та натиснути на кнопку «Зберегти», на останній сторінці доступна кнопка «Завершити».



Викладач: Іван Боженко

Підписатися

Хімікати, потрапляючи в організм людини, впливають на його функціонування. Але з іншого боку сьогодні вони оточують нас всюди. Детальному вивченню питань впливу, токсичності присвячено цей курс.

Розпочинається курс з питань токсикології: ми дізнаємось про те, як хімічні речовини впливають на наше здоров'я.

Біомоніторинг: як можна виміряти вміст цих хімічних речовин в нашому тілі, чому і як з цього можна зробити висновки?

Як вчені оцінюють вплив хімічних речовин на наше здоров'я? Лекції з цього питання будуть прочитані лікарем, який спеціалізується на екологічній медицині та епідеміології.

Починається: 25.04.2021

Завершується: 03.06.2021

Підписників: 10.

Середня оцінка: 4.9.

Відгуки (10):

Андрій Хорольський

оцінка: 5

Цей курс має пройти кожен, він важливий в сучасних умовах кожному, хто хоче керувати своїм організмом!

Рисунок 4.4 – Перегляд курсу, на який не підписаний учень

Приклад проходження тесту учнем приведено на рис. 4.5.

Після проходження тесту зі сторінки тестів за курсом у користувача навпроти відповідного курсу зникає кнопка «Пройти». Вона також зникає, якщо час проходження тесту ще не настав або вже минув. Тоді користувачу навпроти тесту вказується тільки його результат.

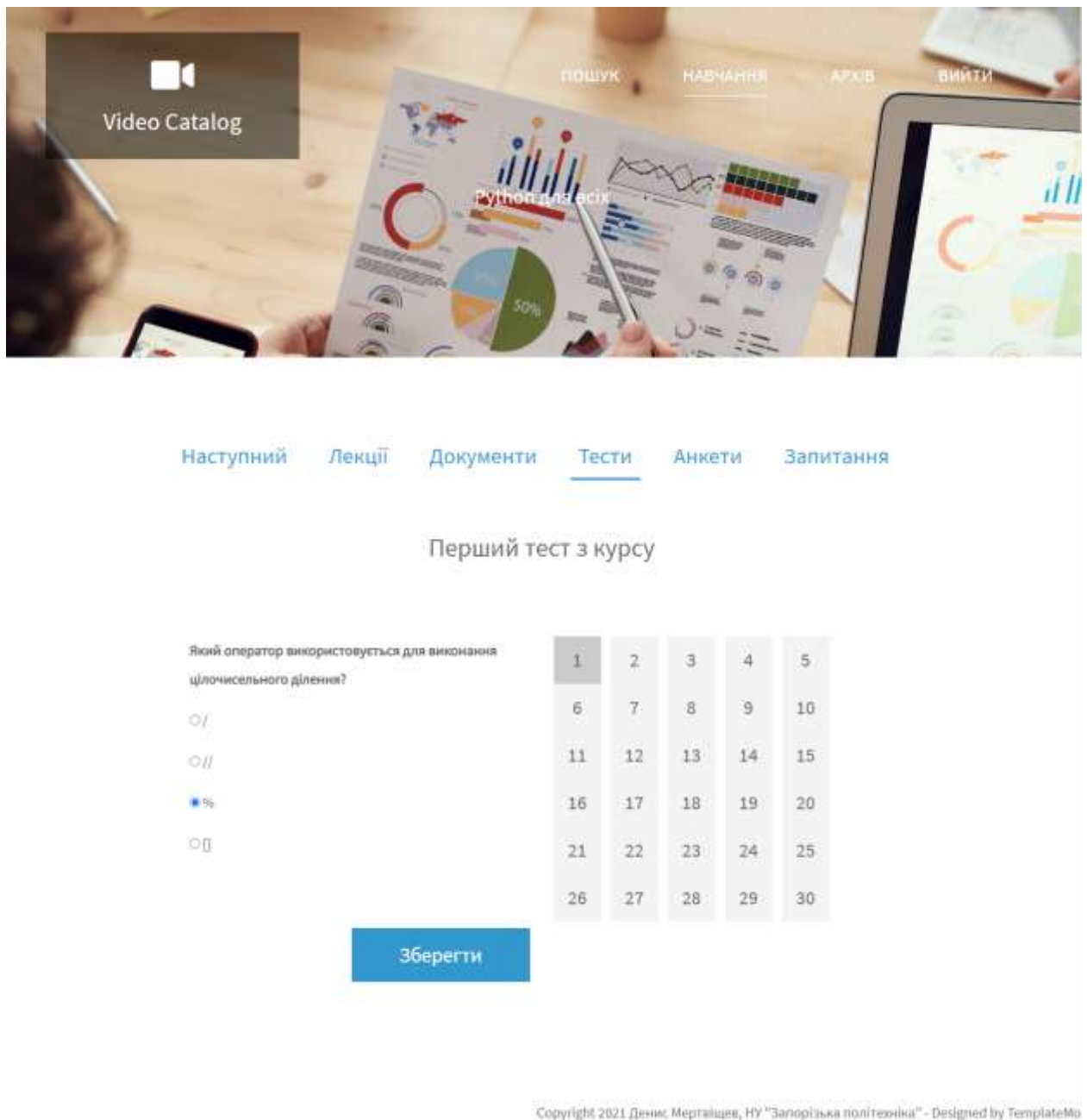


Рисунок 4.5 – Проходження тесту учнем

Приклад заповнення анкети приведено на рис. 4.6. Принцип заповнення анкети такий самий, що і тесту.

Анкета і тест впливають на персоналізацію. Тобто після заповнення анкети або проходження тесту відбувається перевірка правил, що стосуються даної анкети або тесту, визначається, які матеріали потрібно подати користувачу, які пропустити та їхній порядок, ці дані вносяться в БД, а

користувачу на сторінці наступного матеріалу відображається наступний матеріал.

The image shows a web application interface. At the top, there is a navigation bar with links: ПОШУК, НАВЧАННЯ, АРХІВ, and ВИЙТИ. Below this, a 'Video Catalog' section is visible, featuring a video player with a 'Python для всіх' (Python for everyone) video. The main content area displays a survey titled 'Опитування про навчання' (Survey about learning). The survey question is: 'На скільки добре Ви зрозуміли різницю між генераторами, ітераторами, циклами?' (How well do you understand the difference between generators, iterators, and loops?). The response options are:

- ☐ Зрозумів і часто застосовую хоча б 2 з них
- ☐ Зрозумів, але використовую тільки цикли
- ☒ Зрозумів частину
- ☐ До кінця і зараз не розумію

 A rating scale from 1 to 5 is also present, with 3 selected. A 'Зберегти' (Save) button is at the bottom. The footer contains copyright information: Copyright 2021, Денис Мертвіщев, НУ "Запорізька політехніка" - Designed by TemplateMo.

Рисунок 4.6 – Заповнення анкети учнем

Приклад перегляду такого рекомендованого матеріалу приведено на рис. 4.7. Тут доступне повне відео лекції (у даному випадку) або текст лекції (в іншому випадку).

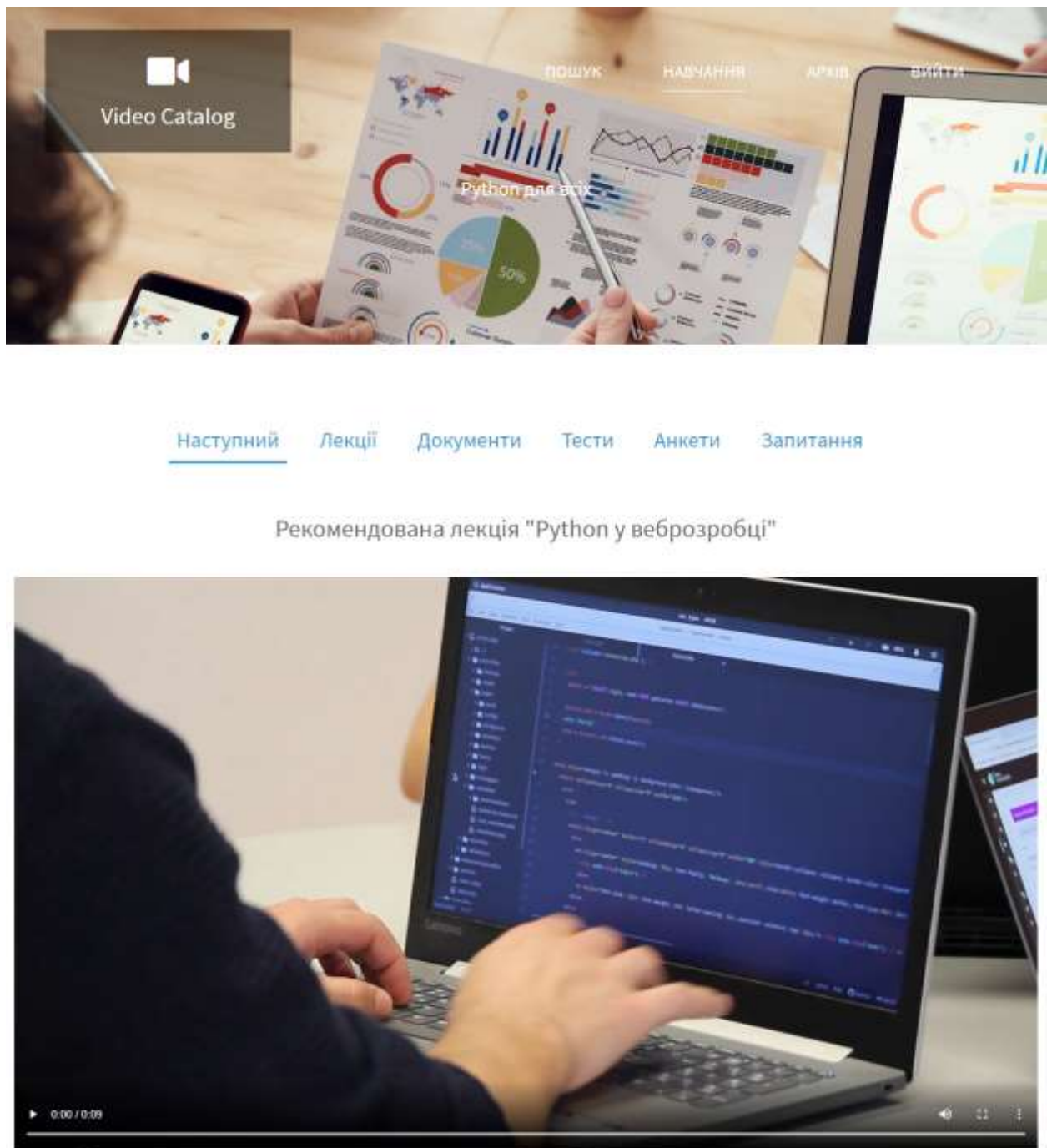


Рисунок 4.7 – Перегляд рекомендованого матеріалу

Принцип роботи викладача з програмою такий самий, що і учня. Пункти меню для викладача включають: «Створення», звідки можна одразу створити новий курс, «Курси», де доступний перелік всіх курсів викладача, звідки можна одразу перейти до редагування курсу (рис. 4.8), «Вийти», звідки можна вийти з облікового запису.

Під час редагування викладачу доступні пункти підменю: «Лекції», «Документи», «Тести», «Анкети», де він може редагувати та додавати нові відповідні матеріали, «Результати», де він може переглядати всі результати його учнів, відсортовані за часом вивчення курсу, «Запитання», де викладач може переглянути запитання до нього учнів та сформулювати відповідь, та «Відгуки», де викладач може ознайомитись з відгуками, які залишили учні, що завершили вивчення курсу.

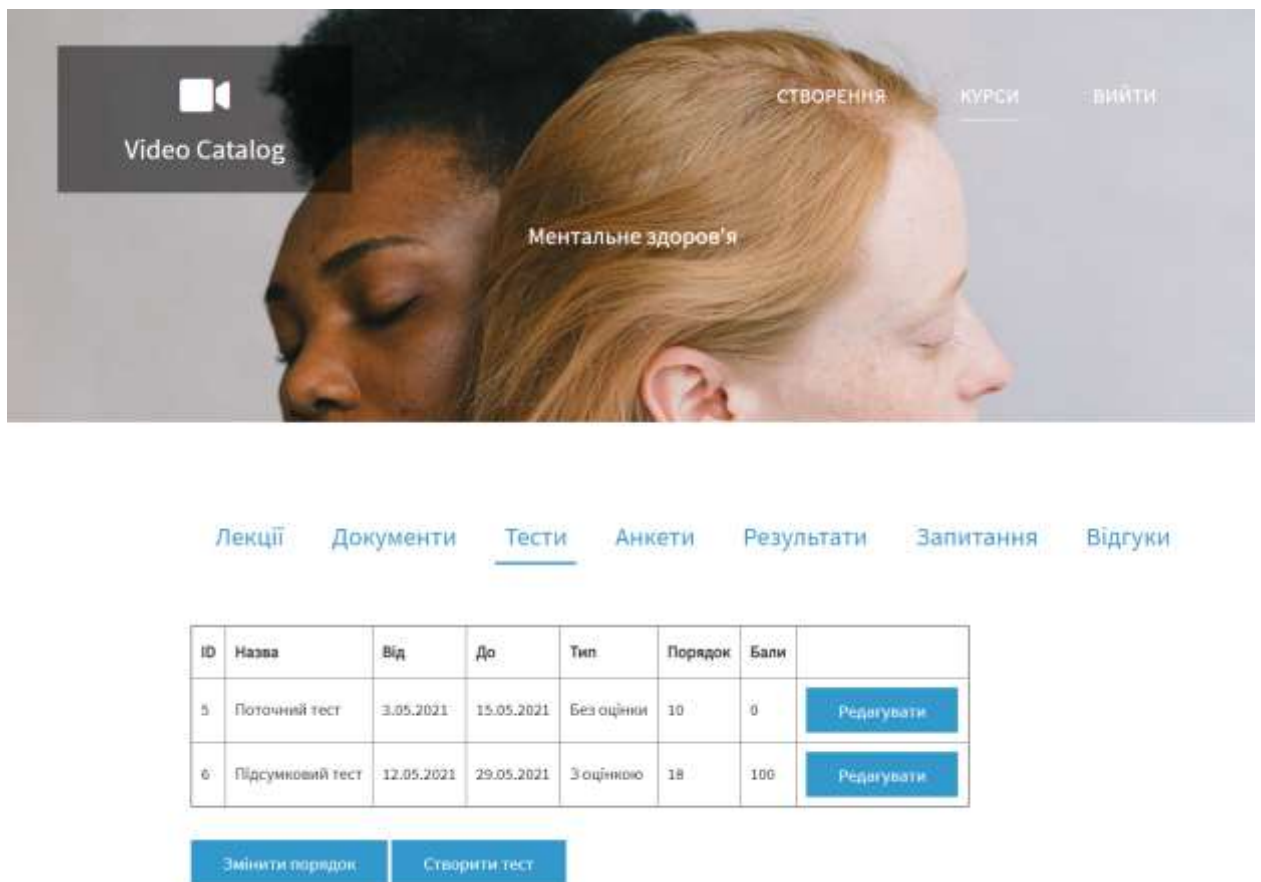


Рисунок 4.8 – Внесення матеріалів курсу викладачем

ВИСНОВКИ

Проблема онлайн навчання розглянута у вигляді створення онлайн курсів, які можуть застосовуватися на будь-якому етапі життя людини без необхідності формалізації освіти у одному з навчальних закладів, з персоналізованими матеріалами. Для розв'язання даної проблеми у вигляді розроблення відповідного ПЗ було отримано наступні результати:

- проаналізовано особливості та причини популярності онлайн курсів, виконано огляд програмних аналогів, включаючи Coursera, Udemu та EdX;
- обрано мову програмування мову Python для реалізації програми на основі виконаного порівняння з мовою Java;
- визначено функціональні вимоги до програми з моделюванням основних сценаріїв роботи учня та викладача з програмою у вигляді діаграм варіантів використання;
- визначено спосіб роботи з даними на основі БД з представленням структури цієї БД у вигляді відповідної схеми;
- реалізовано ПЗ у вигляді вебзастосунку на основі фреймворку Django;
- описано розроблені класи та функції під час реалізації вебзастосунку.

Розроблене ПЗ підтримує процес онлайн навчання з можливістю створення персоналізованої подачі відеоматеріалів і текстових матеріалів, визначаючи персоналізацію за результатами проведених тестів та заповнених учнем анкет за допомогою логічних правил. Учня надається можливість ставити запитання викладачу і отримувати на них відповіді у процесі навчання, залишати відгуки на курси після завершення навчання, які впливають на вибір наступних користувачів, що підбирають собі курси.

Усі завдання дипломної кваліфікаційної роботи повністю виконано.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Top reasons why online study is gaining popularity | WSU Online [Electronic resource]. – Access mode : <https://online.westernsydney.edu.au/blog/top-reasons-online-study-gaining-popularity/>
2. The Increasing Popularity of Online Education [Electronic resource]. – Access mode : <https://www.methodschoools.org/blog/the-increasing-popularity-of-online-education>
3. Koksall, I. The Rise Of Online Learning [Electronic resource] / I. Koksall. – Access mode : <https://www.forbes.com/sites/ilkerkoksall/2020/05/02/the-rise-of-online-learning/?sh=32bca39d72f3>
4. Coursera | Build Skills with Online Courses from Top Instructions [Electronic resource]. – Access mode : <https://www.coursera.org/>
5. Cooke, C. Coursera Review [Electronic resource] : How Popular Is This Elearning Platform? / C. Cooke. – Access mode : <https://upskillwise.com/reviews/coursera/>
6. Coursera Review [Electronic resource] : What It Is, How It Works, + More. – Access mode : <https://learntocodewith.me/reviews/coursera/>
7. Hughes, V. Coursera Review (2021) [Electronic resource] : Is Coursera Worth It? My Verdict. – Access mode : <https://geediting.com/coursera-review-2020-is-coursera-worth-it/>
8. Fenton, W. Coursera Review [Electronic resource] / W. Fenton. – Access mode : <https://www.pcmag.com/reviews/coursera>
9. Online Courses – Learn Anything, On Your Schedule | Udemy [Electronic resource]. – Access mode : <https://www.udemy.com/>
10. Duffy, J. Udemy Review [Electronic resource] / J. Duffy. – Access mode : <https://www.pcmag.com/reviews/udemy>
11. EdX | Free Online Courses by Harward, MIT & more [Electronic resource]. – Access mode : <https://www.edx.org/>

12. Review of Coursera, edX, and Udacity [Electronic resource] : What's Good and What's Bad. – Access mode : <https://www.edukatico.org/en/report/review-of-coursera-edx-and-udacity-what-s-good-and-what-s-bad>
13. Bloch, J. Effective Java [Text] / J. Bloch. – Boston : Addison-Wesley Professional, 2017. – 416 p.
14. Barry, P. Head First Python [Text] : A Brain-Friendly Guide / P. Barry. – Sebastopol, CA : O'Reilly Media, 2016. – 624 p.
15. Dubois, P. MySQL (Developer's Library) [Text] / P. Dubois. – Boston : Addison-Wesley Professional, 2013. – 1153 p.
16. Django Documentation [Electronic resource]. – Access mode : <https://docs.djangoproject.com/en/3.2/>
17. Free Template 552 Video Catalog [Electronic resource]. – Access mode : <https://templatemo.com/tm-552-video-catalog>

ДОДАТОК А
Технічне завдання

Вступ

Програма має виконувати персоналізацію процесу вивчення курсів на основі проблем, які виникають в учня, та побажань щодо питань, які його цікавлять або викликають проблеми.

A.1 Підстави для розробки

Підставою для виконуваної розробки в межах дипломної кваліфікаційної роботи бакалавра є завдання, що було визначене за темою «Програмне забезпечення підтримки вивчення персоналізованих курсів» і затверджене у відповідності з наказом № 103 від 30 березня 2021 р. за Національним університетом «Запорізька політехніка».

A.2 Призначення розробки

Програма має бути призначена для організації вивчення онлайн курсів користувачами будь-якого віку з метою отримання додаткових знань, розширення компетенцій.

A.3 Основні вимоги до програми

A.3.1 Вимоги до функціональних характеристик

За результатами виконання дипломної кваліфікаційної роботи бакалавра програма, що розробляється, повинно мати наступні функціональні характеристики:

- реєстрація облікових записів користувача і авторизація;
- пошук курсів;
- створення курсів;
- редагування даних курсів;

- внесення і редагування відеоматеріалів за курсами;
- створення тестів за курсом;
- проходження тестів за курсом;
- обчислення результату тесту;
- внесення відгуків після проходження курсу;
- визначення результату проходження курсу;
- формулювання запитань до викладача під час проходження курсу;
- створення текстових матеріалів за курсом;
- перегляд відеоматеріалів за курсом;
- читання текстового матеріалу за курсом;
- перегляд персоналізованих відеоматеріалів за курсом;
- читання персоналізованих тестових матеріалів за курсом;
- оформлення підписки на проходження курсу;
- перегляд кількості підписок на курси;
- перегляд власних результатів проходження курсу;
- перегляд переліку пройдених курсів;
- перегляд результатів навчання студентів за курсом;
- перегляд переліку курсів, навчання за якими відбувається;
- перегляд переліку створених курсів;
- перегляд запитань за курсом;
- перегляд відповідей на власні запитання;
- перегляд відгуків за курсом;
- перегляд доступних матеріалів за курсом;
- створення анкети за навчанням з курсу;
- заповнення анкети за навчанням з курсу.

A.3.2 Вимоги до надійності

Для забезпечення надійності має виконуватися валідація даних, що було введено користувачем в поля форм.

A.3.3 Умови експлуатації

Експлуатація програмного забезпечення має виконуватися на основі клієнт-серверної моделі. Для цього потрібно виділити окремий комп'ютер або скористатися виділеним сервером, який буде забезпечувати підтримку реалізації вебсервера. Вебсервер має бути доступним для запитів користувачів постійно.

A.3.4 Вимоги до складу та параметрів технічних засобів

Склад та параметри технічних засобів вебсервера було визначено для реалізації можливості доступу 300 користувачів одночасно. Відповідно вимоги можуть бути знижені, якщо потрібне забезпечення меншої кількості запитів, а для більшої кількості одночасних запитів вимоги навпаки мають бути збільшені. Мінімальні системні вимоги до вебсервера включають:

- процесор: 4 ядра, тактова частота – 2,9 ГГц;
- оперативна пам'ять – 10 Гб;
- платформа – 32-розрядна або 64-розрядна;
- жорсткий диск – 1 Гб вільного простору.

A.3.5 Вимоги до транспортування та збереження

Безпосередньо програма має транспортуватися та зберігатися через репозиторій вебсервера. Збереження на окремих фізичних носіях не передбачається.

A.3.6 Вимоги до програмної документації

Основою розробки даного програмного проекту є технічне завдання на нього.

Опис прийнятих рішень виконується в пояснювальній записці. У розділі «Експлуатація та тестування програми» має бути представлено опис процесу роботи з програмою. Інших програмних документів не передбачається.

А.4 Порядок контролю та приймання

Календарний план завдання на дипломну кваліфікаційну роботу визначає набір етапів, за якими має забезпечуватися виконання, а також результати, які мають бути отримані за виконання відповідного етапу. Саме на основі даних результатів має забезпечуватися приймання роботи.

ДОДАТОК Б
Текст програми

Б.1 Текст файла views/lists.py

```

from django.views.generic import ListView
from django.contrib.auth.mixins import LoginRequiredMixin
from .models import Course
from .models import Subscription
from datetime import date

class CoursesView (LoginRequiredMixin, ListView):
    model = Course
    template_name = 'student/courses.html'
    paginate_by = 9

    def get_queryset(self):
        searchparam = self.request.GET.get('searchby')
        ordertype = self.request.GET.get('ordertype')
        if ordertype == 'pop':
            ordertype = 'subscriptions'
        elif ordertype == 'feedb':
            ordertype = 'score'
        else:
            ordertype = 'subscriptions'
        createdcontext = Course.objects.filter(
            title__contains = searchparam.lower()
        ).order_by(ordertype)
        return createdcontext

    def get_context_data(self, **kwargs):
        createdcontext = super(CoursesView,
self).get_context_data(**kwargs)
        createdcontext ['searchby'] =
self.request.GET.get('searchby')
        ordertype = self.request.GET.get('ordertype')
        createdcontext ['ordertype'] = ordertype
        return createdcontext

class ArchiveCoursesView (LoginRequiredMixin, ListView):
    model = Subscription
    template_name = 'student/archcourses.html'
    paginate_by = 9

    def get_queryset(self):

```

```

courses = super(ArchiveCoursesView, self).get_queryset()
studaccount = self.request.user
courses = courses.filter(account = studaccount))
nowtime = date.today()
courses = courses.filter(endlearn__lte = nowtime)
courses = courses.order_by("-endlearn")
return courses

```

```

class LearnCoursesView (LoginRequiredMixin, ListView):
    model = Subscription
    template_name = 'student/learncourses.html'

    def get_queryset(self):
        courses = super(LearnCoursesView, self).get_queryset()
        studaccount = self.request.user
        courses = courses.filter(account = studaccount))
        nowtime = date.today()
        courses = courses.filter(endlearn__gt = nowtime)
        courses = courses.order_by("-startlearn")
        return courses

```

```

class TestView (LoginRequiredMixin, ListView):
    model = TestQuestion
    template_name = 'student/test.html'
    paginate_by = 1

    def get_queryset(self):
        testnum = self.request.GET.get('testnum')
        currtest = Test.objects.get(id = testnum)
        coursenum = self.request.GET.get('coursenum')
        currcourse = Course.objects.get(id = coursenum)
        questions = TestQuestion.objects.all()
        questions = questions.filter(test = currtest)
        studaccount = self.request.user
        qs = []
        testob = Test.objects.get(course = currcourse, id =
testnum)
        sub = Subscription.objects.get(course = currcourse,
account = studaccount)
        t = AccountTest.objects.get(test = testob, subscription
= sub)
        if testob and sub and not t:

```

```

        if testob.startlearn > nowtime and testob.endlearn
<= nowtime:
            for q in questions:
                an
TestQuestionAnswer.objects.all().filter(testquestion = q)
                an.order_by('order')
                testq = q
                testq['answers'] = an
                qs.append(testq)
            createdcontext = qs
            return createdcontext

    def get_context_data(self, **kwargs):
        createdcontext = super(TestView,
self).get_context_data(**kwargs)
        createdcontext['testnum'] =
self.request.GET.get('testnum')
        return createdcontext

class QueView (LoginRequiredMixin, ListView):
    model = QueQuestions
    template_name = 'student/que.html'
    paginate_by = 1

    def get_queryset(self):
        quenum = self.request.GET.get('quenum')
        currque = Test.objects.get(id = quenum)
        coursenum = self.request.GET.get('coursenum')
        currcourse = Course.objects.get(id = coursenum)
        questions = QueQuestions.objects.all()
        questions = questions.filter(que = currque)
        studaccount = self.request.user
        qs = []
        ob = Que.objects.get(course = currcourse, id = quenum)
        sub = Subscription.objects.get(course = currcourse,
account = studaccount)
        t
SubscriptionQueQuestionAnswers.objects.get(answer__quequestion__que
ob, account = studaccount)
        if ob and sub and not t:
            if ob.start > nowtime and ob.end <= nowtime:
                for q in questions:
                    an
QueQuestionAnswers.objects.all().filter(answer__quequestion = q)

```

```

        queq = q
        queq['answers'] = queq
        qs.append(queq)
    createdcontext = qs
    return createdcontext

    def get_context_data(self, **kwargs):
        createdcontext = super(TestView,
self).get_context_data(**kwargs)
        createdcontext['testnum'] =
self.request.GET.get('testnum')
        return createdcontext

class CreatedCoursesView (LoginRequiredMixin, ListView):
    model = Course
    template_name = 'teacher/courses.html'
    paginate_by = 9

    def get_queryset(self):
        courses = super(CreatedCoursesView, self).get_queryset()
        teacheraccount = self.request.user
        courses = courses.filter(account = teacheraccount)
        courses = courses.order_by("-startlearn")
        return courses

```

Б.2 Текст файла views/details.py

```

from django.contrib.auth.mixins import LoginRequiredMixin
from django.views.generic.detail import DetailView
from .models import Course
from .models import Material
from datetime import date
from .models import AccountTest
from .models import Que
from .models import QueQuestions
from .models import QueQuestionAnswers
from .models import SubscriptionQueQuestionAnswers
from .models import Subscription
from .models import Communication
from .person import personalization

```

```

class CourseView (LoginRequiredMixin, DetailView):
    model = Course
    template_name = 'teacher/course.html'

    def get_context_data(self, **kwargs):
        context = super().get_context_data(**kwargs)
        currcourse = context['object']
        videos = Material.objects.all()
        videos = videos.filter(type = 0)
        videos = videos.filter(course = currcourse)
        videos = videos.order_by("-order")
        context['videos'] = videos

        texts = Material.objects.all()
        texts = texts.filter(type = 1)
        texts = texts.filter(course = currcourse)
        texts = texts.order_by("-order")
        context['texts'] = texts

        tests = Test.objects.all()
        tests = tests.filter(course = currcourse)
        tests = tests.order_by("-order")
        context['tests'] = tests

        ques = Que.objects.all()
        ques = ques.filter(course = currcourse)
        ques = ques.order_by("-order")
        context['ques'] = ques

        messages = Communication.objects.all()
        contmessages = []
        subs = Subscription.objects.all()
        subs = subs.filter(course = currcourse)
        for sub in subs:
            submessages = messages.filter(subscription = sub)
            contmessages.append(submessages)
        contmessages = contmessages.order_by("-quedate")
        context['messages'] = contmessages

        subs = subs.order_by("-endlearn")
        context['results'] = subs

        reviews = []
        for sub in subs:
            if sub.feedback:

```

```

        reviews.append(sub)
    reviews = reviews.order_by("-feeddate")
    context['reviews'] = reviews

    return context

```

```

class MaterialView (LoginRequiredMixin, DetailView):
    model = Material
    template_name = 'student/material.html'

```

```

class QueView (LoginRequiredMixin, DetailView):
    model = Que
    template_name = 'student/que.html'

```

```

    def get_context_data(self, **kwargs):

        context = super().get_context_data(**kwargs)
        currque = context['object']
        questions = QueQuestions.objects.all()
        questions = questions.filter(que = currque)
        ans = QueQuestionAnswers.objects.all()
        qs = []
        for q in qs:
            cq = q
            ans2 = ans.filter(quequestion = q)
            cq['answers'] = ans2
            qs.append(cq)
        context['questions'] = qs

```

```

class StudCourseView (LoginRequiredMixin, DetailView):
    model = Course
    template_name = 'student/course.html'

```

```

    def get_context_data(self, **kwargs):

        context = super().get_context_data(**kwargs)
        currcourse = context['object']
        videos = Material.objects.all()
        videos = videos.filter(type = 0)
        videos = videos.filter(course = currcourse)
        nowtime = date.today()

```

```

        videos = videos.filter(startlearn__lte = nowtime,
endlearn__gte = nowtime)
        videos = videos.order_by("-order")
        context['videos'] = videos

        texts = Material.objects.all()
        texts = texts.filter(type = 1)
        texts = texts.filter(course = currcourse)
        texts = texts.filter(startlearn__lte = nowtime,
endlearn__gte = nowtime)
        texts = texts.order_by("-order")
        context['texts'] = texts

        tests = Test.objects.all()
        tests = tests.filter(course = currcourse)
        tests = tests.filter(startlearn__lte = nowtime,
endlearn__gte = nowtime)
        tests = tests.order_by("-order")
        studaccount = self.request.user
        sub = Subscription.objects.get(account = studaccount,
course = currcourse)
        forcontexttests = []
        for test in tests:
            alltests = AccountTest.objects.all()
            alltests = alltests.filter(subscription = sub)
            if test in alltests:
                t = test
                t['result'] = alltests.get(id = t.id)
                forcontexttests.append(t)
        context['tests'] = forcontexttests

        ques = Que.objects.all()
        ques = ques.filter(course = currcourse)
        ques = ques.filter(star__lte = nowtime, end__gte =
nowtime)

        ques = ques.order_by("-order")
        forcontexttests = []
        readyques = []
        allsubques =
SubscriptionQueQuestionAnswers.objects.all()
        allsubques = allsubques.filter(account = studaccount)
        allqueanswers = QueQuestionAnswers.objects.all()
        mode = False
        for q in ques:

```

```

        allqueanswers2
allqueanswers.filter(quequestion__que__id = q)
    for qq in allqueanswers2:
        if qq in allsubques:
            mode = True
    if not mode:
        readyques.append(qq)
    context['ques'] = readyques
    context['result'] = sub

    mater = personalization.getnext(sub)
    context['nextmaterial'] = mater

    mess = Communication.objects.all()
    mess = mess.filter(subscription = sub)
    mess = mess.order_by("-ansdate")
    context['messages'] = mess

    return context

```

Б.3 Текст файла models.py

```

from django.db import models

class Account (models.Model):
    email = models.CharField(max_length=50, unique=True)
    password = models.CharField(max_length=50)
    surname = models.CharField(max_length=100)
    name = models.CharField(max_length=100)
    phone = models.CharField(max_length=13)
    birth = models.DateTimeField()
    profile = models.CharField(max_length=255)
    info = models.CharField(max_length=255)
    acctype = models.BinaryField(max_length=1)

    def __str__(self):
        return self.email

class Course (models.Model):
    title = models.CharField(max_length=255)
    info = models.TextField()

```

```

image = models.CharField(max_length=255)
startlearn = models.DateTimeField()
endlearn = models.DateTimeField()
created = models.DateTimeField()
subscriptions = models.IntegerField()
account = ForeignKey (Account, on_delete=models.CASCADE)
score = models.FloatField()

```

```

def __str__(self):
    return self.title

```

```

def getfirstparagraph(self):
    return self.title.split('<br>')[0]

```

```

class Meta:
    ordering = ['-subscriptions']

```

```

class Material (models.Model):
    title = models.CharField(max_length=255)
    course = ForeignKey (Course, on_delete=models.CASCADE)
    order = models.IntegerField()
    type = models.BinaryField(max_length=1)
    file = models.FilePathField()
    info = models.CharField(max_length=255)
    group = models.IntegerField()
    startlearn = models.DateTimeField()
    endlearn = models.DateTimeField()

```

```

def __str__(self):
    m = ''
    if self.type == 0
        m = 'Відеолекція: '
    else m = 'Текстова лекція: '
    return m + self.title

```

```

def getstrtype(self):
    if type == 0:
        return 'відеолекція'
    else:
        return 'текстова лекція'

```

```

class Meta:
    ordering = ['-order']

```

```

class Subscription (models.Model):
    account = ForeignKey (Account, on_delete=models.CASCADE)
    course = ForeignKey (Course, on_delete=models.CASCADE)
    startlearn = models.DateTimeField()
    endlearn = models.DateTimeField()
    result = models.FloatField()
    feedback = models.TextField()
    feedpoints = models.IntegerField()
    feeddate = models.DateTimeField()
    lastorder = models.IntegerField()

    def __str__(self):
        return str(self.course) + '/' + str(self.account)

class SubscriptionMaterial (models.Model):
    subscription = ForeignKey (Subscription,
on_delete=models.CASCADE)
    material = ForeignKey (Material, on_delete=models.CASCADE)
    watched = models.DateTimeField()
    order = models.IntegerField ()
    show = models.BinaryField(max_length=1)

class Test (models.Model):
    title = models.CharField(max_length=100)
    course = ForeignKey (Course, on_delete=models.CASCADE)
    order = models.IntegerField()
    starttest = models.DateTimeField()
    endtest = models.DateTimeField()
    typetest = models.BinaryField(max_length=1)
    points = models.FloatField()

    def __str__(self):
        return self.title
    class Meta:
        ordering = ['-order']

class TestQuestion (models.Model):
    qtext = models.CharField(max_length=255)
    test = ForeignKey (Test, on_delete=models.CASCADE)
    order = models.IntegerField()

    def __str__(self):

```

```

        return self.qtext

    class Meta:
        ordering = ['-order']

class TestQuestionAnswer (models.Model):
    atext = models.CharField(max_length=255)
    testquestion = ForeignKey (TestQuestion,
on_delete=models.CASCADE)
    order = models.IntegerField()
    right = models.BinaryField(max_length=1)

    def __str__(self):
        return self.atext

    class Meta:
        ordering = ['-order']

class Que (models.Model):
    course = ForeignKey (Course, on_delete=models.CASCADE)
    order = models.IntegerField()
    start = models.DateTimeField()
    end = models.DateTimeField()

    class Meta:
        ordering = ['-order']

class QueQuestions (models.Model):
    que = ForeignKey (Que, on_delete=models.CASCADE)
    order = models.IntegerField()
    qtext = models.CharField(max_length=255)

    class Meta:
        ordering = ['-order']

class QueQuestionAnswers (models.Model):
    quequestion = ForeignKey (QueQuestions,
on_delete=models.CASCADE)
    qtext = models.CharField(max_length=255)

```

```

class SubscriptionQueQuestionAnswers (models.Model):
    account          =          ForeignKey          (QueQuestionAnswers,
on_delete=models.CASCADE)
    answer = models.IntegerField()
    adate = models.DateTimeField()

class AccountTest (models.Model):
    subscription      =          ForeignKey          (Subscription,
on_delete=models.CASCADE)
    test = ForeignKey (Test, on_delete=models.CASCADE)
    tdate = models.DateTimeField()
    points = models.FloatField()
    class Meta:
        ordering = ['-tdate']

class AccountTestAnswers (models.Model):
    accounttest       =          ForeignKey          (AccountTest,
on_delete=models.CASCADE)
    testquestionanswer =      ForeignKey          (TestQuestionAnswer,
on_delete=models.CASCADE)
    points = models.FloatField()

class Communication (models.Model):
    subscription      =          ForeignKey          (Subscription,
on_delete=models.CASCADE)
    question = models.CharField(max_length=255)
    qudate = models.DateTimeField()
    answer = models.CharField(max_length=255)
    ansdate = models.DateTimeField()

    class Meta:
        ordering = ['-subscription']

```

Б.4 Текст файла learncourses.html

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">

```

```

        <meta name="viewport" content="width=device-width, initial-
scale=1.0">
        <title>Програмне забезпечення підтримки вивчення
персоналізованих курсів</title>
        <link rel="stylesheet" href="fontawesome/css/all.min.css">
        <link
href="https://fonts.googleapis.com/css2?family=Source+Sans+Pro&display
=swap" rel="stylesheet">
        <link rel="stylesheet" href="css/bootstrap.min.css">
        <link rel="stylesheet" href="css/templatemo-video-
catalog.css">
        <!--

TemplateMo 552 Video Catalog

https://templatemo.com/tm-552-video-catalog

-->
</head>
<body>
    <div class="tm-page-wrap mx-auto">
        <div class="position-relative">
            <div class="position-absolute tm-site-header">
                <div class="container-fluid position-
relative">
                    <div class="row">
                        <div class="col-7 col-md-4">
                            <a href="index.html" class="tm-bg-
black text-center tm-logo-container">
                                <i class="fas fa-video tm-site-
logo mb-3"></i>
                                <h1 class="tm-site-name">Video
Catalog</h1>
                            </a>
                        </div>
                        <div class="col-5 col-md-8 ml-auto mr-
0">
                            <div class="tm-site-nav">
                                <nav class="navbar navbar-
expand-lg mr-0 ml-auto" id="tm-main-nav">
                                    <button class="navbar-
toggler tm-bg-black py-2 px-3 mr-0 ml-auto collapsed" type="button"
data-toggle="collapse"
data-target="#navbar-nav" aria-controls="navbar-nav">

```

```

aria-label="Toggle navigation">
aria-expanded="false"
<span>
  <i class="fas fa-bars tm-menu-closed-icon"></i>
  <i class="fas fa-times tm-menu-opened-icon"></i>
</span>
</button>
<div class="collapse navbar-collapse tm-nav" id="navbar-nav">
  <ul class="navbar-nav text-uppercase">
    <li class="nav-item">
      <a class="nav-link tm-nav-link" href="courses.html">Пошук</a>
    </li>
    <li class="nav-item active">
      <a class="nav-link tm-nav-link" href="learncourses.html">Навчання<span class="sr-only">(current)</span></a>
    </li>
    <li class="nav-item">
      <a class="nav-link tm-nav-link" href="archcourses.html">Архів</a>
    </li>
    <li class="nav-item">
      <a class="nav-link tm-nav-link" href="logout.html">Вийти</a>
    </li>
  </ul>
</div>
</nav>
</div>
</div>
</div>
</div>
<div class="tm-welcome-container tm-fixed-header tm-fixed-header-2">
  <div class="text-center">

```

```

                                <p class="pt-5 px-3 tm-welcome-text
tm-welcome-text-2 mb-1 mt-lg-0 mt-5 text-white mx-auto">Ваші поточні
курсисти</p>
                                </div>
                            </div>

                            <div id="tm-fixed-header-bg"></div>
                        </div>
                        <main>
                            <div class="container-fluid px-0">
                                <div class="mx-auto tm-content-container mt-
4 px-3">
                                    <div class="row tm-catalog-item-list
mb-4">
                                        {% for course in object_list %}
                                        <div class="col-lg-4 col-md-6 col-sm-
12 tm-catalog-item">
                                            <div class="tm-bg-gray p-4">
                                                
                                                <h3 class="tm-text-primary mb-
3"> <a href= "{% url 'studcourseview' course.id %}">{{
course.getfirstparagraph }}</a></h3>
                                                <p>{{ course.info }}</p>
                                                <p>Завершується: {{
course.endlearn }}</p>
                                            </div>
                                        </div>
                                        {% endfor %}
                                    </div>
                                </div>
                            </main>

                            <footer class="row pt-5">
                                <div class="col-12">
                                    <p class="text-right">Copyright 2021 Денис
Мертвіщев, НУ "Запорізька політехніка"

                                - Designed by <a
href="https://templatemo.com"
target="_parent">TemplateMo</a></p>
                                </div>
                            </footer>
                        </div>

```

```
</div>

<script src="js/jquery-3.4.1.min.js"></script>
<script src="js/bootstrap.min.js"></script>
<script src="js/parallax.min.js"></script>
</body>
</html>
```

ДОДАТОК В
Слайди презентації



Рисунок В.1 – Слайд № 1

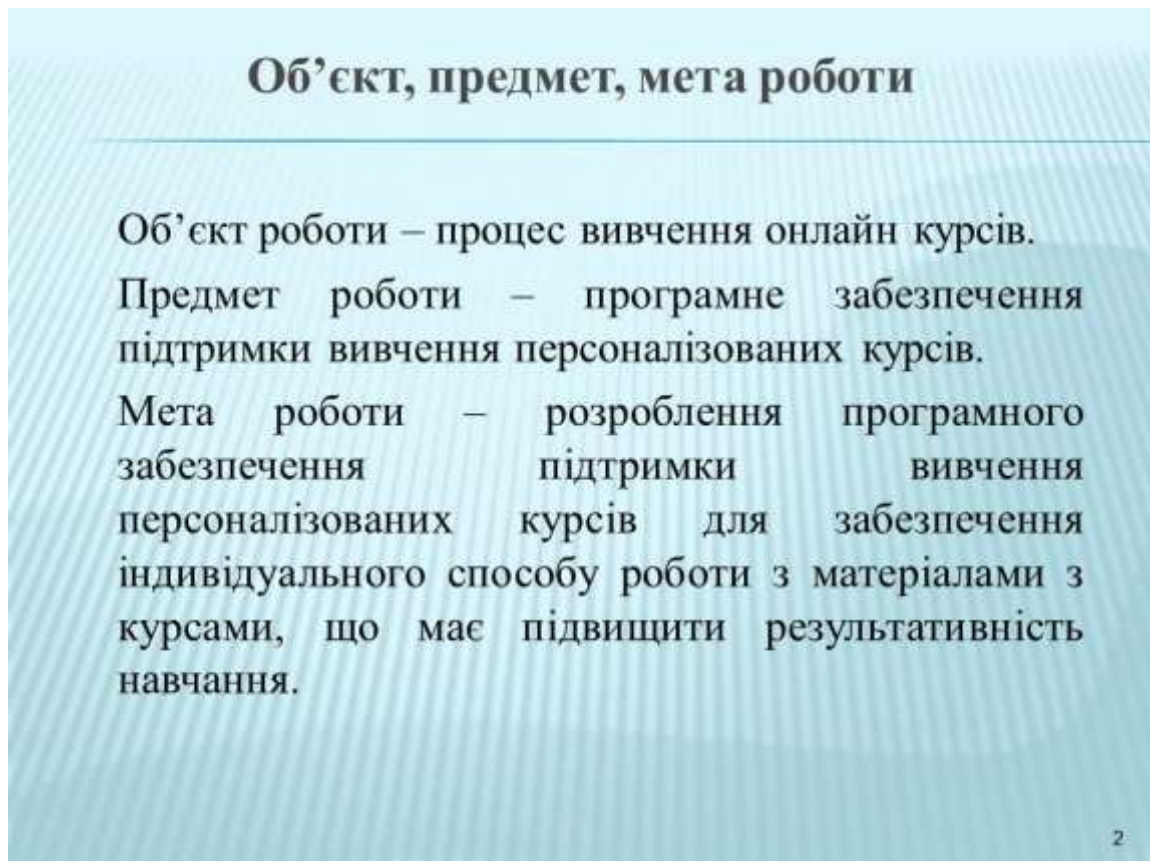


Рисунок В.2 – Слайд № 2

Завдання роботи

- ✓ виділення вимог до програмного забезпечення підтримки вивчення персоналізованих курсів;
- ✓ вибір мови і засобів проєктування і розробки;
- ✓ проєктування програмного забезпечення;
- ✓ реалізація і налагодження програмного забезпечення.

3

Рисунок В.3 – Слайд № 3

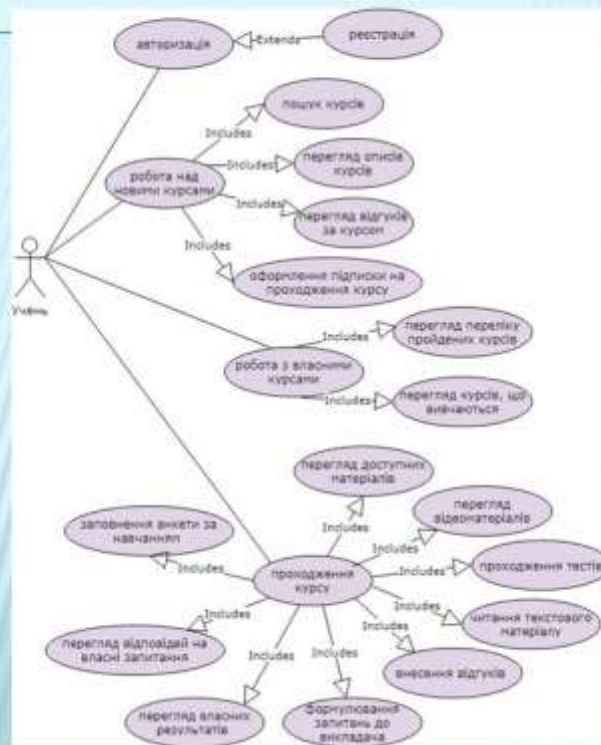
Порівняння мов програмування для реалізації програми

Покривник	Python	Java
Синтаксис	Простий	Складніший
Керування пам'яттю	Автоматичне	Автоматичне
Типізація	Динамічна	Статична
Швидкість виконання	Середня (за рахунок використання інтерпретатора)	Менша
Вивчення нових можливостей	Легке	Достатньо складне (тривале)
Вебфреймворки	Spring	Django, Flask

4

Рисунок В.4 – Слайд № 4

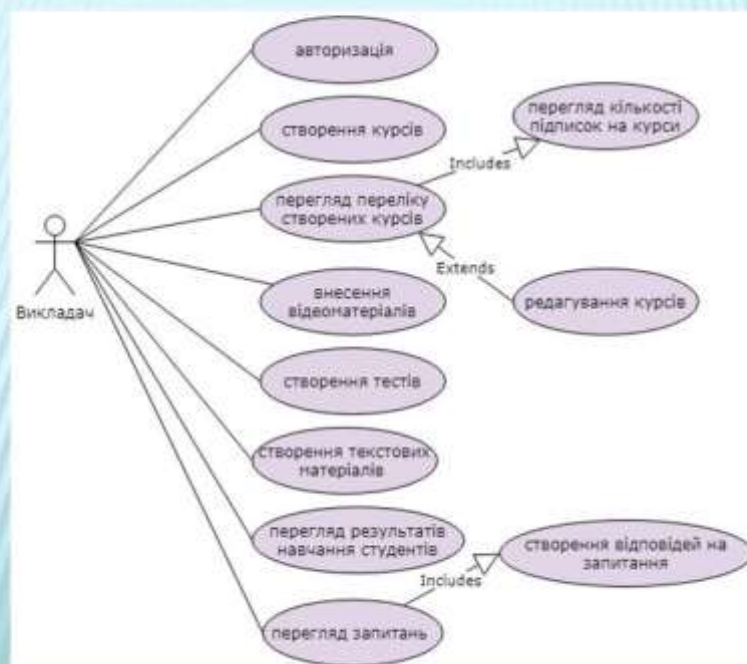
Діаграми варіантів використання для учня



5

Рисунок В.5 – Слайд № 5

Діаграма варіантів використання для викладача



6

Рисунок В.6 – Слайд № 6

Діаграма діяльності навчання

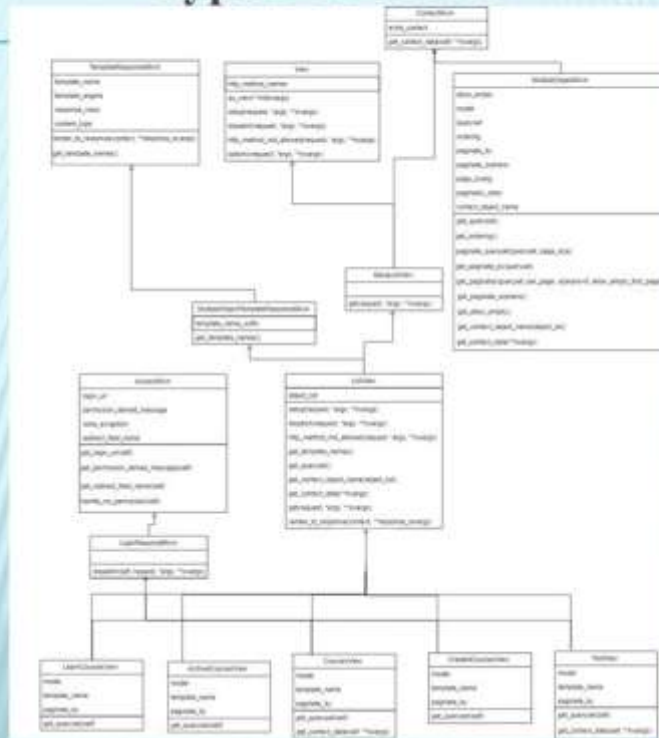
```

graph TD
    Start(( )) --> Step1[Використання доступних матеріалів]
    Step1 --> Step2[Використання доступних методів]
    Step2 --> Step3[Використання доступних вміст]
    Step3 --> Step4[Платити за матеріал]
    Step4 --> Step5[Використання програмного посереддя]
    Step5 --> Step6[Закінчення]
    Step5 --> Step7{ }
    Step7 -- час не закінчився? --> Step1
    Step7 -- час закінчився --> Step8[Підведення підсумку]
    Step8 --> End(( ))
    Step6 --> Step9{ }
    Step9 -- перевірка результату --> Step10[Перевірка результату]
    Step9 -- перевірка процесу --> Step11[Аналіз процесу]
    Step10 --> Step12{ }
    Step11 --> Step12
    Step12 --> Step13[Аналіз процесу]
    Step13 --> Step14[Використання допоміжного матеріалу]
    Step14 --> Step15[Підведення підсумку]
    Step15 --> End
  
```

The diagram illustrates the learning process flow. It begins with a start node leading to a sequence of steps: 'Використання доступних матеріалів' (Using available materials), 'Використання доступних методів' (Using available methods), and 'Використання доступних вміст' (Using available content). This is followed by 'Платити за матеріал' (Pay for material) and 'Використання програмного посереддя' (Using software). A decision diamond follows, with the 'yes' path leading to 'Закінчення' (End) and the 'no' path leading to a merge point. From the merge point, the flow goes to 'Перевірка результату' (Check result) and 'Аналіз процесу' (Analyze process). Another decision diamond follows, with the 'yes' path leading to 'Перевірка результату' and the 'no' path leading to 'Аналіз процесу'. Both paths lead to a merge point, followed by 'Аналіз процесу', 'Використання допоміжного матеріалу' (Using auxiliary material), and finally 'Підведення підсумку' (Summary), which leads to the end node. A feedback loop exists from a decision diamond after 'Використання програмного посереддя' back to the start of the 'Використання доступних матеріалів' step if 'час не закінчився?' (time not finished?).

Рисунок В.8 – Слайд № 8

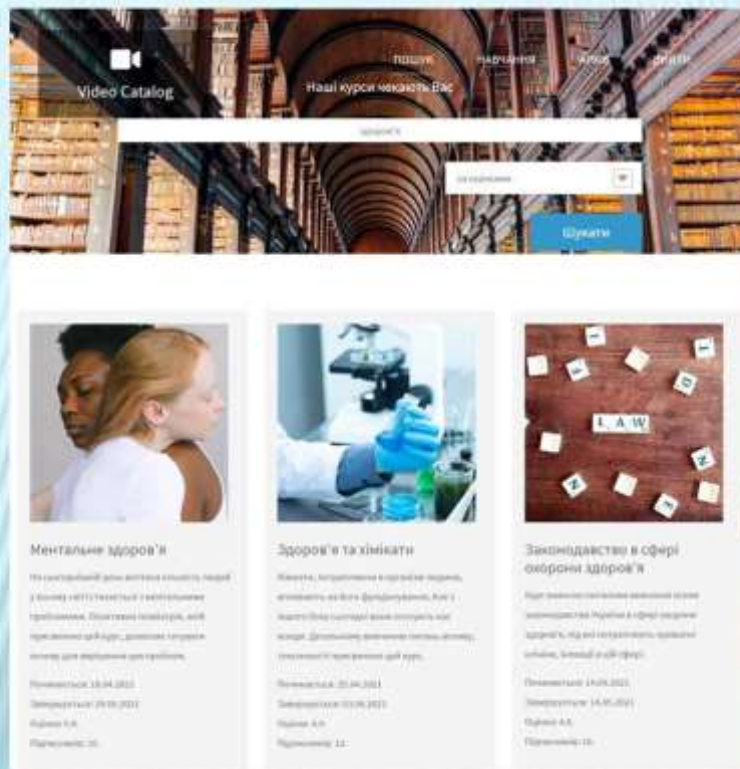
Діаграма класів представлень переліків курсів та тестів



9

Рисунок В.9 – Слайд № 9

Пошук нових курсів



10

Рисунок В.10 – Слайд № 10

Формування даних для персоналізації




11

Рисунок В.11 – Слайд № 11

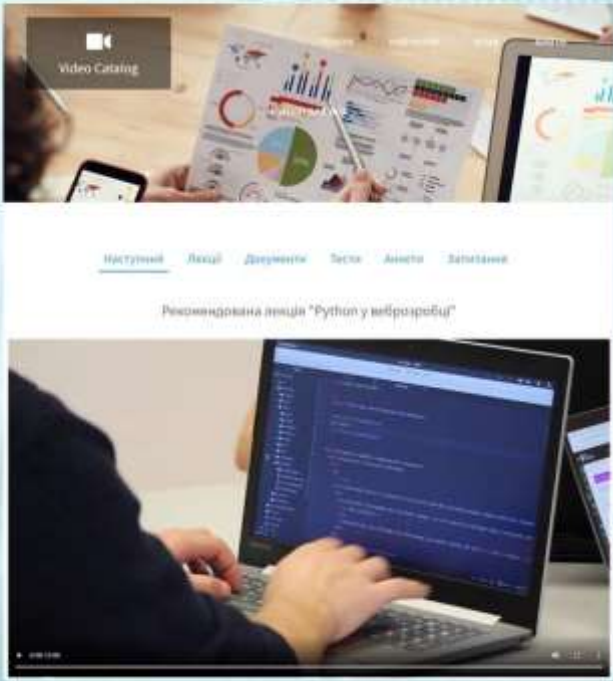
Персоналізація курсів

$t_2 (23 \ 27 \ 29) = 2 \rightarrow m_{14-}$

$t_1 (2 \ 4 \ 7) = 1,$
 $t_1 (9 \ 11 \ 12) = 0 \rightarrow m_{10+5},$
 m_{12+6}

$t_3 (47 \ 48 \ 49) = 8 \rightarrow m_{21-}$

$a_3 (21 \ 2, 22 \ 6, 23 \ 8) \rightarrow m_{1-}$



12

Рисунок В.12 – Слайд № 12

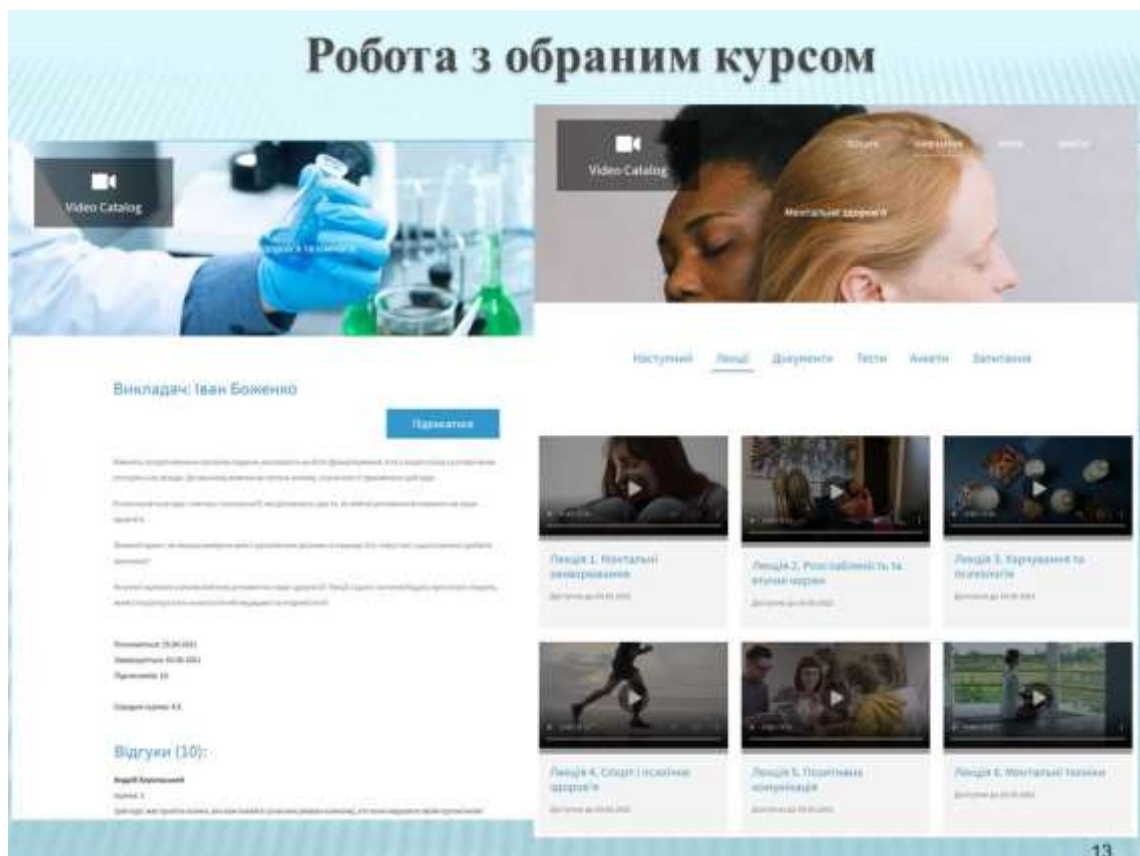


Рисунок В.13 – Слайд № 13

Висновки

Проблема онлайн навчання розглянута у вигляді створення онлайн курсів, які можуть застосовуватися на будь-якому етапі життя людини без необхідності формалізації освіти у одному з навчальних закладів, з персоналізованими матеріалами. Для розв'язання даної проблеми у вигляді розроблення відповідного програмного забезпечення було отримано наступні результати:

- проаналізовано особливості та причини популярності онлайн курсів, виконано огляд програмних аналогів, включаючи Coursera, Udemy та EdX;
- обрано мову програмування мову Python для реалізації програми на основі виконаного порівняння з мовою Java;
- визначено функціональні вимоги до програми з моделювання основних сценаріїв роботи учня та викладача з програмою у вигляді діаграм варіантів використання;
- визначено спосіб роботи з даними на основі БД з представленням структури цієї БД у вигляді відповідної схеми;
- реалізовано програмне забезпечення у вигляді вебзастосунку на основі фреймворку Django;
- описано розроблені класи та функції під час реалізації вебзастосунку.

Усі завдання дипломної кваліфікаційної роботи повністю виконано.

Рисунок В.14 – Слайд № 14