

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проекту (роботи)

бакалавра

(ступінь вищої освіти)

на тему ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ДЛЯ ДОПОМОГИ

(назва теми)

ПРИ НАВЧАННІ ПРОГРАМУВАННЮ. SOFTWARE TO

SUPPORT TRAINING IN PROGRAMMING

Виконав(ла): студент(ка) 4 курсу, групи КНТ-220

Спеціальності 122 Комп'ютерні науки

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Комп'ютерні науки

ПОХИЛА Н.Ю.

(ПРІЗВИЩЕ та ініціали)

Керівник ЛЬОВКІН В.М.

(ПРІЗВИЩЕ та ініціали)

Рецензент ГОЛУБ Т.В.

(ПРІЗВИЩЕ та ініціали)

2024

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет _____ КНТ _____
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. _____
Сергій СУББОТІН
« _____ » _____ 2024 року

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

ПОХИЛИ Назара Юрійовича

(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Програмне забезпечення для допомоги при навчанні програмуванню. Software to Support Training in Programming

керівник проєкту (роботи) к.т.н., доцент ЛЬОВКІН Валерій Миколайович _____,
(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від « 24 » квітня 2024 року № 168

2. Строк подання студентом проєкту (роботи) 10 червня 2024 року

3. Вихідні дані до проєкту (роботи) технічне завдання

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

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

Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	ПРІЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	ЛЬОВКІН В.М., доцент		
Нормоконтроль	ДЕЙНЕГА Л.Ю., ст. викладач		

7. Дата видачі завдання « 15 » квітня 2024 року.

КАЛЕНДАРНИЙ ПЛАН

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

Студент(ка)

(підпис) Назар ПОХИЛА
(Ім'я ПРІЗВИЩЕ)

Керівник проєкту (роботи)

(підпис) Валерій ЛЬОВКІН
(Ім'я ПРІЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 93 с., 27 рис., 3 табл., 3 дод., 9 джерел.

ОНЛАЙН-НАВЧАННЯ, ПРОБЛЕМНЕ НАВЧАННЯ, ПРОГРАМУВАННЯ, ЕКСПЕРТ, ВЕБСИСТЕМА.

Об'єкт роботи – процес навчання програмуванню. Предмет роботи – програмне забезпечення для допомоги при навчанні програмуванню. Мета роботи – розробити програмне забезпечення для допомоги при навчанні програмуванню з підтримкою можливості навчання на основі курсів та проблемного навчання.

Матеріали, методи та технічні засоби: мова Python, система MySQL, фреймворк Django, що є базовими засобами розробки.

Результати. Досягнуто успішних результатів за виконанням всіх завдань роботи, що сукупно дозволило розробити заплановане програмне забезпечення.

Висновки. Програма призначена для використання її експертами та учнями. Учнями програма може використовуватись для проходження курсів, підведення підсумків навчання та визначення власних проблемних питань, за якими потрібно виконати індивідуальне навчання, вибору потрібного варіанту серед тих, які запропонували експерти. Експертами програма може використовуватись для роботи над створенням власних курсів, на які може записуватися одразу багато учнів, та для формування власних пропозицій стосовно індивідуальної підготовки учнів, які потребують навчання за наявними проблемними питаннями (проблемного навчання).

Галузь використання. Навчання людей програмуванню за мовами і технологіями та проблемами, які виникають індивідуально та потребують індивідуальної підготовки.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 93 pages, 27 figures, 3 tables, 3 appendixes, 9 sources.

ONLINE LEARNING, PROBLEM LEARNING, PROGRAMMING, EXPERT, WEB SYSTEM.

The object of the work is the process of teaching programming. The subject of the work is software to help with learning programming. The goal of the work is to develop software to assist in teaching programming with support for course-based learning and problem-based learning.

Materials, methods and technical means: Python language, MySQL system, Django framework, which are basic development tools.

The results. Successful results were achieved in the performance of all work tasks, which collectively allowed to develop the planned software.

Conclusions. The program is aimed at usage by its experts and students. Students can use the program to complete courses, control their studies and identify their own problematic issues for which individual training should be carried out, choosing the right option among those offered by experts. Experts can use the program to work on creating their own courses for which many students can enroll at once, and to formulate their own proposals for individual training of students who need training on existing problematic issues (problematic training).

Field of use. Teaching people to program based on languages, technologies and problems that arise individually and require individual training.

ЗМІСТ

	С.
Перелік скорочень та умовних познач.....	7
Вступ.....	8
1 Аналіз предметної області.....	9
1.1 Огляд програм допомоги для навчання програмуванню.....	9
1.2 Порівняння аналогів для визначення характеристик власної розробки	11
1.3 Висновки за розділом 1	13
2 Проєктування програми.....	14
2.1 Вибір засобів проєктування і розробки	14
2.2 Моделювання взаємодії користувачів з програмою.....	14
2.3 Визначення структури бази даних	23
2.4 Висновки за розділом 2	34
3 Розробка програми	35
3.1 Визначення послідовності роботи з програмою	35
3.2 Особливості програмної реалізації.....	36
3.3 Висновки за розділом 3	47
4 Експлуатація та тестування програми	48
4.1 Призначення програми	48
4.2 Умови виконання програми	48
4.3 Експлуатація програми.....	49
4.4 Тестування програми.....	53
4.5 Висновки за розділом 4	56
Висновки	57
Перелік джерел посилання	58
Додаток А Технічне завдання	59
Додаток Б Текст програми	65
Додаток В Слайди презентації.....	86

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

ПЗ – програмне забезпечення;

ФВР – фреймворк веброзробки.

ВСТУП

На даний момент на ринку представлено декілька широко відомих платформ, включаючи Coursera, Udey та інші, які серед курсів пропонують підготовку і з програмування. Ці курси є авторськими, часом достатньо привабливими, однак всі вони мають типовий процес підготовки, який не кожному потенційному учню потрібен. Особливо в галузі програмування іноді потрібно забезпечити підготовку з окремих питань, які є більш спеціалізованими та потребують і спеціалізованих знань у експерта. У такому випадку наявні готові курси можуть не відповідати потребам учня, а це призводить до того, що потреби таких користувачів не задовольняються. Щоби вирішити цю проблему та забезпечити різносторонню підготовку, а за рахунок індивідуального (проблемного) навчання і привертання додаткової уваги до програми, необхідно розробити програмну систему, яка дозволить реалізувати підтримку навчання і за курсами з програмування, і за індивідуальною підготовкою.

Це дозволило сформулювати мету роботи як розробити програмне забезпечення (ПЗ) для допомоги при навчанні програмуванню з підтримкою можливості навчання на основі курсів та проблемного навчання.

Таким чином сформульована мета має бути реалізована через завдання дипломної кваліфікаційної роботи:

- проаналізувати проблему розробки програми для допомоги при навчанні програмуванню;
- спроектувати ПЗ для допомоги при навчанні програмуванню;
- реалізувати ПЗ для допомоги при навчанні програмуванню та виконати його тестування.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд програм допомоги для навчання програмуванню

Звичайно, що на даний момент багато платформ онлайн-навчання можуть розглядатися як засоби допомоги для навчання програмуванню, оскільки вони містять і такі курси серед інших. Тому з них і розпочато огляд.

Coursera [1] (рис. 1.1) є однією з найвідоміших на даний момент платформ для навчання, що надає як безкоштовні, так і платні курси, в тому числі з програмування.

The screenshot shows the Coursera website interface. At the top, there are navigation links for 'People', 'Companies', 'Universities', and 'Government'. Below this is the Coursera logo and a search bar. The main content area displays the 'Crash Course on Python' by Google. It includes a description of the course, the instructor 'Google Career Certificates', and a 'Sign up for free' button. A sidebar on the right provides additional details: a 4.8-star rating, a 'Beginner' level, a duration of 32 hours, and a 'Flexible schedule' option. At the bottom, there are tabs for 'Information', 'Results', 'Modules', 'Recommendations', 'Reviews', and 'Feedback'.

Рисунок 1.1 – Курс з програмування на платформі Coursera

Курси можна зберігати для того, щоб потім до них повернутись. На платні курси можна реєструватись для проходження. Для отримання результату проходження курсу проводиться перевірка знань, яка зазвичай має тестовою формою. Може бути представлено декілька тестів за одним курсом. Але визначити запит на навчання за тим, з чим є проблема в учня, неможливо.

Udemy [2] (рис. 1.2) має подібні засоби до Coursera, також надаючи можливості реєстрації на курсах, в тому числі з програмування, оформлюючи цей процес як фактично покупки в магазині. При цьому сплата відбувається за конкретний курс, на відміну від Coursera, де доступні також підписки для навчання.

The screenshot displays the Udemy website interface. At the top, there is a search bar and navigation links for 'Udemy Business', 'Teach on Udemy', 'Log in', and 'Sign up'. The main content area features the course 'The Complete Python Bootcamp From Zero to Hero in Python' by Jose Portilla. The course is rated 4.6 stars with 505,439 ratings and 1,905,333 students. It is priced at \$22.99, an 87% discount from the original price of \$174.99. The course includes 22 hours of on-demand video, 19 coding exercises, 15 articles, and full lifetime access. A 'What you'll learn' section lists various topics, including building games, creating a portfolio, and learning advanced Python features. The page also mentions a 30-day money-back guarantee and a coupon code 'LEADERSALE24B'.

Рисунок 1.2 – Курс з програмування на платформі Udemy

Особливістю Udemy порівняно з Coursera є також можливість створення курсів, у тому числі з програмування, будь-якими авторами. Це створює різноманіття, однак призводить до різного рівня якості, а відповідно і ефективності навчання, закладеної самим курсом. Coursera запрошує фахівців для створення курсів. Ці фахівці є зазвичай представниками відомих у світі університетів. Таким чином це сприяє якості.

Спеціалізованою платформою саме в навчанні програмуванню є Programming Hub [3] (рис. 1.3). Тут користувачі можуть переглядати курси з

програмування за категорією, при цьому пошук недоступний, але зважаючи на обмежену кількість курсів та програму в бета-версії це загалом є логічним.

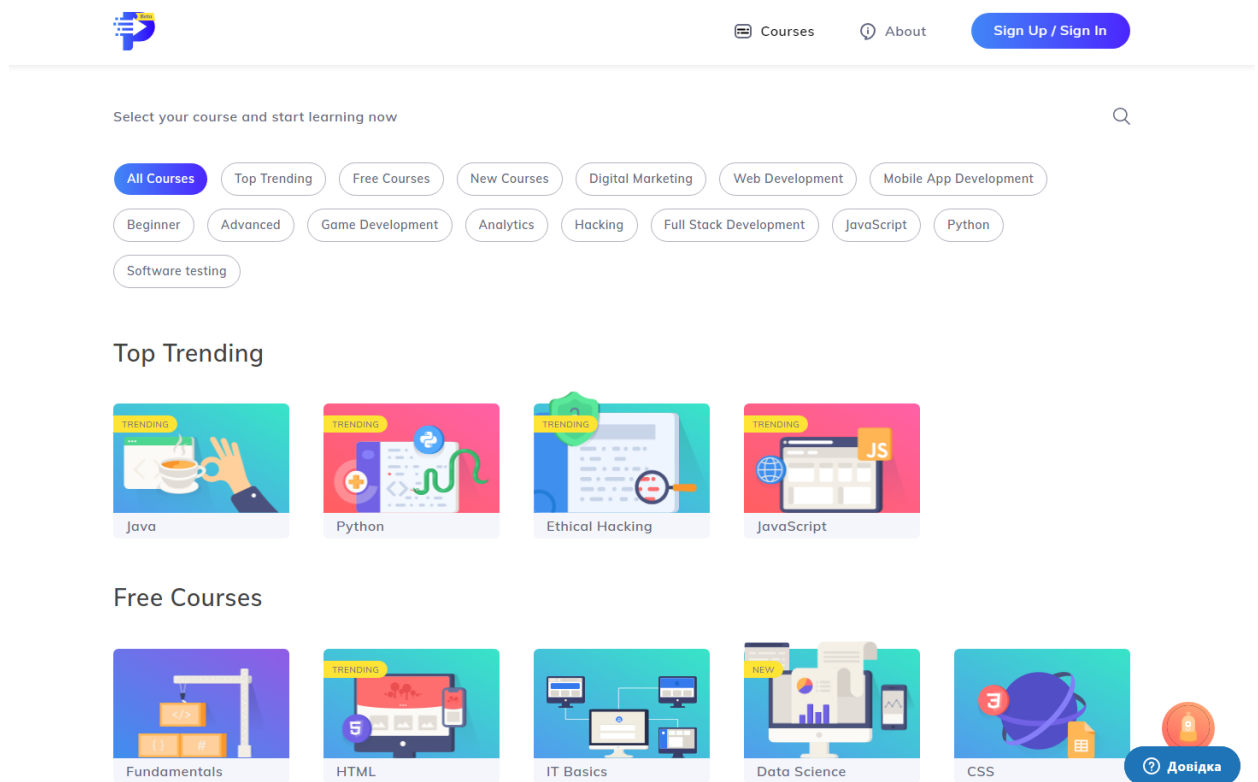


Рисунок 1.3 – Курси з програмування на платформі Programming Hub

Авторство курсів з програмування не визначається, до того ж і визначати власні проблеми, які потребують навчання, учні не можуть.

1.2 Порівняння аналогів для визначення характеристик власної розробки

Якщо порівнювати між собою Coursera, Udemy, Programming Hub та виконувану розробку (табл. 1.1), то слід визначити основну відмінність розробки як можливість пошуку експертів для навчання за проблемами учня. Окрім того розробка повинна містити пошук курсів з програмування, можливість визначення зацікавленості в курсі, щоб потім повернутися до

переліку таких курсів, можливість оцінювання курсів з програмування після проходження таких курсів.

Таблиця 1.1 – Порівняння існуючих аналогів та розроблюваного ПЗ для допомоги при навчанні програмуванню

Характеристика	Coursera	Udemy	Program- ming Hub	Викону- вана розробка
Навчання за готовими курсами з програмування	+	+	+	+
Пошук курсів з програмування	+	+	—	+
Потенційні автори курсів	Обмежене коло	Без обмежень	Обмежене коло	Без обмежень, але з наданням права
Визначення зацікавленості в курсі	+	+	—	+
Оцінювання курсів з програмування	+	+	—	+
Пошук експертів для навчання за проблемами учня	—	—	—	+
Оцінювання експертів	—	—	—	+

Потенційними авторами курсів може бути будь-який користувач, який є експертом, однак безпосередньо створювати курс він може тільки після надання відповідного статусу модератором. Це може відбуватися після отримання позитивних оцінок від учнів, для яких він виконував проблемне навчання.

1.3 Висновки за розділом 1

Розділ містить результати огляду загальноцільових платформ для онлайн-навчання, включаючи Coursera та Udey, платформи для навчання програмування Programming Hub. Підсумком огляду є винесення основних потрібних можливостей програми для реалізації допомоги при навчання програмуванню та формування особливостей програми, яка створюється. Основною відмінністю розробки є можливість пошуку експертів для навчання за проблемами учня.

2 ПРОЄКТУВАННЯ ПРОГРАМИ

2.1 Вибір засобів проєктування і розробки

Мова Python є основною для реалізації ПЗ для допомоги при навчанні програмуванню, враховуючи її гнучкість, простоту, наявність широкої підтримки, готових засобів. Фреймворк веброзробки (ФВР) для Python обирався серед Django [4]-[5] і TurboGears [6]-[8] (табл. 2.1). Django є монолітним рішенням, яке включає в себе фактично всі потрібні для розробки потужного вебдодатку складові. Це може бути проблемою для деяких проєктів, проте вебдодаток у цій роботі визначається і жорстко обмеженими строками розробки, і характеризується потужністю, тому монолітний підхід є важливою перевагою. Тому обрано було як ФВР Django, враховуючи також те, що обидва ФВР дозволяють працювати з потужними системами взаємодії з базами даних.

Таблиця 2.1 – Порівняння ФВР для створення ПЗ для допомоги при навчанні програмуванню

Критерій порівняння	Django	TurboGears
Підхід до забезпечення роботи з компонентами	Монолітний	Модульний
Підхід до розробки	Все включено	Легкий, але придатний до розширення
Підтримка баз даних	Наявна зокрема для PostgreSQL, MySQL, SQLite, Oracle	Використовує SQLAlchemy, а через нього можлива робота з різними базами

2.2 Моделювання взаємодії користувачів з програмою

Моделювання сценаріїв роботи з ПЗ навчання програмуванню для учня включає як основні:

- пошук потрібних курсів з програмування;
- перегляд власних проблем, за якими потрібне навчання;
- перегляд пропозицій від експертів на власну проблему;
- перегляд переліку пропозицій на всі власні проблеми;
- реєстрація на вивчення курсів з програмування;
- визначення зацікавлення учня у вивченні курсу з програмування;
- перегляд курсів з програмування, у вивченні яких зацікавлений учень;
- перегляд переліку курсів з програмування, на вивчення яких зареєструвався учень;
- перегляд тем за курсом з програмування;
- перегляд курсів з програмування за заданою мовою програмування;
- перегляд курсів з програмування за заданою технологією;
- перегляд курсів з програмування за експертом;
- перегляд пропозицій експерта з навчання за проблемами в галузі програмування;
- перегляд навчань з програмування, на які зареєструвався учень;
- перегляд матеріалу за курсом з програмування або навчанням за проблемою;
- проходження тесту для перевірки засвоєння курсу з програмування;
- підтвердження пропозиції про навчання за проблемою;
- авторизація;
- перегляд сторінки експерта;
- перегляд сторінки курсу з програмування;
- внесення проблеми в галузі програмування, за якою є потреба в навчанні;
- перегляд виставлених оцінок за курс з програмування;
- перегляд виставлених експерту оцінок за результатами індивідуального навчання (рис. 2.1).

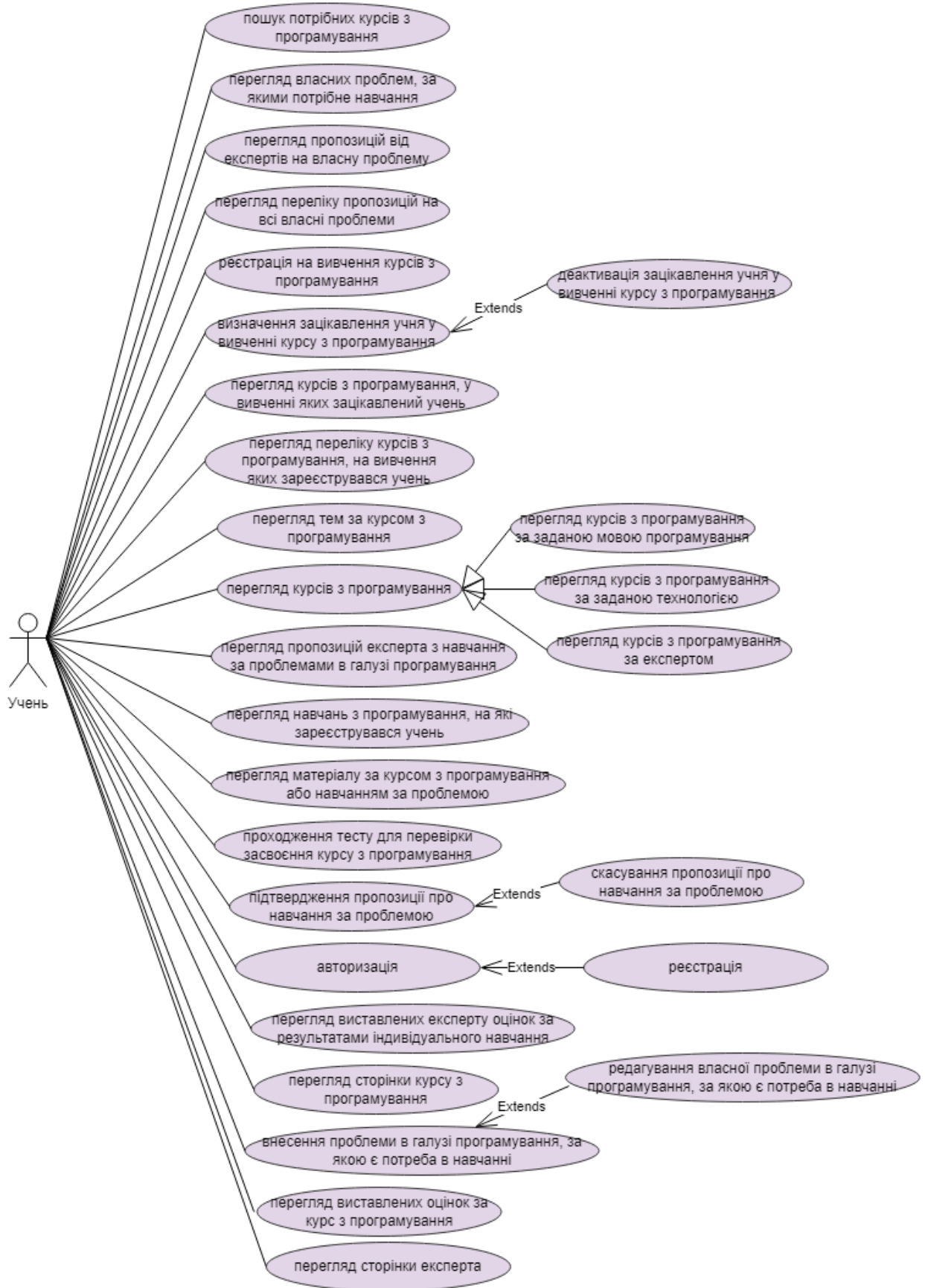


Рисунок 2.1 – Діаграма варіантів використання ПЗ для допомоги при навчанні програмуванню учнем

Але серед сценаріїв учня є альтернативні:

- деактивація зацікавлення учня у вивченні курсу з програмування до визначення зацікавлення учня у вивченні курсу з програмування, коли зацікавлення створене, але не є актуальним;
- скасування пропозиції про навчання за проблемою до підтвердження пропозиції про навчання за проблемою, коли пропозиція не зацікавила учня;
- реєстрація до авторизації, коли учень не зареєструвався раніше;
- редагування власної проблеми в галузі програмування, за якою є потреба в навчанні, до внесення проблеми в галузі програмування, за якою є потреба в навчанні, коли проблема вже внесена, проте інформація про проблему змінилась.

Моделювання сценаріїв роботи з ПЗ навчання програмуванню для експерта включає як основні:

- пошук потрібних курсів з програмування;
- пошук серед проблем учнів, за якими потрібно провести навчання;
- перегляд проблем учня, за якими він потребує навчання;
- перегляд пропозицій від експертів на проблему учня;
- перегляд курсів з програмування за заданою мовою програмування;
- перегляд курсів з програмування за заданою технологією;
- перегляд курсів з програмування за експертом;
- перегляд пропозицій експерта з навчання за проблемами в галузі програмування;
- перегляд власного матеріалу за навчанням за проблемою;
- додавання матеріалу до навчання за проблемою;
- створення пропозиції про навчання за проблемою;
- авторизація;
- перегляд сторінки експерта;
- перегляд сторінки учня;
- перегляд сторінки курсу з програмування;
- перегляд виставлених оцінок за курс з програмування;

– перегляд виставлених експерту оцінок за результатами індивідуального навчання (рис. 2.2).

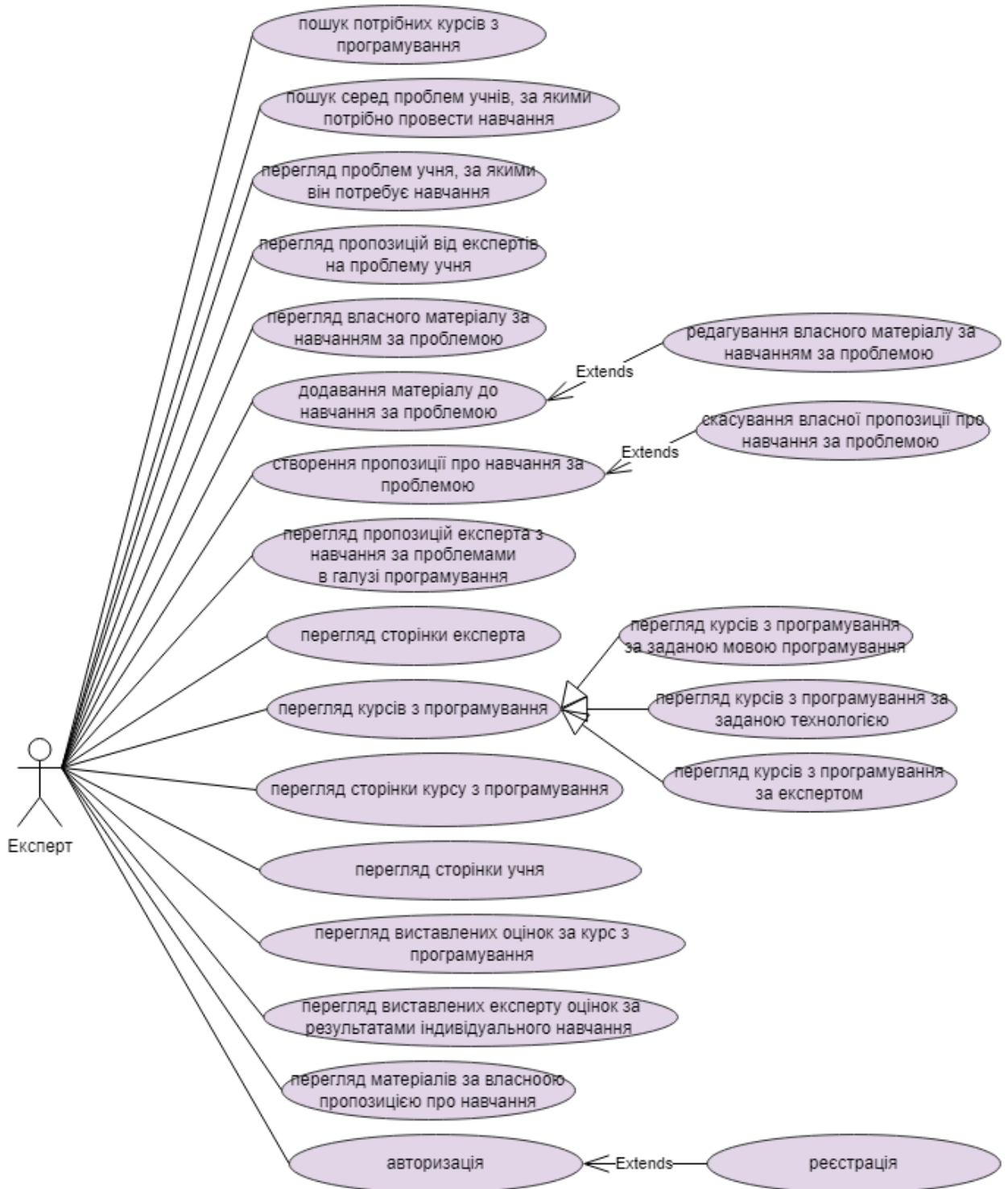


Рисунок 2.2 – Діаграма варіантів використання ПЗ для допомоги при навчанні програмуванню експертом

Але серед сценаріїв експерта є альтернативні:

- редагування власного матеріалу за навчанням за проблемою до додавання матеріалу до навчання за проблемою, коли матеріал був цим самим експертом раніше доданий;

- скасування власної пропозиції про навчання за проблемою до створення пропозиції про навчання за проблемою, коли пропозиція була створена, але експерт вже не має можливості її виконати;

- авторизація до реєстрації, коли експерт не зареєстрований в програмі.

Моделювання сценаріїв роботи з ПЗ навчання програмуванню для авторизованого автора включає як основні сценарії експерта та наступні:

- перегляд тем за власним курсом з програмування;
- перегляд учнів зареєстрованих на власний курс з програмування;
- перегляд власного матеріалу за курсом з програмування;
- створення курсу з програмування;
- додавання матеріалу до курсу з програмування;
- перегляд запитань для перевірки знань за курсом з програмування;
- створення запитань для перевірки знань за курсом з програмування;
- додавання співавтора до власного курсу з програмування.

Але серед сценаріїв авторизованого автора є альтернативні:

- редагування даних курсу з програмування до створення курсу з програмування, коли курс з програмування був цим же авторизованим автором створений, а потрібно внести зміни;

- редагування матеріалу курсу з програмування до додавання матеріалу до курсу з програмування, коли матеріал курсу з програмування був раніше доданий цим же авторизованим автором, але потрібно внести зміни;

- редагування запитання для перевірки знань за курсом з програмування до створення запитань для перевірки знань за курсом з програмування, коли створення запитання реалізовано, але дані неточні;

- вилучення запитання для перевірки знань за курсом з програмування до створення запитань для перевірки знань за курсом з програмування, коли створення запитання реалізовано, але запитання зайве;

– редагування відповіді на запитання для перевірки знань за курсом з програмування до створення запитань для перевірки знань за курсом з програмування, коли створення запитання реалізовано, але одна з відповідей на запитання є неточною;

– вилучення відповіді на запитання для перевірки знань за курсом з програмування до створення запитань для перевірки знань за курсом з програмування, коли створення запитання реалізовано, але одна з відповідей на запитання є зайвою;

– вилучення співавтора з власного курсу з програмування до додавання співавтора до власного курсу з програмування, коли співавторство не відповідає дійсності;

– вилучення курсу з програмування до створення курсу з програмування, коли курс з програмування був цим же авторизованим автором створений, але більше за ним не має виконуватися навчання (рис. 2.3).

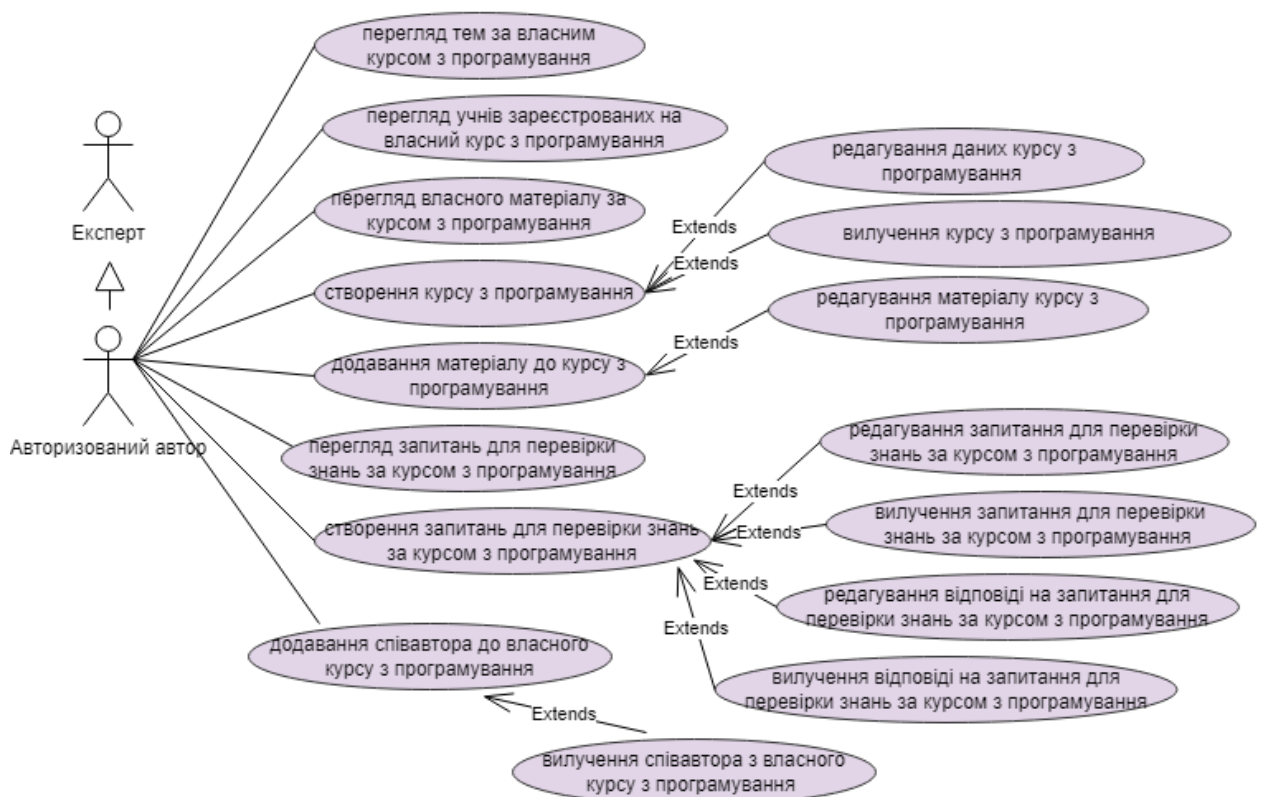


Рисунок 2.3 – Діаграма варіантів використання ПЗ для допомоги при навчанні програмуванню авторизованим автором

Моделювання сценаріїв роботи з ПЗ навчання програмуванню для модератора включає як основні:

- пошук експертів з навчання програмуванню;
- визнання експерта авторизованим автором;
- пошук потрібних курсів з програмування;
- пошук серед проблем учнів, за якими потрібно провести навчання;
- перегляд проблем учня, за якими він потребує навчання;
- перегляд пропозицій від експертів на проблему учня;
- перегляд курсів з програмування за заданою мовою програмування;
- перегляд курсів з програмування за заданою технологією;
- перегляд курсів з програмування за експертом;
- перегляд пропозицій експерта з навчання за проблемами в галузі програмування;
- перегляд матеріалу за навчанням за проблемою;
- авторизація;
- перегляд сторінки експерта;
- перегляд сторінки учня;
- перегляд сторінки курсу з програмування;
- перегляд виставлених оцінок за курс з програмування;
- перегляд виставлених експерту оцінок за результатами індивідуального навчання;
- перегляд тем за курсом з програмування;
- перегляд учнів зареєстрованих на курс з програмування;
- перегляд матеріалу за курсом з програмування;
- перегляд запитань для перевірки знань за курсом з програмування;
- перегляд тем за курсом з програмування (рис. 2.4).

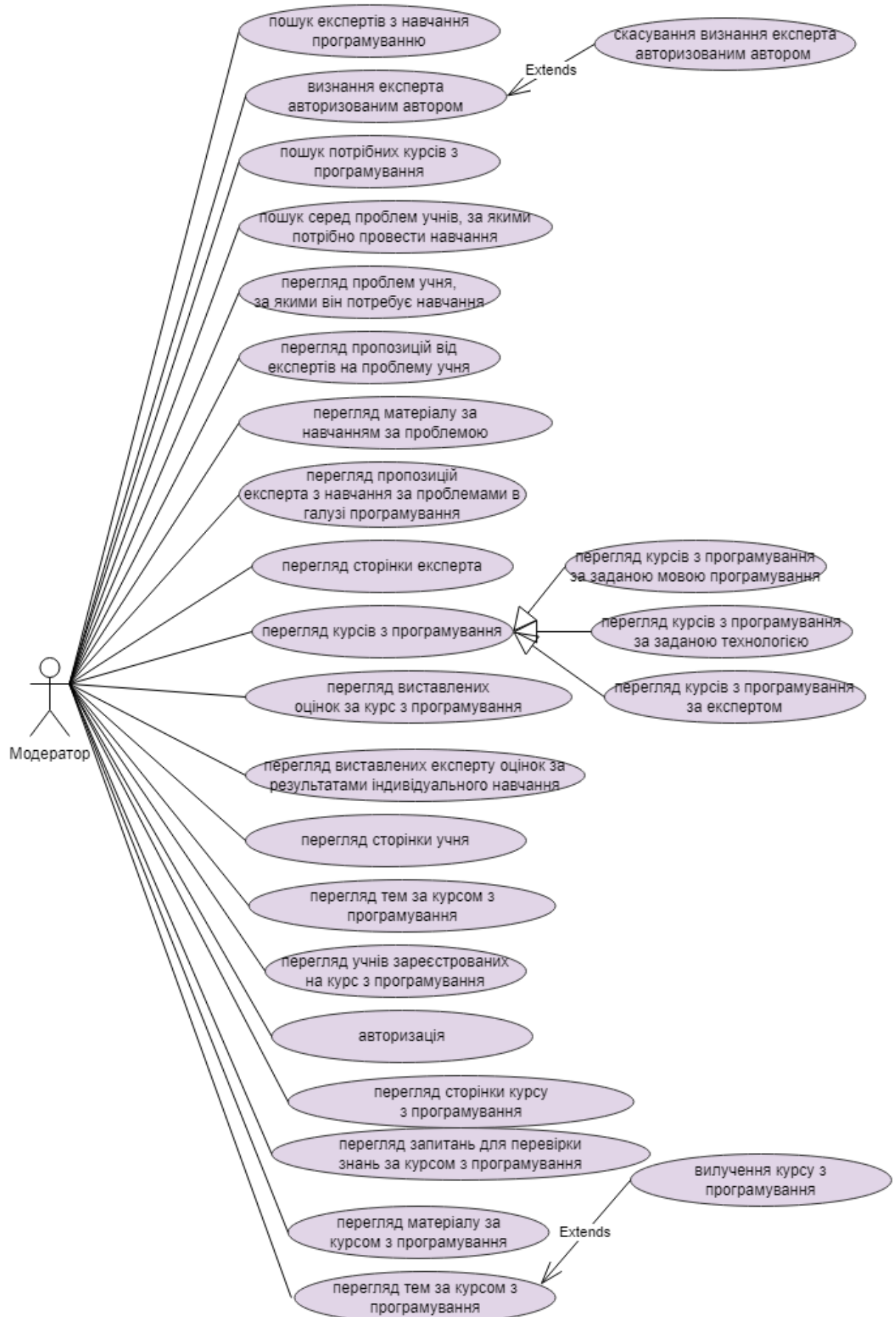


Рисунок 2.4 – Діаграма варіантів використання ПЗ для допомоги при навчанні програмуванню модератором

Але серед сценаріїв модератора є альтернативні:

- скасування визнання експерта авторизованим автором до визнання експерта авторизованим автором, коли експерт був визнаний авторизованим автором, але це вже не є справедливим;
- вилучення курсу з програмування до перегляду тем за курсом з програмування, коли курс з програмування не потрібен більше, не відповідає певним правилам.

2.3 Визначення структури бази даних

Структура таблиці Programlanguage відповідає мовам програмування за полями:

- id – числовий ідентифікатор мови програмування;
- lang_name – назва мови програмування.

Структура таблиці Programtechnology відповідає технологіям, які можуть використовуватися під час розробки програмного забезпечення, за полями:

- id – числовий ідентифікатор технології програмування;
- tech_name – назва технології, що може використовуватися під час розробки програмного забезпечення.

Структура таблиці Programcourse відповідає курсам, які можуть використовуватися для навчання програмуванню, за полями:

- id – числовий ідентифікатор курсу з програмування;
- short_caption – коротка назва курсу з програмування, що призначена для відображення в результатах пошуку зокрема;
- long_title – повна назва курсу з програмування;
- requirements – початкові вимоги, яким має відповідати учень, щоби він міг навчатися за цим курсом з програмування;
- poster – фотографія для курсу з програмування;

- result_skills – вміння, які отримає учень за результатами вивчення даного курсу з програмування;
- add_description – деталізований опис курсу з програмування, яким може містити і розмітку для форматування тексту;
- assessed_time – очікуваний час, який необхідно витрати для оволодіння всім даним курсом з програмування;
- knowledge_level – числовий рівень знань, на якому викладається даний курс з програмування;
- learningprice – вартість навчання за даним курсом з програмування;
- mark – середня оцінка, яку було виставлено даному курсу з програмування учнями, які зареєструвались на його вивчення;
- repeat – максимальна допустима кількість повторних разів проходження підсумкового тесту з даного курсу з програмування, якщо учень не зміг скласти тест на мінімальний допустимий бал з першого разу;
- min_level – мінімальний допустимий бал, на який потрібно здати тест за даним курсом з програмування, щоб успішно завершити навчання за ним.

Курси фактично визначають певний стандартний потік навчання програмуванню, створюються відповідальними авторами, яким надано таке право на рівні програмного забезпечення. Кожен такий курс передбачає проходження тесту для перевірки знань, що в підсумку вказує на успішність навчання. На такі курси мають записуватися для навчання учні. Курси можуть бути платними, а можуть бути і безкоштовними. Такий варіант навчання програмуванню забезпечується таблицею Programcourse.

У той же час учні можуть потребувати індивідуальної підготовки за спеціалізованими темами, які не є типовими. У такому випадку вони фіксують проблеми, за якими необхідно виконати навчання в галузі програмування, а експерти, які можуть бути і не відповідальними авторами, надають свої пропозиції щодо реалізації навчання. Такі пропозиції є платними. Але в такому випадку учень вирішує, чи погоджуватися на таку пропозицію. Такий варіант

навчання програмуванню забезпечується таблицею Programproblem, що описана нижче.

Структура таблиці Programcoursetopic відповідає темам курсів, які можуть використовуватися для навчання програмуванню, за полями:

- id – числовий ідентифікатор теми курсу з програмування;
- programcourse – курс з програмування, за яким вивчається дана тема (визначається зовнішнім ключем);
- short_caption – коротка назва теми курсу з програмування;
- topic_name – повна назва теми курсу з програмування;
- sequence_number – порядковий номер, за яким має вивчатися ця тема в даному курсі з програмування;
- add_description – повний опис цієї теми курсу з програмування.

Структура таблиці Programcoursestuff відповідає матеріалам вивчення курсів з програмування за полями:

- id – числовий ідентифікатор матеріалу, що вивчається за темою для навчання програмуванню за курсом;
- programcoursetopic – тема, за якою використовується цей матеріал для навчання програмуванню (визначається зовнішнім ключем);
- stuff_name – назва матеріалу, що вивчається за темою для навчання програмуванню за курсом;
- sequence_number – порядковий номер, за яким має вивчатися даний матеріал в даній темі для навчання програмуванню за курсом;
- assessed_time – очікуваний час, який необхідно витратити для повного вивчення цього матеріалу;
- add_description – додатковий опис даного матеріалу для навчання програмуванню за курсом.

Кожен такий матеріал може мати або текстове представлення, або представлятися посиланням на відеоматеріал. Оскільки вміст матеріалу може бути або першим, або другим, але не двома одночасно, то такий вміст було виділено у наступні таблиці, кожна з яких пов'язана зв'язком один-до-одного

з таблицею матеріалів, що використовуються в курсах для навчання програмуванню. Тоді кожен запис таблиці матеріалів має один з зв'язок з одним з рядків однієї з цих двох таблиць та не має зв'язку з одним з рядків з другої з цих таблиць.

Структура таблиці `Programcoursestuffhtml` відповідає форматovanому (розміченому) тексту матеріалу для навчання програмуванню за полями:

- `id` – числовий ідентифікатор форматovanого (розміченого) тексту матеріалу, що вивчається за темою для навчання програмуванню за курсом;
- `programcoursestuff` – матеріал, що вивчається за темою для навчання програмуванню за курсом, який визначається цим форматovanим (розміченим) текстом (визначається зовнішнім ключем);
- `clear_html` – текст матеріалу, який може містити елементи розмітки для представлення його у вигляді сторінки.

Структура таблиці `Programcoursestufffile` відповідає відеоматеріалу для навчання програмуванню за полями:

- `id` – числовий ідентифікатор відеоматеріалу для навчання програмуванню;
- `programcoursestuff` – матеріал, що вивчається за темою для навчання програмуванню за курсом, який визначається цим відеоматеріалом за посиланням (визначається зовнішнім ключем);
- `fileurl` – посилання на файл для відображення відео;
- `description` – опис відеоматеріалу.

Структура таблиці `Programcoursetechnology` відповідає технологіям програмування, вивчення яких забезпечується за певним курсом з програмування, реалізуючи зв'язок багато-до-багатьох між технологіями програмування та курсами з програмування на основі полів:

- `id` – числовий ідентифікатор технології програмування стосовно курсу з програмування;
- `programcourse` – курс з програмування, в якому вивчається ця технологія програмування (визначається зовнішнім ключем);

– `programtechnology` – технологія програмування, яка вивчається за цим курсом.

Структура таблиці `Programcourselanguage` відповідає мовам програмування, вивчення яких забезпечується за певним курсом з програмування, реалізуючи зв'язок багато-до-багатьох між мовами програмування та курсами з програмування на основі полів:

– `id` – числовий ідентифікатор мови програмування стосовно курсу з програмування;

– `programcourse` – курс з програмування, в якому вивчається ця технологія програмування (визначається зовнішнім ключем);

– `programlanguage` – мова програмування, яка вивчається за цим курсом (визначається зовнішнім ключем).

Структура таблиці `Programcoursetutor` відповідає відповідальним авторам створених ними курсів з програмування за полями:

– `id` – числовий ідентифікатор відповідального автора щодо створеного ним курсу з програмування;

– `programcourse` – курс з програмування, який був створений цим автором (визначається зовнішнім ключем);

– `tutor` – відповідальний автор, який створив цей курс з програмування (визначається зовнішнім ключем).

Структура таблиці `Tutor` відповідає експертам (відповідальним авторам курсів) за полями:

– `id` – числовий ідентифікатор експерта, який здійснює навчання програмуванню на платформі;

– `officialname` – офіційне повне ім'я експерта, який здійснює навчання програмуванню;

– `poster` – фотографія експерта;

– `username` – логін експерта, який здійснює навчання програмуванню;

– `password` – пароль експерта, який здійснює навчання програмуванню;

- `officialstatus` – статус експерта, який здійснює навчання програмуванню: відповідальний автор курсів на платформі або експерт, який тільки може пропонувати свою допомогу щодо проблем, які визначив для себе учень;

- `specialization` – особливості спеціалізації експерта, який здійснює навчання програмуванню;

- `mark` – середня оцінка, яку було виставлено даному експерту за результатами навчання за проблемами учнів;

- `experience` – досвід експерта, який здійснює навчання програмуванню.

Структура таблиці `Learner` відповідає учню, який навчається програмуванню на платформі, за полями:

- `id` – числовий ідентифікатор учня, який навчається програмуванню на платформі;

- `officialname` – офіційне повне ім'я учня, який навчається програмуванню на платформі;

- `username` – логін учня, який навчається програмуванню на платформі;

- `password` – пароль учня, який навчається програмуванню на платформі;

- `poster` – фотографія учня.

Структура таблиці `Programcourselicense` відповідає реєстрації учня на вивчення курсу з програмування за полями:

- `id` – числовий ідентифікатор реєстрації учня на вивчення курсу з програмування;

- `programcourse` – курс з програмування, на вивчення якого зареєструвався учень (визначається зовнішнім ключем);

- `learner` – учень, який зареєструвався на вивчення даного курсу з програмування (визначається зовнішнім ключем);

- `license_date` – дата, коли учень зареєструвався на вивчення курсу з програмування;

– final_date – дата, коли учень закінчив вивчення курсу з програмування;

– paidamount – сума, яку сплатив за реєстрацію на вивчення даного курсу з програмування учень;

– final_result – підсумкова оцінка за результатами тестування, яку за цим курсом з програмування отримав учень;

– mark – оцінка, яку даному курсу з програмування виставив учень;

– repeat – спроба, з якої було отримано дану оцінку.

Структура таблиці Programcoursestufflearning відповідає відмітці про вивчення певного матеріалу курсу з програмування за полями:

– id – числовий ідентифікатор відмітки про вивчення певного матеріалу курсу з програмування;

– programcoursestuff – матеріал курсу з програмування, який вивчений учнем (визначається зовнішнім ключем);

– learner – учень, який навчається програмуванню на платформі та який оволодів цим матеріалом курсу з програмування (визначається зовнішнім ключем);

– learning_time – час, коли ця подія була зафіксована.

Структура таблиці Programcourseinterest відповідає зацікавленню учнів у певних курсах з програмування, на які вони ще не зареєструвалися, за полями:

– id – ідентифікатор зацікавлення учня у певному курсі з програмування, на який він ще не зареєструвався;

– programcourse – курс з програмування, в якому зацікавлений учень (визначається зовнішнім ключем);

– learner – учень, який зацікавлений у цьому курсі з програмування (визначається зовнішнім ключем).

Структура таблиці Programproblem відповідає проблемі, за якою учню необхідно виконати навчання в галузі програмування, з полями:

- `id` – ідентифікатор проблеми, за якою учню необхідно виконати навчання в галузі програмування;
- `learner` – учень, якому необхідно виконати навчання за певною проблемою в галузі програмування (визначається зовнішнім ключем);
- `problem_description` – опис проблеми в галузі програмування, за якою учень потребує навчання;
- `activeness` – відмітка, необхідна для визначення того, чи ця проблема ще є актуальною, чи учень вже оволодів цією проблемою певним чином;
- `create_date` – час внесення даних про проблему, за якою учню необхідно виконати навчання в галузі програмування;
- `max_price` – вартість навчання, яку максимально за цією проблемою готовий сплатити учень.

Структура таблиці `Programproblemoffer` відповідає пропозиції, які експерти надають стосовно навчання за проблемами в галузі програмування, які виникають у учнів, за полями:

- `id` – ідентифікатор пропозиції, яку експерт надає стосовно навчання за проблемою в галузі програмування, яка виникла у учня;
- `tutor` – експерт, який готовий виконати навчання учня за вказаною проблемою в галузі програмування (визначається зовнішнім ключем);
- `programproblem` – проблема, за якою учню необхідно виконати навчання в галузі програмування (визначається зовнішнім ключем);
- `offer_description` – особливості виконання навчання в галузі програмування за визначеною учнем проблемою в галузі програмування, які готовий надати експерт за власною пропозицією;
- `create_date` – час, коли створена пропозиція, яку експерт надає стосовно навчання за проблемою в галузі програмування, яка виникла у учня;
- `status` – стан, у якому перебуває пропозиція, яку експерт надає стосовно навчання за проблемою в галузі програмування, яка виникла у учня;
- `price_offer` – вартість, за яку експерт готовий навчати учня за сформульованою ним проблемою в галузі програмування;

– approve_date – час, коли пропозиція експерта стосовно навчання за проблемою в галузі програмування, яка виникла у учня, була погоджена або скасована учнем;

– mark – оцінка, виставлена експерту за організоване проблемне навчання учнем.

Структура таблиці Programproblemstuff відповідає матеріалу, який застосовується експертом для навчання учня за проблемою в галузі програмування, яка є у учня, за полями:

– id – ідентифікатор матеріалу, який застосовується експертом для навчання учня за проблемою в галузі програмування, яка є в учня;

– programproblemoffer – пропозиція, яку експерт надає стосовно навчання за проблемою в галузі програмування, що виникла у учня, та за якою надається цей матеріал для навчання;

– stuff_name – назва матеріалу, який застосовується експертом для навчання учня за проблемою в галузі програмування, яка є в учня;

– sequence_number – порядковий номер, за яким має вивчатися даний матеріал, що застосовується експертом для навчання учня за проблемою в галузі програмування, яка є в учня;

– add_description – додатковий опис даного матеріалу, що застосовується експертом для навчання учня за проблемою в галузі програмування, яка є в учня.

Структура таблиці Programproblemstuffhtml відповідає форматovanому (розміченому) тексту матеріалу для навчання програмуванню за полями:

– id – числовий ідентифікатор форматovanого (розміченого) тексту матеріалу, який застосовується експертом для навчання учня за проблемою в галузі програмування, яка є в учня;

– programproblemstuff – матеріал, який застосовується експертом для навчання учня за проблемою в галузі програмування, яка є в учня, до якого належить цей форматований (розмічений) текст (визначається зовнішнім ключем);

– `clear_html` – текст матеріалу, який застосовується експертом для навчання учня за проблемою в галузі програмування, яка є в учня і який може містити елементи розмітки для представлення його у вигляді сторінки.

Структура таблиці `Programproblemstufffile` відповідає відеоматеріалу для навчання за проблемою учня в галузі програмування за полями:

– `id` – числовий ідентифікатор відеоматеріалу для навчання програмуванню;

– `programproblemstuff` – матеріал, який застосовується експертом для навчання учня за проблемою в галузі програмування, яка є в учня і який визначається цим відеоматеріалом за посиланням (визначається зовнішнім ключем);

– `fileurl` – посилання на файл для відображення відео, що застосовується експертом для навчання учня за проблемою в галузі програмування;

– `description` – опис відеоматеріалу, який застосовується експертом для навчання учня за проблемою в галузі програмування.

Структура таблиці `Programcoursetest` відповідає тестовому запитанню для перевірки засвоєння курсу з програмування за полями:

– `id` – числовий ідентифікатор тестового запитання для перевірки засвоєння курсу з програмування;

– `programcourse` – курс з програмування, за яким ставиться тестове запитання для перевірки засвоєння знань (визначається зовнішнім ключем);

– `test_question` – текст запитання для перевірки засвоєння знань за курсом з програмування;

– `sequence_number` – порядковий номер даного тестового запитання серед всіх запитань, які використовуються для перевірки засвоєння курсу з програмування;

– `points` – кількість балів, яку можна отримати за правильну відповідь на це тестове запитання (рис. 2.5).

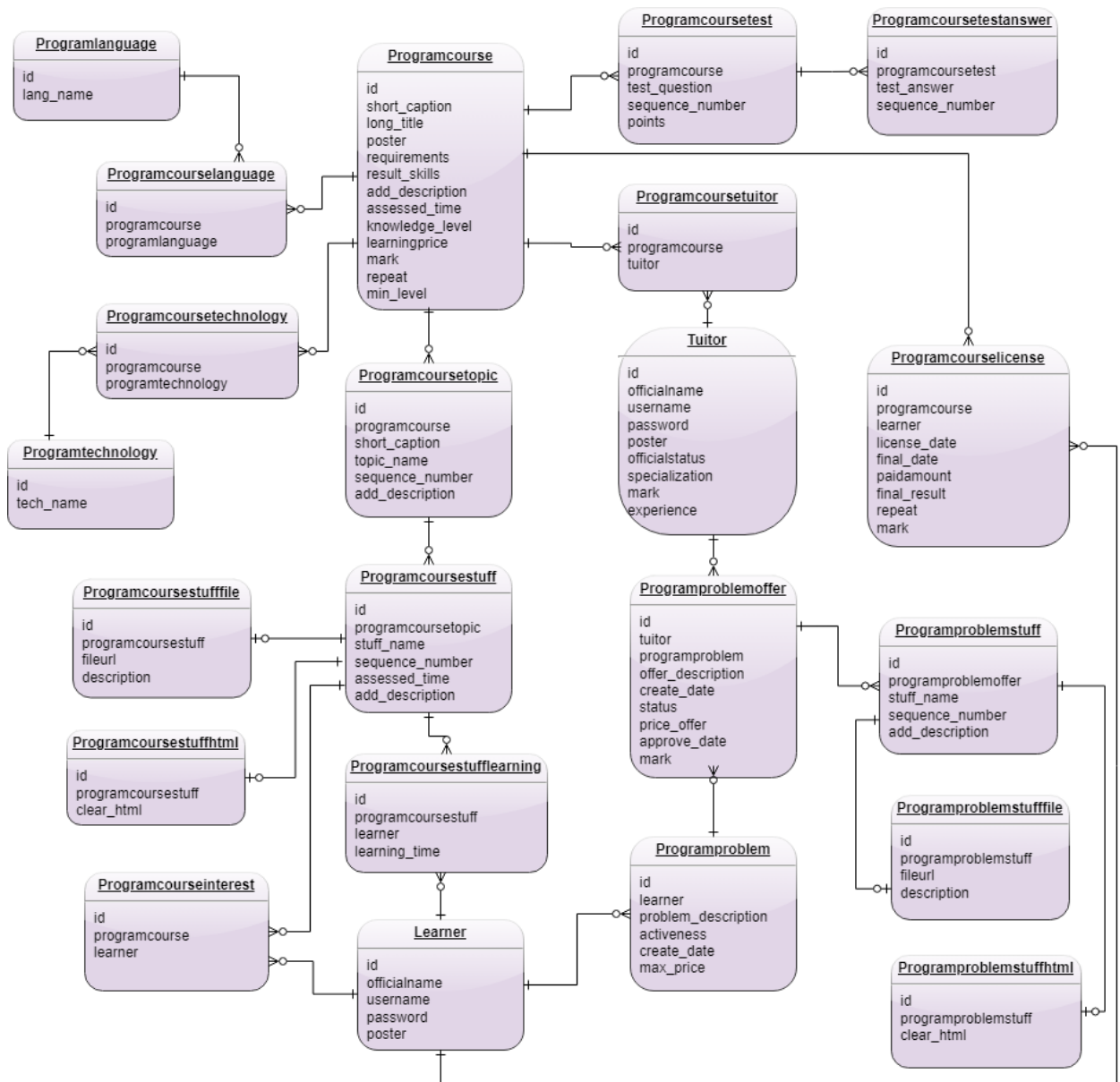


Рисунок 2.5 – Структура бази даних ПЗ для допомоги при навчанні програмуванню

Структура таблиці Programcoursetestanswer визначає відповідь на тестове запитання для перевірки засвоєння курсу з програмування за полями:

- id – числовий ідентифікатор відповіді на тестове запитання для перевірки засвоєння курсу з програмування;
- programcoursetest – тестове запитання для перевірки засвоєння курсу з програмування, до якого належить ця відповідь (визначається зовнішнім ключем);

– test_answer – текст відповіді на тестове запитання для перевірки засвоєння курсу з програмування;

– sequence_number – порядковий номер даної відповіді серед відповідей на це тестове запитання для перевірки засвоєння курсу з програмування.

2.4 Висновки за розділом 2

Визначені сценарії роботи користувачів, включаючи експертів, модераторів та учнів, структура бази даних ПЗ для допомоги при навчанні програмуванню побудовані на основі об'єднання можливостей роботи зі стандартними курсами, які розробляються авторизованими у програмі авторами та на які реєструються учні, та навпаки визначення проблемних питань у галузі програмування учнями, на які власні пропозиції щодо навчання висувають експерти.

3 РОЗРОБКА ПРОГРАМИ

3.1 Визначення послідовності роботи з програмою

Послідовність роботи з програмою визначена схемою (рис. 3.1).

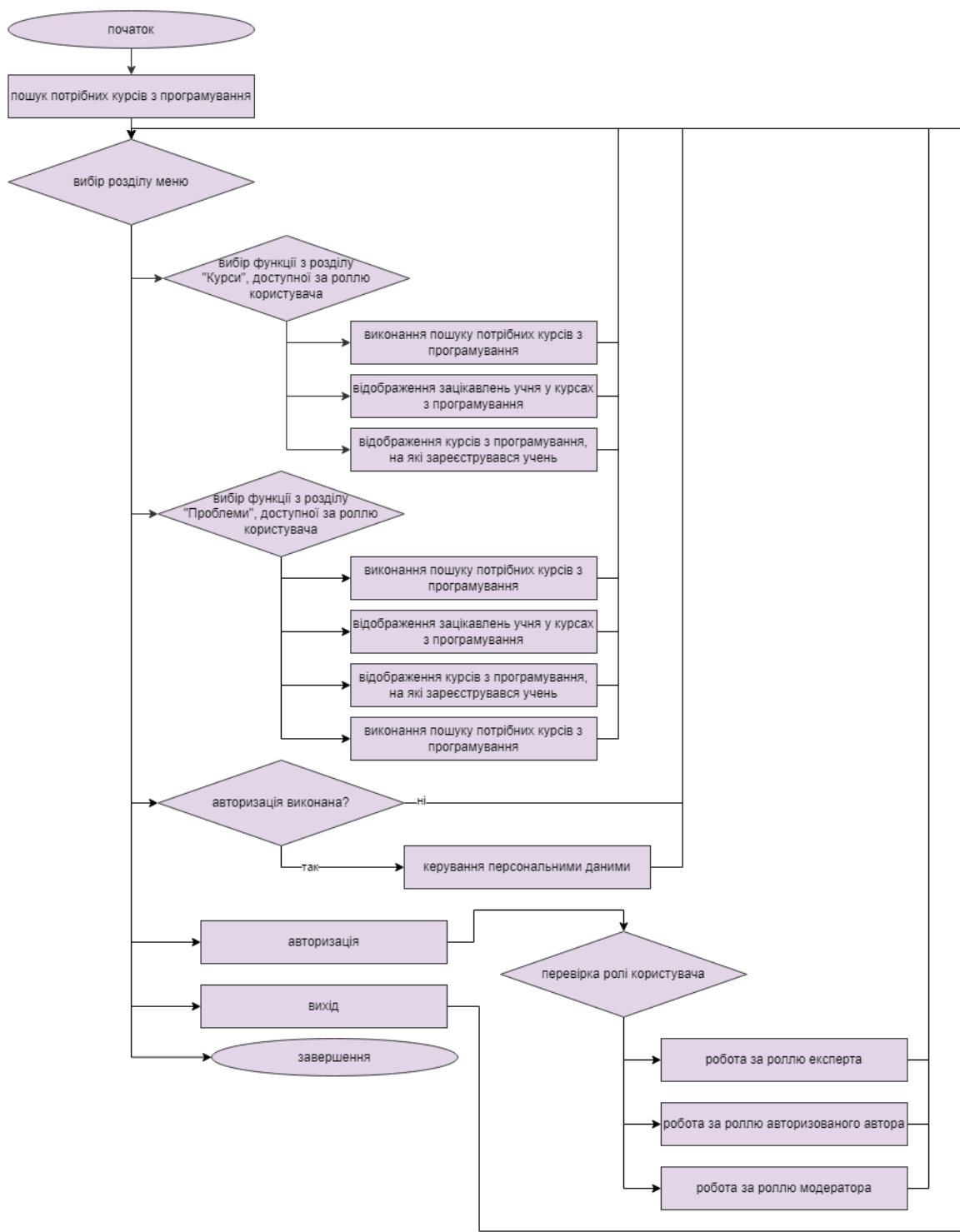


Рисунок 3.1 – Схема функціонування ПЗ для допомоги навчанню програмування

Схема функціонування програми побудована на базовому режимі, який визначає роботу учня з програмою, а ця робота побудована на виборі функцій з меню. Проте після виконання авторизації відбувається перевірка ролі: якщо учень, то базова логіка використовується, а якщо експерт, авторизований автор або модератор, то здійснюється робота за призначеною для них логікою, яка фактично включає ті ж розділи меню, але вже наповнені дещо іншими функціями.

3.2 Особливості програмної реалізації

Мови програмування визначені моделлю класу `Programlanguage` з єдиним програмно визначеним полем `lang_name`, що є полем назви мови програмування з типом `CharField`, а ідентифікатор стандартно реалізується у складі моделі ФВР Django, тому не потребує перевизначення.

Технології, які можуть використовуватися під час розробки програмного забезпечення, визначені моделлю класу `Programtechnology` з програмно визначеним полем `tech_name` типу `CharField`, що є полем назви технології, що може використовуватися під час розробки програмного забезпечення.

Курси, які можуть використовуватися для навчання програмуванню, визначені моделлю класу `Programcourse` з програмно визначеними полями:

- поле `short_caption` типу `CharField`, що є полем короткої назви курсу з програмування;
- поле `long_title` типу `CharField`, що є полем повної назви курсу з програмування;
- поле `poster` типу `ImageField`, що є полем фотографії курсу з програмування;
- поле `requirements` типу `TextField`, що є полем початкових вимог, яким має відповідати учень, щоби він міг навчатися за цим курсом з програмування;

- поле `result_skills` типу `TextField`, що є полем вмінь, які отримає учень за результатами вивчення даного курсу з програмування;
- поле `add_description` типу `TextField`, що є полем деталізованого опису курсу з програмування, яким може містити і розмітку для форматування тексту;
- поле `assessed_time` типу `DateTimeField`, що є полем очікуваного часу, який необхідно витрати для оволодіння всім даним курсом з програмування;
- поле `knowledge_level` типу `PositiveSmallIntegerField`, що є полем числового рівня знань, на якому викладається даний курс з програмування;
- поле `learningprice` типу `DecimalField`, що є полем вартості навчання за даним курсом з програмування;
- поле `mark` типу `DecimalField`, що є полем середньої оцінки, яку було виставлено даному курсу з програмування учнями, які зареєструвались на його вивчення;
- поле `repeat` типу `PositiveSmallIntegerField`, що є полем максимально допустимої кількості повторних разів проходження підсумкового тесту з даного курсу з програмування, якщо учень не зміг скласти тест на мінімальний допустимий бал з першого разу;
- поле `min_level` типу `DecimalField`, що є полем мінімального допустимого балу, на який потрібно здати тест за даним курсом з програмування, щоб успішно завершити навчання за ним;
- поле `authors` типу `ManyToManyField`, що є полем авторів, які є авторизованими експертами та які створили цей курс з програмування.

Курси, які можуть використовуватися для навчання програмуванню, визначені моделлю класу `Programcoursetopic` з програмно визначеними полями:

- поле `programcourse` типу `ForeignKey`, що є полем курсу з програмування, за яким вивчається дана тема;
- поле `short_caption` типу `CharField`, що є полем короткої назви теми курсу з програмування;

- поле `topic_name` типу `CharField`, що є полем повної назви теми курсу з програмування;
- поле `sequence_number` типу `PositiveSmallIntegerField`, що є полем порядкового номеру, за яким має вивчатися ця тема в даному курсі з програмування;
- поле `add_description` типу `TextField`, що є полем повного опису цієї теми курсу з програмування.

Матеріали вивчення курсів з програмування визначені моделлю класу `Programcoursestuff` з програмно визначеними полями:

- поле `programcoursetopic` типу `ForeignKey`, що є полем теми, за якою використовується цей матеріал для навчання програмуванню;
- поле `stuff_name` типу `CharField`, що є полем назви матеріалу, що вивчається за темою для навчання програмуванню за курсом;
- поле `sequence_number` типу `PositiveSmallIntegerField`, що є полем порядкового номеру, за яким має вивчатися даний матеріал в даній темі для навчання програмуванню за курсом;
- поле `assessed_time` типу `TimeField`, що є полем очікуваного часу, який необхідно витратити для повного вивчення цього матеріалу;
- поле `add_description` типу `TextField`, що є полем додаткового опису даного матеріалу для навчання програмуванню за курсом;
- поле `technologies` типу `ManyToManyField`, що є полем технологій, які можуть використовуватися під час розробки програмного забезпечення та які засвоюються під час вивчення цього курсу з програмування;
- поле `languages` типу `ManyToManyField`, що є полем мов програмування, які засвоюються під час вивчення цього курсу з програмування.

Форматований (розмічений) текст матеріалу для навчання програмуванню визначений моделлю класу `Programcoursestuffhtml` з програмно визначеними полями:

- поле `programcoursestuff` типу `OneToOneField`, що є полем матеріалу, що вивчається за темою для навчання програмуванню за курсом, який визначається цим форматованим (розміченим) текстом;

- поле `clear_html` типу `TextField`, що є полем тексту матеріалу, який може містити елементи розмітки для представлення його у вигляді сторінки.

Відеоматеріал для навчання програмуванню визначений моделлю класу `Programcoursestufffile` з програмно визначеними полями:

- поле `programcoursestuff` типу `OneToOneField`, що є полем матеріалу, що вивчається за темою для навчання програмуванню за курсом, який визначається цим відеоматеріалом за посиланням;

- поле `fileurl` типу `URLField`, що є полем посилання на файл для відображення відео;

- поле `description` типу `TextField`, що є полем опису відеоматеріалу.

Експерти (відповідальні автори курсів з програмування визначені моделлю класу `Tutor` з програмно визначеними полями:

- поле `officialname` типу `CharField`, що є полем офіційного повного імені експерта, який здійснює навчання програмуванню;

- поле `username` типу `CharField`, що є полем логіна експерта, який здійснює навчання програмуванню;

- поле `password` типу `CharField`, що є полем пароллю експерта, який здійснює навчання програмуванню;

- поле `poster` типу `ImageField`, що є полем фотографії експерта;

- поле `officialstatus` типу `PositiveSmallIntegerField`, що є полем статусу експерта, який здійснює навчання програмуванню: відповідальний автор курсів на платформі або експерт, який тільки може пропонувати свою допомогу щодо проблем, які визначив для себе учень, або модератор;

- поле `specialization` типу `CharField`, що є полем особливості спеціалізації експерта, який здійснює навчання програмуванню;

- поле `mark` типу `DecimalField`, що є полем середньої оцінки, яку було виставлено даному експерту за результатами навчання за проблемами учнів;

- поле `experience` типу `TextField`, що є полем досвіду експерта, який здійснює навчання програмуванню.

Учні, які навчаються програмуванню на платформі, визначені моделлю класу `Learner` з програмно визначеними полями:

- поле `officialname` типу `CharField`, що є полем офіційного повного імені учня, який навчається програмуванню на платформі;
- поле `username` типу `CharField`, що є полем логіна учня, який навчається програмуванню на платформі;
- поле `poster` типу `ImageField`, що є полем фотографії учня;
- поле `password` типу `CharField`, що є полем паролю учня, який навчається програмуванню на платформі;
- поле `programcourses` типу `ManyToManyField` відповідає набору зацікавлень даного учня в певних курсах з програмування, на які він ще не зареєструвався.

Реєстрація учня на вивчення курсу з програмування визначена моделлю класу `Programcourselicense` з програмно визначеними полями:

- поле `programcourse` типу `ForeignKey`, що є полем курсу з програмування, на вивчення якого зареєструвався учень;
- поле `learner` типу `ForeignKey`, що є полем учня, який зареєструвався на вивчення даного курсу з програмування (визначається зовнішнім ключем);
- поле `license_date` типу `DateTimeField`, що є полем часу, коли учень зареєструвався на вивчення курсу з програмування;
- поле `final_date` типу `DateTimeField`, що є полем часу, коли учень закінчив вивчення курсу з програмування;
- поле `paidamount` типу `DecimalField`, що є полем суми, яку сплатив за реєстрацію на вивчення даного курсу з програмування учень;
- поле `final_result` типу `DecimalField`, що є полем підсумкової оцінки за результатами тестування, яку за цим курсом з програмування отримав учень;
- поле `mark` типу `PositiveSmallIntegerField`, що є полем оцінки, яку даному курсу з програмування виставив учень;

– поле `repeat` типу `PositiveSmallIntegerField`, що є полем спроби, з якої було отримано дану оцінку.

Відмітки про вивчення певного матеріалу курсу з програмування визначені моделлю класу `Programcoursestufflearning` з програмно визначеними полями:

– поле `programcoursestuff` типу `ForeignKey`, що є полем матеріалу курсу з програмування, який вивчений учнем;

– поле `learner` типу `ForeignKey`, що є полем учня, який навчається програмуванню на платформі та який оволодів цим матеріалом курсу з програмування (визначається зовнішнім ключем);

– поле `learning_time` типу `DateTimeField`, що є полем часу, коли ця подія була зафіксована.

Проблеми, за якими учню необхідно виконати навчання в галузі програмування, визначені моделлю класу `Programproblem` з програмно визначеними полями:

– поле `learner` типу `ForeignKey`, що є полем учня, якому необхідно виконати навчання за певною проблемою в галузі програмування;

– поле `problem_description` типу `TextField`, що є полем опису проблеми в галузі програмування, за якою учень потребує навчання;

– поле `activeness` типу `BooleanField`, що є полем відмітки, необхідної для визначення того, чи ця проблема ще є актуальною, чи учень вже оволодів цією проблемою певним чином;

– поле `create_date` типу `DateTimeField`, що є полем часу внесення даних про проблему, за якою учню необхідно виконати навчання в галузі програмування;

– `max_price` типу `DecimalField`, що є полем вартості навчання, яку максимально за цією проблемою готовий сплатити учень.

Пропозиції, які експерти надають стосовно навчання за проблемами в галузі програмування, які виникають у учнів, визначені моделлю класу `Programproblemoffer` з програмно визначеними полями:

- поле `tutor` типу `ForeignKey`, що є полем експерта, який готовий виконати навчання учня за вказаною проблемою в галузі програмування;
- поле `programproblem` типу `ForeignKey`, що є полем проблеми, за якою учню необхідно виконати навчання в галузі програмування;
- поле `offer_description` типу `TextField`, що є полем особливості виконання навчання в галузі програмування за визначеною учнем проблемою в галузі програмування, які готовий надати експерт за власною пропозицією;
- поле `create_date` типу `DateTimeField`, що є полем часу, коли створена пропозиція, яку експерт надає стосовно навчання за проблемою в галузі програмування, яка виникла у учня;
- поле `mark` типу `PositiveSmallIntegerField`, що є полем оцінки, виставленої експертом за організоване проблемне навчання учнем;
- поле `status` типу `CharField`, що є полем стану, у якому перебуває пропозиція, яку експерт надає стосовно навчання за проблемою в галузі програмування, яка виникла у учня;
- поле `price_offer` типу `DecimalField`, що є полем вартості, за яку експерт готовий навчати учня за сформульованою ним проблемою в галузі програмування;
- поле `approve_date` типу `DateTimeField`, що є полем часу, коли пропозиція експерта стосовно навчання за проблемою в галузі програмування, яка виникла у учня, була погоджена або скасована учнем.

Матеріал, який застосовується експертом для навчання учня за проблемою в галузі програмування, яка є у учня, визначений моделлю класу `Programproblemstuff` з програмно визначеними полями:

- поле `programproblemoffer` типу `ForeignKey`, що є полем пропозиції, яку експерт надає стосовно навчання за проблемою в галузі програмування, що виникла у учня, та за якою надається цей матеріал для навчання;
- поле `stuff_name` типу `CharField`, що є полем назви матеріалу, який застосовується експертом для навчання учня за проблемою в галузі програмування, яка є в учня;

- поле `sequence_number` типу `PositiveIntegerField`, що є полем порядкового номеру, за яким має вивчатися даний матеріал, що застосовується експертом для навчання учня за проблемою в галузі програмування, яка є в учня;

- поле `add_description` типу `TextField`, що є полем додаткового опису даного матеріалу, що застосовується експертом для навчання учня за проблемою в галузі програмування, яка є в учня.

Форматований (розмічений) текст матеріалу для навчання програмуванню визначений моделлю класу `Programproblemstuffhtml` з програмно визначеними полями:

- поле `programproblemstuff` типу `OneToOneField`, що є полем матеріалу, який застосовується експертом для навчання учня за проблемою в галузі програмування, яка є в учня, до якого належить цей форматований (розмічений) текст;

- поле `clear_html` типу `TextField`, що є полем тексту матеріалу, який застосовується експертом для навчання учня за проблемою в галузі програмування, яка є в учня і який може містити елементи розмітки для представлення його у вигляді сторінки.

Відеоматеріал для навчання за проблемою учня в галузі програмування визначений моделлю класу `Programproblemstufffile` з програмно визначеними полями:

- поле `programproblemstuff` типу `OneToOneField`, що є полем матеріалу, який застосовується експертом для навчання учня за проблемою в галузі програмування, яка є в учня і який визначається цим відеоматеріалом за посиланням;

- поле `fileurl` типу `URLField`, що є полем посилання на файл для відображення відео, що застосовується експертом для навчання учня за проблемою в галузі програмування;

– поле `description` типу `TextField`, що є полем опису відеоматеріалу, який застосовується експертом для навчання учня за проблемою в галузі програмування.

Тестові запитання для перевірки засвоєння курсу з програмування визначені моделлю класу `Programcoursetest` з програмно визначеними полями:

– поле `programcourse` типу `ForeignKey`, що є полем курсу з програмування, за яким ставиться тестове запитання для перевірки засвоєння знань;

– поле `test_question` типу `CharField`, що є полем тексту запитання для перевірки засвоєння знань за курсом з програмування;

– поле `sequence_number` типу `PositiveSmallIntegerField`, що є полем порядкового номеру даного тестового запитання серед всіх запитань, які використовуються для перевірки засвоєння курсу з програмування;

– поле `points` типу `PositiveSmallIntegerField`, що є полем кількості балів, яку можна отримати за правильну відповідь на це тестове запитання.

Відповіді на тестове запитання для перевірки засвоєння курсу з програмування визначені моделлю класу `Programcoursetestanswer` з програмно визначеними полями:

– `programcoursetest` типу `ForeignKey`, що є полем тестове запитання для перевірки засвоєння курсу з програмування, до якого належить ця відповідь (визначається зовнішнім ключем);

– `test_answer` типу `CharField`, що є полем тексту відповіді на тестове запитання для перевірки засвоєння курсу з програмування;

– `sequence_number` типу `PositiveSmallIntegerField`, що є полем порядкового номеру даної відповіді серед відповідей на це тестове запитання для перевірки засвоєння курсу з програмування.

Представлення додатку розроблені для забезпечення сценаріїв, які були сформовані для учня, для авторизованого автора, для модератора, для експерта, проте з об'єднанням подібної логіки в одній функції. Наприклад, у програмі мала б бути передбачена функція перегляду проблем учня, за якими

він потребує навчання. Ця функція забезпечується для експерта. Але також мала б бути передбачена функція для учня перегляду власних проблем, за якими потрібне навчання. Фактично дії виконуються в основі однакові. Дані потрібні однакові. Але дещо інакше організовуються параметри. У першому випадку передається логін потрібного учня, а якщо авторизований користувач є експертом, то йому ці дані надаються. У другому випадку логін може не передаватися, а він береться за авторизованим обліковим записом. Тому і реалізацію підтримки на рівні представлення в такому випадку логічно було організувати однією функцією. Це функція `learner_programproblem_view`. Параметр `learner_name` у цій функції є опціональним: за замовчуванням він не має значення. Якщо значення вказано, то перевіряється, що користувач є експертом. Якщо значення не вказано, то береться логін авторизованого користувача, який має бути в такому випадку учнем.

Функція `programcourse_topics_view` забезпечує реалізацію представлення перегляду тем за власним курсом з програмування, а також перегляду тем за курсом з програмування, на вивчення якого зареєструвався учень. Сам номер курсу з програмування визначається параметром `progcourse_id`. За кожною темою надається перелік матеріалів, які їй відповідають, але тільки загальна частина, не включаючи вміст матеріалів.

Загальний перегляд сторінки курсу з програмування підтримується функцією `programcourse_view`, а номер курсу з програмування визначається параметром `progcourse_id`. Якщо авторизувався у програмі учень, то для нього перевіряється також, чи він зацікавлений у цьому курсі з програмування на даний момент, а також чи він підписався на цей курс.

Перегляд учнів, зареєстрованих на власний курс з програмування, забезпечується функцією `programcourse_licenses_view`, при чому параметр `progcourse_id` задає номер курсу з програмування. Перелік учнів повертається тільки тоді, коли курс з програмування, визначений номером, має автором авторизованого користувача, який є авторизованим автором до того ж.

Перегляд виставлених оцінок за курс з програмування реалізовано функцією `programcourse_marks_view`, де номер самого курсу з програмування визначається параметром `progcourse_id`.

Пошук потрібних курсів з програмування реалізується функцією `programcourse_find_view`. Параметри, задані в пошуковій формі, розбираються в функції на основі параметру `request` та атрибуту `GET`. Параметр `title` цього атрибуту задає текстову частину для пошуку потрібних курсів з програмування. Якщо параметр `full` не визначений, то пошук виконується тільки в назві курсу з програмування, а ось якщо він визначений, то пошук відбувається у всіх параметрах курсу, включаючи вимоги, вміння, опис. Параметр `langs` повинен включати номери мов програмування, перелічені через кому, які мають вивчатися в цьому курсі з програмування. Параметр `technologies` повинен включати номери технологій програмування, перелічені через кому, які мають вивчатися в цьому курсі з програмування. Рівень навчання задається в параметрі `level`. Параметр `price` задає максимальну вартість, яку допустимо сплатити за навчання. Параметр `minmark` задає найменшу оцінку, яку може мати курс з програмування в результатах пошуку. Параметр `maxmark` задає найбільшу оцінку, яку може мати курс з програмування в результатах пошуку. Варіант сортування результуючих курсів з програмування задається параметром `sort`:

- 0 – за зменшенням середньої оцінки курсів з програмування;
- 1 – за зменшенням вартості навчання, яку необхідно сплатити за курс з програмування;
- 2 – за збільшенням вартості навчання, яку необхідно сплатити за курс з програмування;
- 3 – за алфавітним порядком стосовно короткої назви курсу з програмування;
- 4 – за зворотнім алфавітним порядком стосовно короткої назви курсу з програмування.

Пошук серед проблем учнів у галузі програмування, за якими потрібно провести навчання, реалізується функцією `programproblem_find_view`. Обираються тільки ті проблеми в галузі програмування, визначені учнями, які є активним на момент пошуку. Параметри, задані в пошуковій формі, розбираються в функції на основі параметру `request` та атрибуту `GET`. Параметр `title` цього атрибуту задає текстову частину для пошуку серед проблем учнів у галузі програмування. Параметрами `mincreation` і `maxcreation` задається період, у який мали б бути створені записи про проблеми. Кожен параметр визначає дату і час. У той же час проміжок, в який має потрапляти максимальна вартість навчання, яку учень готовий сплатити, повинен знаходитись в інтервалі від `pricebot` до `pricetop`. Варіант сортування проблем у галузі програмування задається параметром `sort`:

- 0 – за зменшенням максимальної вартості, яку за навчання готовий сплатити учень;
- 1 – за збільшенням максимальної вартості, яку за навчання готовий сплатити учень;
- 2 – за датою створення за збільшенням;
- 3 – за датою створення з найменшої.

Інші функції подібно створені. Створення самих вебсторінок, які є результатами роботи представлень, забезпечено з використанням шаблону [9].

3.3 Висновки за розділом 3

Схема функціонування ПЗ для допомоги навчання програмуванню визначена на основі розкриття послідовності роботи зі сценаріями в режимі за замовчуванням, який є підтримкою учня, та можливістю переходу до режимів експерта, авторизованого автора або модератора. Описані створені класи моделей, деталі забезпечення представлень, які мають враховувати можливості різних ролей користувача та особливості сценаріїв в залежності від такого вибору.

4 ЕКСПЛУАТАЦІЯ ТА ТЕСТУВАННЯ ПРОГРАМИ

4.1 Призначення програми

Програма, яку створено, призначена для використання її експертами (вчителями) та учнями.

Учнями програма може використовуватись для реєстрації, проходження курсів, підведення підсумків навчання та визначення власних проблемних питань, за якими потрібно виконати індивідуальне навчання, вибору потрібного варіанту серед тих, які запропонували експерти.

Експертами програма може використовуватись для роботи над створенням власних курсів, на які може записуватися одразу багато учнів, та для формування власних пропозицій стосовно індивідуальної підготовки учнів, які потребують навчання за наявними проблемними питаннями.

4.2 Умови виконання програми

Умови виконання ПЗ та роботи з ним повинні визначатися з врахуванням відповідного запланованого навантаження. У даному випадку навантаження визначалось 50 запитами на секунду, а тому мінімально технічні засоби повинні включати:

- процесор не менше Intel Xeon E5440 з частотою 2.83 ГГц;
- оперативна пам'ять – 12 Гб;
- жорсткий диск – 40 Гб простору.

При цьому необхідно враховувати, що має бути забезпечена і програмна частина, яка на сервері має включати інтерпретатор Python версії 3, засоби ФВР Django, систему керування базами даних MySQL та надання відповідних прав доступу.

4.3 Експлуатація програми

Експлуатація програми побудована на взаємодії в першу чергу з меню, а вже далі додатково через певні функції, які містяться на сторінках вебдодатку. Тому меню побудована за єдиним принципом для різних ролей користувачів, включаючи два базові розділи: «Курси», «Проблеми». Тільки модератор додатково має пункт «Експерти», який дозволяє виконувати пошук потрібного експерта, після чого знайденому експерту можна надати статус авторизованого автора або скасувати його для того експерта, який такий статус вже має. Всі користувачі мають окрім 2 базових розділів у меню ще пункт «Персонально», який з'являється з того моменту, коли користувач авторизувався. Авторизація для експертів та учнів відбувається окремо, проте перейти до іншого варіанта можна зі сторінки авторизації, перемикаючись між варіантами. Авторизація експерта, авторизованого автора, модератора відбувається через той самий механізм.

Учень починає роботу зі сторінки пошуку курсів з програмування (рис. 4.1).

The screenshot shows the 'ПОШУК КУРСУ З ПРОГРАМУВАННЯ' (Search for programming course) form on the NazarProgramTutor website. The form is overlaid on a background of code. It includes the following elements:

- Header:** 'NazarProgramTutor' logo, navigation links 'Курси', 'Проблеми', 'Персонально', and a 'Вихід' (Exit) button.
- Form Title:** 'ПОШУК КУРСУ З ПРОГРАМУВАННЯ'.
- Search Field:** 'Назва курсу або інформація з опису' (Course name or description information) with a search icon and a checkbox for 'пошук тільки в назві' (search only in name).
- Language Filter:** 'Мови програмування: Python' with a dropdown menu showing 'C++' and a 'Додати' (Add) button.
- Technology Filter:** 'Технології програмування:' with a dropdown menu showing '.NET Maui' and a 'Додати' (Add) button.
- Learning Level Filter:** 'Рівень навчання:' with a dropdown menu showing 'складний' (complex).
- Rating Filters:** 'Максимальна оцінка' (Maximum rating) with a dropdown showing '5', and 'Мінімальна оцінка' (Minimum rating) with a dropdown showing '4'.
- Value Filter:** 'Максимальна вартість' (Maximum value) with a text input field.
- Sorting:** 'Напрямок сортування' (Sorting direction) with a dropdown menu showing 'рейтинг (за зменшенням)' (rating (by decreasing)).
- Search Button:** 'Шукати' (Search).

Рисунок 4.1 – Форма пошуку курсів з програмування

Сторінка пошуку курсів з програмування містить форму, в якій потрібно задати ряд параметрів пошуку, а мови та технології програмування можуть включати декілька варіантів (обрати з переліку потрібний варіант, натиснути кнопку додати, а для того щоб вилучити – натиснути на назву мови або технології, що призведе до вилучення). Кнопка «Шукати» відображає результати пошуку курсів з програмування.

У результатах пошуку внизу показується біля кожного курсу те, чи він є зареєстрованим та чи в ньому виявив зацікавленість учень, який авторизувався. Для експертів зацікавленості не відображаються.

Сторінка курсу з програмування (рис. 4.2) містить коротку та повну назву курсу (відповідно чорним та червоним кольором), опис (між якими знаходиться відмітка про віднесення курсу з програмування до зацікавлених, яку можна скасувати, натискаючи на галочку), інформацію про рівень навчання, очікувану кількість годин навчання, повну вартість навчання, середню виставлену оцінку, постер, інформацію про авторів курсу. За кожним автором курсу можна перейти до сторінки експерту. Окрім того за посиланням біля оцінки за курсом з програмування можна відкрити сторінку зі всіма оцінками, які учні, що були зареєстровані на вивчення курсу, виставили.

Для того щоб зареєструватися на вивчення курсу з програмування, потрібно натиснути на кнопку «Підписатися». Якщо учень вже зареєструвався на курс з програмування, то кнопка не доступна.

Після реєстрації на курс з програмування починає активуватися на сторінці нижче перелік матеріалів за темами (рис. 4.3). Кожна тема курсу з програмування демонструється на окремій вкладці. За кожною темою наводиться перелік матеріалів. Доступні матеріали демонструються як посилання. Тоді можна натиснути на назву матеріалу та відкрити його для перегляду на окремій сторінці, де показується відеоматеріал або текстова сторінка. Після того, як відео переглянуто або текст переглянуто, відкривається доступ до наступного матеріалу за порядком. В кінці доступним стає тестування за курсом з програмування для проходження оцінювання.

NazarProgramTutor

Курси ▾ Проблеми ▾ Персонально

Вихід

Складний

Рівень навчання

44 год.

Орієнтовний час вивчення

django

не для всіх

10000 €

Повна вартість навчання

5.0

Виставлена оцінка

всі оцінки

Автори курсу

Іван Полтавцев

Натан Романенко

Підписатися

Рисунок 4.2 – Сторінка курсу з програмування

NazarProgramTutor

Курси ▾ Проблеми ▾ Персонально

Вихід

Матеріали для навчання за темами

Розширені принципи створення додатків

Узагальнені відношення між моделями

Створення власних фільтрів шаблонів

Застосування тегів шаблонів

Параметри налаштування середовища

Створення логів

Кешування

Кешування пам'яті

Кешування бази даних

Кешування файлової системи

Кешування локальної пам'яті

Бекенд кешування

Кешування представлень

Кешування фрагментів шаблонів

Аналіз результатів

Використання засобів моніторингу для визначення вузьких місць

Метрики виконання додатку

Пришвидження за рахунок оптимізації

Вплив оптимізації запитів на результати

Вплив кешування на результати

Вимірювання одночасного виконання запитів

Розширені засоби захисту

Тестування додатків

Оптимізація фільтрів у деталях

Рисунок 4.3 – Теми курсу з програмування

Пошук курсів з програмування відбувається через розділ «Курси» та пункт «Пошук» в будь-який потрібний учню момент. З цього ж розділу пункт «Зацікавлення» надає можливість переглянути всі курси з програмування, які позначив учень як такі, в яких він зацікавлений. Пункт «Реєстрації» також доступний з даного розділу і дозволяє одразу отримати доступ до переліку курсів з програмування, на вивчення яких учень зареєструвався.

Розділ меню «Проблеми» для учня включає пункти:

- пункт «Власні» для доступу до всіх проблем в галузі програмування, за якими учень потребує навчання;
- пункт «Пропозиції» для доступу до всіх пропозицій щодо проблемного навчання в галузі програмування, які були створені експертами відносно тих проблем, про які заявив учень, що авторизувався;
- пункт «Реєстрації» для доступу до тих пропозицій, які були підтверджені учнем та за якими відбувається навчання;
- пункт «Додати» для додавання запиту на навчання стосовно нової проблеми в галузі програмування, яка виникла в учня.

Експерт, як і учень, має ті самі розділи в меню, але вони наповнені іншими пунктами. Розділ «Курси» для експерта містить тільки можливості переходу до пошуку курсів з програмування. Якщо експерт є ще й авторизованим автором, то у нього вже до цього пункту в даному розділі додається можливість перегляду власних курсів з програмування. З цієї сторінки можна редагувати курси, додавати авторів, вилучати курси за потреби, переходити до матеріалів курсу, звідки їх також можна редагувати, додавати. У цьому розділі для авторизованого автора також доступний пункт «Створити» для створення нового курсу з програмування.

Для експерта розділ «Проблеми» включає засоби пошуку серед проблем, визначених учнями, перегляду тих пропозицій, які авторизований експерт створив. Якщо експерт відкриває сторінку з описаною проблемою щодо навчання учня (рис. 4.4), то демонструються подані пропозиції, а експерт

може додати свою. Якщо експерт подав пропозицію про навчання, то може її скасувати.

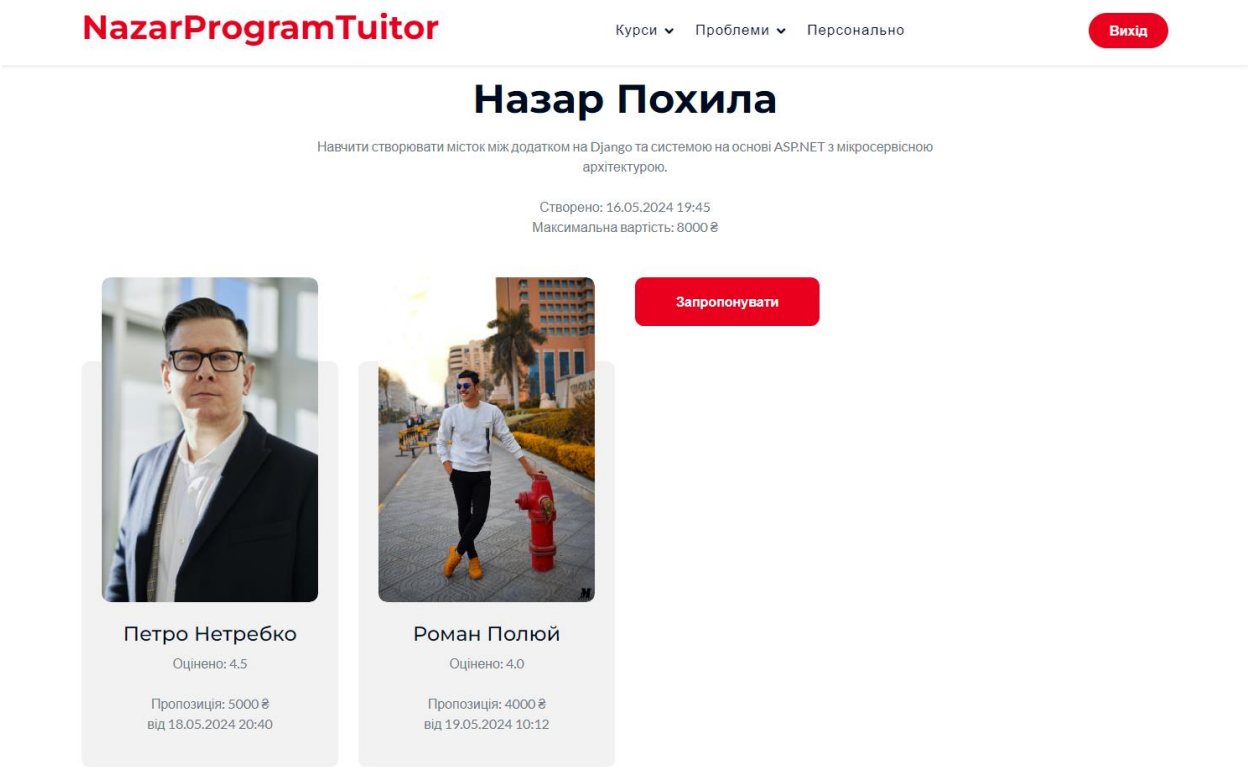


Рисунок 4.4 – Проблема в галузі програмування, за якою учень потребує навчання, та пропозиції за цією проблемою

4.4 Тестування програми

Тестування є обов’язковим і важливим для реалізації перевірки того, чи вдалось реалізувати заплановані функціональні можливості. Для перевірки необхідно кожен таку можливість перевірити та встановити результат (табл. 4.1). Якщо він співпадає з тим результатом, який мав бути отриманий, то результат позитивний і в таблиці позначається «+». Якщо він не співпадає, то позначається «-».

Результатом тестування ПЗ для підтримки навчання програмуванню є успішне виконання всіх функціональних можливостей.

Таблиця 4.1 – Результати тестування ПЗ для навчання програмуванню

Функціональна можливість	Результат
1	2
Пошук потрібних курсів з програмування	+
Пошук серед проблем учнів, за якими потрібно провести навчання	+
Перегляд проблем учня, за якими він потребує навчання	+
Перегляд пропозицій від експертів на проблему учня	+
Перегляд переліку пропозицій на всі проблеми учня	+
Реєстрація учня на вивчення курсу з програмування	+
Визначення зацікавлення учня у вивченні курсу з програмування	+
Деактивація зацікавлення учня у вивченні курсу з програмування	+
Перегляд курсів з програмування, у вивченні яких зацікавлений користувач	+
Перегляд переліку курсів з програмування, на вивчення яких зареєструвався учень	+
Перегляд тем за курсом з програмування	+
Перегляд курсів з програмування за заданою мовою програмування	+
Перегляд курсів з програмування за заданою технологією	+
Перегляд курсів з програмування за експертом	+
Перегляд пропозицій експерта з навчання за проблемами в галузі програмування	+
Перегляд навчань з програмування, на які зареєструвався учень	+
Перегляд учнів зареєстрованих на курс з програмування	+

Продовження таблиці 4.1

1	2
Перегляд матеріалу за курсом з програмування або навчанням за проблемою	+
Проходження тесту для перевірки засвоєння курсу з програмування	+
Створення курсу з програмування	+
Додавання матеріалу до курсу з програмування або навчання за проблемою	+
Редагування даних курсу з програмування	+
Редагування матеріалу курсу з програмування або навчання за проблемою	+
Створення пропозиції про навчання за проблемою	+
Скасування пропозиції про навчання за проблемою	+
Підтвердження пропозиції про навчання за проблемою	+
Авторизація	+
Пошук експертів з навчання програмуванню	+
Визнання експерта авторизованим автором	+
Реєстрація	+
Скасування визнання експерта авторизованим автором	+
Перегляд сторінки експерта	+
Перегляд сторінки учня	+
Перегляд сторінки курсу з програмування	+
Внесення проблеми в галузі програмування, за якою є потреба в навчанні	+
Редагування проблеми в галузі програмування, за якою є потреба в навчанні	+
Створення запитань для перевірки знань за курсом з програмування	+

Продовження таблиці 4.1

1	2
Редагування запитання для перевірки знань за курсом з програмування	+
Вилучення запитання для перевірки знань за курсом з програмування	+
Редагування відповіді на запитання для перевірки знань за курсом з програмування	+
Вилучення відповіді на запитання для перевірки знань за курсом з програмування	+
Перегляд запитань для перевірки знань за курсом з програмування	+
Перегляд виставлених оцінок за курс з програмування	+
Вилучення курсу з програмування	+
Перегляд виставлених експерту оцінок за результатами індивідуального навчання	+

4.5 Висновки за розділом 4

Реалізоване ПЗ для навчання програмуванню перевірено відносно реалізації запланованих функцій для експертів, учнів, модераторів. Проведене для цього тестування виявилось успішним та засвідчило виконання всіх функціональних вимог з очікуваними результатами.

ВИСНОВКИ

Через виконання завдань дипломної кваліфікаційної роботи сукупно було досягнуто мету, а успішну розробку ПЗ для допомоги при навчанні програмуванню з підтримкою можливості навчання на основі курсів та проблемного навчання було перевірено за допомогою проведення тестування ПЗ за сценаріями, що мали ним виконуватись. Сценарії визначались для учня, експерта, який може додатково бути також авторизованим автором курсів, а також для модератора. Для забезпечення підтримки визначених сценаріїв було детально опрацьовано структуру бази даних, що мало прямим результатом визначення моделей додатку, оскільки розроблене ПЗ у підсумку є Django-додатком.

Створена програма призначена для використання її експертами та учнями. Учнями програма може використовуватись для проходження курсів, підведення підсумків навчання та визначення власних проблемних питань, за якими потрібно виконати індивідуальне навчання, вибору потрібного варіанту серед тих, які запропонували експерти. Експертами програма може використовуватись для роботи над створенням власних курсів, на які може записуватися одразу багато учнів, та для формування власних пропозицій стосовно індивідуальної підготовки учнів, які потребують навчання за наявними проблемними питаннями (проблемного навчання).

Наявність можливостей проблемного навчання є особливістю виконаної розробки. Це дозволяє навчатися не тільки базовим знанням, але і досягати спеціалізованої підготовки в тих ситуаціях, коли, наприклад, існуючі джерела не дають чіткої відповіді, а навчання базується на професійному досвіді, врахуванні взаємозв'язків проблеми тощо. За рахунок наявності такого індивідуального навчання має реалізуватися привертання додаткової уваги до програми, що є важливим у даній розвиненій сфері.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Coursera. Degrees, Certificates, & Free Online Courses [Electronic resource]. – Access mode : www.coursera.org.
2. Online Courses – Learn Anything, On Your Schedule. Udemy [Electronic resource]. – Access mode : <https://www.udemy.com/>.
3. Learn Programming, Coding Online – Programming Hub [Electronic resource]. – Access mode : <https://programminghub.io/>.
4. Django – Full Stack Python [Electronic resource]. – Access mode : <https://www.fullstackpython.com/django.html>.
5. Dasanayaka, P. M. Web Frameworks for Python – Django & TurboGears [Electronic resource] / P. M. Dasanayaka. – Access mode : <https://medium.com/@piyumimdasanayaka/web-frameworks-for-python-django-turbogears-6733d9ea07ac>.
6. TurboGears Tutorial [Electronic resource]. – Access mode : <https://www.tutorialspoint.com/turbogears/index.htm>.
7. The Web Framework that scales with you – TurboGears2 Website 3.0 documentation [Electronic resource]. – Access mode : <https://turbogears.org/>.
8. TurboGears – Full Stack Python [Electronic resource]. – Access mode : <https://www.fullstackpython.com/turbogears.html>.
9. Cental – Car Rent Website Template [Electronic resource]. – Access mode : <https://htmlcodex.com/car-rent-website-template/>.

ДОДАТОК А
Технічне завдання

Вступ

На даний момент на ринку представлено декілька широко відомих платформ, включаючи Coursera, Udey та інші, які серед курсів пропонують підготовку і з програмування. Ці курси є авторськими, часом достатньо привабливими, однак всі вони мають типовий процес підготовки, який не кожному потенційному учню потрібен. Особливо в галузі програмування іноді потрібно забезпечити підготовку з окремих питань, які є більш спеціалізованими та потребують і спеціалізованих знань у експерта. У такому випадку наявні готові курси можуть не відповідати потребам учня, а це призводить до того, що потреби таких користувачів не задовольняються. Щоби вирішити цю проблему та забезпечити різносторонню підготовку, а за рахунок індивідуального навчання і привертання додаткової уваги до програми, необхідно розробити програмну систему, яка дозволить реалізувати підтримку навчання і за курсами з програмування, і за індивідуальною підготовкою.

A.1 Підстави для розробки

Наказом № 168 від 24 квітня 2024 р. за Національним університетом «Запорізька політехніка» визначено тему «Програмне забезпечення для допомоги при навчанні програмуванню» для виконання дипломної кваліфікаційної роботи.

A.2 Призначення розробки

Програма, яку буде створено в перспективі виконання даної роботи, призначена для використання її експертами (вчителями) та учнями. Учнями програма повинна використовуватись для реєстрації, проходження курсів, підведення підсумків навчання та визначення власних проблемних питань, за

якими потрібно виконати індивідуальне навчання, вибору потрібного варіанту серед тих, які запропонували експерти. Експертами програма повинна використовуватись для роботи над створенням власних курсів, на які може записуватися одразу багато учнів, та для формування власних пропозицій стосовно індивідуальної підготовки учнів, які потребують навчання за наявними проблемними питаннями.

А.3 Основні вимоги до програми

А.3.1 Вимоги до функціональних характеристик

Функціональні характеристики програми для допомоги при навчанні програмуванню повинні включати:

- пошук потрібних курсів з програмування;
- пошук серед проблем учнів, за якими потрібно провести навчання;
- перегляд проблем учня, за якими він потребує навчання;
- перегляд пропозицій від експертів на проблему учня;
- перегляд переліку пропозицій на всі проблеми учня;
- реєстрація учнів на вивчення курсів з програмування;
- визначення зацікавлення учня у вивченні курсу з програмування;
- деактивація зацікавлення учня у вивченні курсу з програмування;
- перегляд курсів з програмування, у вивченні яких зацікавлений користувач;
- перегляд переліку курсів з програмування, на вивчення яких зареєструвався учень;
- перегляд тем за курсом з програмування;
- перегляд курсів з програмування за заданою мовою програмування;
- перегляд курсів з програмування за заданою технологією;
- перегляд курсів з програмування за експертом;
- перегляд пропозицій експерта з навчання за проблемами в галузі програмування;

- перегляд навчань з програмування, на які зареєструвався учень;
- перегляд учнів зареєстрованих на курс з програмування;
- перегляд матеріалу за курсом з програмування або навчанням за проблемою;
- проходження тесту для перевірки засвоєння курсу з програмування;
- створення курсу з програмування;
- додавання матеріалу до курсу з програмування або навчання за проблемою;
- редагування даних курсу з програмування;
- редагування матеріалу курсу з програмування або навчання за проблемою;
- створення пропозиції про навчання за проблемою;
- підтвердження пропозиції про навчання за проблемою;
- авторизація;
- пошук експертів з навчання програмуванню;
- визнання експерта авторизованим автором;
- скасування визнання експерта авторизованим автором;
- перегляд сторінки експерта;
- перегляд сторінки учня;
- перегляд сторінки курсу з програмування;
- внесення проблеми в галузі програмування, за якою є потреба в навчанні;
- редагування проблеми в галузі програмування, за якою є потреба в навчанні;
- створення запитань для перевірки знань за курсом з програмування;
- редагування запитання для перевірки знань за курсом з програмування;
- вилучення запитання для перевірки знань за курсом з програмування;
- редагування відповіді на запитання для перевірки знань за курсом з програмування;

- реєстрація;
- вилучення відповіді на запитання для перевірки знань за курсом з програмування;
- скасування пропозиції про навчання за проблемою;
- перегляд запитань для перевірки знань за курсом з програмування;
- перегляд виставлених оцінок за курс з програмування;
- вилучення курсу з програмування;
- перегляд виставлених експерту оцінок за результатами індивідуального навчання.

A.3.2 Умови експлуатації

Експлуатація можлива тільки за використання сервера, на якому можливо встановити і в подальшому встановлено засоби, які включають інтерпретатор Python версії 3, засоби створення вебдодатків на основі ФВР Django, забезпечено доступ до системи керування базами даних MySQL з можливістю програмного створення бази даних засобами ФВР Django.

A.3.3 Вимоги до складу та параметрів технічних засобів

Технічні засоби визначені на рівні, що має забезпечувати приблизно 50 запитів на секунду:

- процесор не менше Intel Xeon E5440 з частотою 2.83 ГГц;
- оперативна пам'ять – 12 Гб;
- жорсткий диск – 40 Гб простору.

A.4 Порядок контролю та приймання

Виконання дипломної кваліфікаційної роботи бакалавра має бути закінчено у восьмитижневий термін з розподіленням за етапами, що визначено

завданням на роботу у відповідному календарному плані. Після виконання даного плану реалізується процедура захисту роботи. Захист відбувається перед екзаменаційною комісією.

ДОДАТОК Б
Текст програми

```

from django.shortcuts import render
from .models import Programcourse, Learner,
Programcourselicense, Tutor, Programlanguage, Programtechnology,
Programproblemoffer, Programproblem, Programcoursetopic,
Programcoursestuff
from django.db.models import Q
from django.urls import reverse
from django.http import HttpResponseRedirect

def programlanguage_courses_view(request, proglanguage_id):
    proglang = Programlanguage.objects.filter(id =
proglanguage_id)
    if proglang:
        proglang = proglang[0]
    else:
        return render(
            request,
            "programtutor/error.html",
            {
                "level": "Курси з програмування",
                "message": "Немає такої мови програмування.",
            },
        )
    programcourses =
Programcourse.objects.all().prefetch_related('technologies').prefetch_
related('languages').prefetch_related('authors').filter(languages__in
= [proglanguage_id]).prefetch_related('programcourses').order_by('-
mark')
    if request.user.is_active:
        learner = Learner.objects.filter(username =
request.user.username).prefetch_related('programcourses')
        if learner:
            learner = learner[0]
            for pcourse in programcourses:
                if pcourse in learner.programcourses:
                    pcourse.interested = True
                else:
                    pcourse.interested = False
            plicenses =
Programcourselicense.objects.filter(programcourse = pcourse, learner =
learner)
            if plicenses:

```

```

        pcourse.license = True
    else:
        pcourse.license = False
    return render(request, "programtutor/langcourses.html",
programcourses)

def programtechnology_courses_view(request, progtechnology_id):
    progtech = Programtechnology.objects.filter(id =
progtechnology_id)
    if progtech:
        progtech = progtech[0]
    else:
        return render(
            request,
            "programtutor/error.html",
            {
                "level": "Курси з програмування",
                "message": "Немає такої технології
програмування.",
            },
        )
    programcourses =
Programcourse.objects.all().prefetch_related('technologies').prefetch_
related('languages').prefetch_related('authors').filter(technologies__
in = [progtechnology_id]).order_by('-mark')
    if request.user.is_active:
        learner = Learner.objects.filter(username =
request.user.username).prefetch_related('programcourses')
        if learner:
            learner = learner[0]
            for pcourse in programcourses:
                if pcourse in learner.programcourses:
                    pcourse.interested = True
                else:
                    pcourse.interested = False
            plicenses =
Programcourselicense.objects.filter(programcourse = pcourse, learner =
learner)

            if plicenses:
                pcourse.license = True
            else:
                pcourse.license = False

```

```

        return render(request,
            "programtutor/technologycourses.html", programcourses)

    def expertauthor_courses_view(request, tutor_name = None):
        if tutor_name:
            course_tutor = Tutor.objects.filter(username =
tutor_name)
        elif request.user.is_active:
            course_tutor = Tutor.objects.filter(username =
request.user.username)
        if course_tutor:
            course_tutor = course_tutor[0]
            programcourses =
Programcourse.objects.all().prefetch_related('technologies').prefetch_
related('languages').prefetch_related('authors').filter(authors__in =
[course_tutor]).order_by('-mark')
            if request.user.is_active:
                learner = Learner.objects.filter(username =
request.user.username).prefetch_related('programcourses')
                if learner:
                    learner = learner[0]
                    for pcourse in programcourses:
                        if pcourse in learner.programcourses:
                            pcourse.interested = True
                        else:
                            pcourse.interested = False
                    plicenses =
Programcourselicense.objects.filter(programcourse = pcourse, learner =
learner)

                    if plicenses:
                        pcourse.license = True
                    else:
                        pcourse.license = False
            return render(request,
                "programtutor/expauthcourses.html", programcourses)
        else:
            return render(
                request,
                "programtutor/error.html",
                {
                    "level": "Курси з програмування",
                    "message": "Необхідно бути або авторизованим
автором, або вказаного авторизованого автора не існує.",

```

```

        },
    )

    def expert_problems_view(request, tutor_name = None):
        if tutor_name:
            course_tutor = Tutor.objects.filter(username =
tutor_name)
        elif request.user.is_active:
            course_tutor = Tutor.objects.filter(username =
request.user.username)
        if course_tutor:
            course_tutor = course_tutor[0]
            programoffers =
Programproblemoffer.objects.all().select_related('programproblem').sel
ect_related('tutor').select_related('programproblem__learner
').filter(tutor = course_tutor).order_by('-approve_date')
            return render(request,
"programtutor/expertoffers.html", programoffers)
        else:
            return render(
                request,
                "programtutor/error.html",
                {
                    "level": "Курси з програмування",
                    "message": "Необхідно бути або авторизованим
експертом, або вказаного експерта не існує.",
                },
            )

    from .forms import TutorForm

    def expert_page_view(request, tutor_name = None):
        if tutor_name:
            course_tutor_param = Tutor.objects.filter(username =
tutor_name)
        else:
            course_tutor_param = None
        if request.user.is_active:
            course_tutor = Tutor.objects.filter(username =
request.user.username)
        if course_tutor:
            course_tutor = course_tutor[0]
            if not course_tutor_param:

```

```

        course_tutor_param = course_tutor
        programoffers =
Programproblemoffer.objects.all().select_related('programproblem').sel
ect_related('tutor').select_related('programproblem__learner
').filter(tutor = course_tutor).order_by('-approve_date')[:3]
        programcourses =
Programcourse.objects.all().prefetch_related('technologies').prefetch_
related('languages').prefetch_related('authors').filter(authors__in =
[course_tutor]).order_by('-mark')[:3]
        if course_tutor_param and course_tutor ==
course_tutor_param:
            tutor_form = TutorForm(request.POST or None,
instance = course_tutor)
            if request.method == "POST" and
tutor_form.is_valid():
                tutor_form.save()
                return
HttpResponseRedirect(reverse("programtutor:tutor",
args=(course_tutor_param.username,)))
            else:
                tutor_form = None
            if request.user.is_active:
                learner = Learner.objects.filter(username =
request.user.username).prefetch_related('programcourses')
                if learner:
                    learner = learner[0]
                    for pcourse in programcourses:
                        if pcourse in learner.programcourses:
                            pcourse.interested = True
                        else:
                            pcourse.interested = False
                    plicenses =
Programcourselicense.objects.filter(programcourse = pcourse, learner =
learner)

                    if plicenses:
                        pcourse.license = True
                    else:
                        pcourse.license = False
                return render(request, "programtutor/expert.html",
{"offers": programoffers, "courses": programcourses, "tutor_form":
tutor_form })
        return render(
            request,
            "programtutor/error.html",

```

```

        {
            "level": "Курси з програмування",
            "message": "Необхідно бути або авторизованим експертом, або вказаного експерта не існує.",
        },
    )

    from .forms import LearnerForm

    def learner_page_view(request, learner_name = None):
        course_learner = None
        if learner_name:
            course_learner_param = Learner.objects.filter(username = learner_name)
            if course_learner_param:
                course_learner_param = course_learner_param[0]
            if request.user.is_active:
                course_learner = Learner.objects.all().prefetch_related('programcourses').filter(username = request.user.username)
                if course_learner:
                    course_learner = course_learner[0]
                    if not course_learner_param:
                        course_learner_param = course_learner
                course_tutor = Tutor.objects.filter(username = request.user.username)
                if course_tutor:
                    course_tutor = course_tutor [0]
            else:
                return render(
                    request,
                    "programtutor/error.html",
                    {
                        "level": "Профіль учня",
                        "message": "Необхідно бути авторизованим учнем або експертом.",
                    },
                )
            if course_learner == course_learner_param or course_tutor:
                programoffers = Programproblemoffer.objects.all().select_related('programproblem').select_related('tutor').select_related('programproblem__learner

```

```

').filter(programproblem__learner = course_learner).order_by('-
approve_date')[:3]
        programcourses =
Programcourselicense.objects.all().prefetch_related('programcourse').p
refetch_related('learner').order_by('-license_date')[:3]
        if request.user.is_active:
            for pcourse in programcourses:
                if pcourse in course_learner.programcourses:
                    pcourse.interested = True
                else:
                    pcourse.interested = False
            plicenses =
Programcourselicense.objects.filter(programcourse = pcourse, learner =
learner)

            if plicenses:
                pcourse.license = True
            else:
                pcourse.license = False
        if course_learner == course_learner_param:
            learner_form = LearnerForm(request.POST or None,
instance = course_learner)
            if request.method == "POST" and
learner_form.is_valid():
                learner_form.save()
                return
HttpResponseRedirect(reverse("programtutor:learner",
args=(course_learner_param.username,)))
        else:
            learner_form = None
            return render(request, "programtutor/learner.html",
{"offers": programoffers, "courses": programcourses, "learner_form":
learner_form })
        return render(
            request,
            "programtutor/error.html",
            {
                "level": "Профіль учня",
                "message": "Немає прав для доступу до сторінки учня.
Учень може отримати доступ тільки до своєї сторінки",
            },
        )

def programproblem_view(request, problem_id):
    if problem_id:

```

```

        progproblem =
Programproblem.objects.all().select_related('learner').filter(id =
problem_id)
        if progproblem:
            progproblem = progproblem[0]
        if progproblem and request.user.is_active:
            course_learner =
Learner.objects.all().prefetch_related('programcourses').filter(userna
me = request.user.username)
            if course_learner:
                course_learner = course_learner[0]
            course_tutor = Tutor.objects.filter(username =
request.user.username)
            if course_tutor:
                course_tutor = course_tutor [0]
        else:
            return render(
                request,
                "programtutor/error.html",
                {
                    "level": "Пропозиції щодо навчання",
                    "message": "Пропозиції щодо навчання за
проблемою учня доступні авторизованим користувачам. Окрім того дані
проблеми можуть бути вказані некоректно.",
                },
            )
        if course_learner == progproblem.learner or course_tutor:
            programoffers =
Programproblemoffer.objects.all().select_related('programproblem').sel
ect_related('tutor').select_related('programproblem__learner
').filter(programproblem = progproblem).order_by('-create_date')
            return render(request,
"programtutor/programproblem.html", {"offers": programoffers})
        else:
            return render(
                request,
                "programtutor/error.html",
                {
                    "level": "Пропозиції щодо навчання",
                    "message": "Недостатньо прав для доступу до
пропозицій щодо навчання за проблемою учня.",
                },
            )

```

```

def learner_programproblem_view(request, learner_name = None):
    course_learner = None
    if learner_name:
        course_learner_param = Learner.objects.filter(username =
learner_name)
        if course_learner_param:
            course_learner_param = course_learner_param[0]
        if request.user.is_active:
            course_learner =
Learner.objects.all().prefetch_related('programcourses').filter(userna
me = request.user.username)
            if course_learner:
                course_learner = course_learner[0]
                if not course_learner_param:
                    course_learner_param = course_learner
            course_tutor = Tutor.objects.filter(username =
request.user.username)
            if course_tutor:
                course_tutor = course_tutor [0]
        else:
            return render(
                request,
                "programtutor/error.html",
                {
                    "level": "Проблеми учня в галузі програмування
для початку",
                    "message": "Потрібно спершу авторизуватися.",
                },
            )
        if course_learner == course_learner_param or course_tutor:
            programproblems =
Programproblem.objects.all().select_related('learner').filter(learner
= course_learner).order_by('-create_date')
            return render(request,
"programtutor/learnerprogramproblems.html", {"problems":
programproblems })
        else:
            return render(
                request,
                "programtutor/error.html",
                {
                    "level": "Проблеми учня в галузі програмування
для початку",

```

```

        "message": "Недостатньо прав для доступу до
проблем учня в галузі програмування.",
    },
)

def programcourse_topics_view(request, progcourse_id):
    if progcourse_id:
        progcourse =
Programcourse.objects.all().prefetch_related('authors').filter(id =
progcourse_id)
        if progcourse:
            progcourse = progcourse[0]
        if progcourse and request.user.is_active:
            course_learner =
Learner.objects.all().prefetch_related('programcourses').filter(userna
me = request.user.username)
            if course_learner:
                course_learner = course_learner[0]
            course_tutor = Tutor.objects.filter(username =
request.user.username)
            if course_tutor:
                course_tutor = course_tutor [0]
        else:
            return render(
                request,
                "programtutor/error.html",
                {
                    "level": "Курс з програмування",
                    "message": "такий курс з програмування відсутній
або ви не авторизувались.",
                },
            )
        if not request.user.is_active:
            return render(
                request,
                "programtutor/error.html",
                {
                    "level": "Курс з програмування",
                    "message": "Ви не автооризувались.",
                },
            )
        if course_learner:

```

```

        licenses =
Programcourselicense.objects.filter(programcourse = progcourse,
learner = course_learner)
        if licenses or (course_tutor and course_tutor in
progcourse.authors) or (course_tutor.officialstatus == 2):
            programcourse_topics =
Programcoursetopic.objects.all().select_related('programcourse').filter(
programcourse = progcourse).order_by('sequence_number')
            for programtopic in programcourse_topics:
                programtopic.stuffs =
Programcoursestuff.objects.filter(programcoursetopic =
programtopic).order_by('sequence_number')
                for programstuff in programtopic.stuffs:
                    progleraning =
Programcoursestufflearning.objects.filter(programcoursestuff =
programstuff, learner = course_learner)
                    if progleraning:
                        programstuff.done = True
                    else:
                        programstuff.done = False
            if progcourse in course_learner.programcourses:
                progcourse.interested = True
            else:
                progcourse.interested = False
            return render(request,
"programtutor/coursetopics.html", {"course": progcourse, "topics":
programcourse_topics })
        else:
            return render(
                request,
                "programtutor/error.html",
                {
                    "level": "Курс з програмування",
                    "message": "Ви не зареєструвались на даний курс
з програмування або не є автором даного курсу.",
                },
            )

def programcourse_view(request, progcourse_id):
    if progcourse_id:
        progcourse =
Programcourse.objects.all().prefetch_related('authors').filter(id =
progcourse_id)

```

```

        if progcourse:
            progcourse = progcourse[0]
        if progcourse and request.user.is_active:
            course_learner =
Learner.objects.all().prefetch_related('programcourses').filter(userna
me = request.user.username)
            if course_learner:
                course_learner = course_learner[0]
        if course_learner:
            if progcourse in course_learner.programcourses:
                progcourse.interested = True
            else:
                progcourse.interested = False
        plicenses =
Programcourselicense.objects.filter(programcourse = progcourse,
learner = course_learner)
        if plicenses:
            progcourse.license = True
        else:
            progcourse.license = False
        return render(request, "programtutor/programcourse.html",
{"course": progcourse })

```

```

def programcourse_licenses_view(request, progcourse_id):
    if progcourse_id:
        progcourse =
Programcourse.objects.all().prefetch_related('authors').filter(id =
progcourse_id)
        if progcourse:
            progcourse = progcourse[0]
        if progcourse and request.user.is_active:
            course_tutor = Tutor.objects.filter(username =
request.user.username)
            if course_tutor:
                course_tutor = course_tutor [0]
        else:
            return render(
                request,
                "programtutor/error.html",
                {
                    "level": "Курс з програмування",

```

```

        "message": "Інформація про зареєстрованих на
курс учнів доступна тільки авторизованому користувачу та за наявності
такого курсу.",
    },
)
if not request.user.is_active:
    return render(
        request,
        "programtutor/error.html",
        {
            "level": "Курс з програмування",
            "message": "Інформація про зареєстрованих на
курс учнів доступна тільки авторизованому користувачу.",
        },
    )
    if course_tutor and ((course_tutor in progcourse.authors)
or (course_tutor.officialstatus == 2)):
        licenses =
Programcourselicense.objects.all().select_related('programcourse').sel
ect_related('learner').filter(programcourse = progcourse).order_by('-
license_date')
        return render(request,
"programtutor/progcourselicenses.html", {"course": progcourse,
"licenses": licenses })
    else:
        return render(
            request,
            "programtutor/error.html",
            {
                "level": "Курс з програмування",
                "message": "Інформація про зареєстрованих на
курс учнів доступна тільки авторизованим автором зазначеного курсу з
програмування.",
            },
        )

def programcourse_marks_view(request, progcourse_id):
    marks = []
    if progcourse_id:
        progcourse =
Programcourse.objects.all().prefetch_related('authors').filter(id =
progcourse_id)
        if progcourse:

```

```

        progcourse = progcourse[0]
        marks =
Programcourselicense.objects.all().select_related('learner').filter(pr
ogramcourse = progcourse).order_by('-final_date')
        if progcourse and request.user.is_active:
            course_learner =
Learner.objects.all().prefetch_related('programcourses').filter(userna
me = request.user.username)
            if course_learner:
                course_learner = course_learner[0]
        if course_learner:
            if progcourse in course_learner.programcourses:
                progcourse.interested = True
            else:
                progcourse.interested = False
        return render(request, "programtutor/coursemarks.html",
{"course": progcourse, "marks": marks })
    def expert_marks_view(request, tutor_name = None):
        if tutor_name:
            course_tutor = Tutor.objects.filter(username =
tutor_name)
        elif request.user.is_active:
            course_tutor = Tutor.objects.filter(username =
request.user.username)
        if course_tutor:
            course_tutor = course_tutor[0]
            marks =
Programproblemoffer.objects.all().select_related('programproblem').sel
ect_related('programproblem__learner ').filter(tutor =
course_tutor).order_by('-approve_date')
            return render(request, "programtutor/expertmarks.html",
{"marks": marks })
        else:
            return render(
                request,
                "programtutor/error.html",
                {
                    "level": "Оцінки експерта",
                    "message": "Потрібно задати коректні дані
експерта або бути авторизованим експертом.",
                },
            )

```

```

def programcourse_find_view(request):
    course_caption_part = self.request.GET.get('title', None)
    programcourses =
Programcourse.objects.all().prefetch_related('languages').prefetch_rel
ated('technologies')
    if course_caption_part:
        programcourses =
programcourses.filter(short_caption__icontains = course_caption_part)
        full_text_search = self.request.GET.get('full', None)
        if full_text_search:
            programcourses =
programcourses.filter(Q(long_title__icontains =
course_caption_part)|Q(requirements__icontains = course_caption_part)
|Q(result_skills__icontains = course_caption_part)
|Q(add_description__icontains = course_caption_part))
            proglangs = self.request.GET.get('langs', '')
            if proglangs:
                proglangs = proglangs.split(',')
                search_proglangs = []
                for proglang in proglangs:
                    proglang_id = int(proglang)
                    proglang = Programlanguage.objects.filter(id =
proglang_id)

                    if proglang:
                        proglang = proglang[0]
                        search_proglangs.append(proglang)
                programcourses = programcourses.filter(languages__in =
search_proglangs)
            progtechnologies = self.request.GET.get('technologies', '')
            if progtechnologies:
                progtechnologies = progtechnologies.split(',')
                search_progtechnologies = []
                for technology in progtechnologies:
                    technology_id = int(technology)
                    technology = Programtechnology.objects.filter(id =
technology_id)

                    if technology:
                        technology = technology [0]
                        search_progtechnologies.append(technology)
                programcourses = programcourses.filter(technologies__in
= search_progtechnologies)
            course_knowledge_level = self.request.GET.get('level', None)
            if course_knowledge_level:

```

```

        course_knowledge_level = int(course_knowledge_level)
        programcourses = programcourses.filter(knowledge_level =
course_knowledge_level)
        course_learningprice = self.request.GET.get('price', None)
        if course_learningprice:
            course_learningprice = int(course_learningprice)
            programcourses =
programcourses.filter(learningprice__lte = course_learningprice)
        course_min_mark = self.request.GET.get('minmark', None)
        if course_min_mark:
            course_min_mark = int(course_min_mark)
            programcourses = programcourses.filter(mark__gte =
course_min_mark)
        course_max_mark = self.request.GET.get('maxmark', None)
        if course_max_mark:
            course_max_mark = int(course_max_mark)
            programcourses = programcourses.filter(mark__lte =
course_max_mark)
        course_sort = self.request.GET.get('sort', None)
        if course_sort:
            course_sort = int(course_sort)
            if course_sort == 0:
                programcourses = programcourses.order_by('-mark')
            elif course_sort == 1:
                programcourses = programcourses.order_by('-
learningprice')
            elif course_sort == 2:
                programcourses =
programcourses.order_by('learningprice')
            elif course_sort == 3:
                programcourses =
programcourses.order_by('short_caption')
            elif course_sort == 4:
                programcourses = programcourses.order_by('-
short_caption')
        if request.user.is_active:
            learner = Learner.objects.filter(username =
request.user.username).prefetch_related('programcourses')
            if learner:
                learner = learner[0]
                for pcourse in programcourses:
                    if pcourse in learner.programcourses:
                        pcourse.interested = True

```

```

        else:
            pcourse.interested = False
            plicenses =
Programcourselicense.objects.filter(programcourse = pcourse, learner =
learner)

            if plicenses:
                pcourse.license = True
            else:
                pcourse.license = False
        return render(request,
"programtutor/programcoursefind.html", {"courses": programcourses,
"languages": Programlanguage.objects.all(), "technologies":
Programtechnology.objects.all()})

def programproblem_find_view(request):
    problem_caption_part = self.request.GET.get('title', None)
    programproblems =
Programproblem.objects.all().select_related('learner')
    programproblems = programproblems.filter(activeness = True)
    if problem_caption_part:
        programproblems =
programproblems.filter(problem_description__icontains =
problem_caption_part)
    problem_create_min = self.request.GET.get('mincreation',
None)

    from datetime import datetime
    if problem_create_min:
        problem_create_min =
datetime.strptime(problem_create_min, '%m.%d.%y %H:%M')
        programproblems =
programproblems.filter(create_date__gte = problem_create_min)
    problem_create_max = self.request.GET.get('maxcreation',
None)

    if problem_create_max:
        problem_create_max =
datetime.strptime(problem_create_max, '%m.%d.%y %H:%M')
        programproblems =
programproblems.filter(create_date__lte = problem_create_max)
    problem_max_price = self.request.GET.get('pricetop', None)
    if problem_max_price:
        problem_max_price = int(problem_max_price)
        programproblems = programproblems.filter(max_price__lte
= problem_max_price)

```

```

        problem_min_price = self.request.GET.get('pricebot', None)
        if problem_min_price:
            problem_min_price = int(problem_min_price)
            programproblems = programproblems.filter(max_price__gte
= problem_min_price)
        problem_sort = self.request.GET.get('sort', None)
        if problem_sort:
            problem_sort = int(problem_sort)
            if problem_sort == 0:
                programproblems = programproblems.order_by('-
max_price')
            elif problem_sort == 1:
                programproblems =
programproblems.order_by('max_price')
            elif problem_sort == 2:
                programproblems =
programproblems.order_by('create_date')
            elif problem_sort == 3:
                programproblems = programproblems.order_by('-
create_date')
        return render(request,
"programtutor/programproblemfind.html", programproblems)

    def problemoffer_cancel_view(request, progproblemoffer_id):
        if progproblemoffer_id:
            prog_problem_offer =
Programproblemoffer.objects.all().select_related('tutor').select_rela
ted('programproblem').select_related('programproblem__learner').filter
(id = progproblemoffer_id)
            if prog_problem_offer:
                prog_problem_offer = prog_problem_offer[0]
            else:
                return render(
                    request,
                    "programtutor/error.html",
                    {
                        "level": "Пропозиція щодо навчання за
проблемою учня",
                        "message": "Такої пропозиції не існує.",
                    },
                )
            if prog_problem_offer and request.user.is_active:

```

```

        course_tutor = Tutor.objects.filter(username =
request.user.username)
        if course_tutor:
            course_tutor = course_tutor[0]
        course_learner = Learner.objects.filter(username =
request.user.username)
        if course_learner:
            course_learner = course_learner[0]
        if not course_tutor and not course_learner:
            return render(
                request,
                "programtutor/error.html",
                {
                    "level": "Пропозиція щодо навчання за
проблемою учня",
                    "message": "Немає прав для виконання
скасування.",
                },
            )
        if (course_tutor and prog_problem_offer.tutor ==
course_tutor and prog_problem_offer.status == 0) or (course_learner
and prog_problem_offer.programproblem.learner == course_learner and
prog_problem_offer.status == 0):
            prog_problem_offer.status = 2
            prog_problem_offer.save()
            return
HttpResponseRedirect(reverse("programtutor:programproblem", args=(
prog_problem_offer.programproblem.id,)))
        else:
            return render(
                request,
                "programtutor/error.html",
                {
                    "level": "Пропозиція щодо навчання за проблемою
учня",
                    "message": "Немає прав для виконання
скасування.",
                },
            )

from .forms import ProgramproblemofferForm

def problemoffer_add_view(request):

```

```

        course_tutor = Tutor.objects.filter(username =
request.user.username)
        if course_tutor:
            course_tutor = course_tutor[0]
        if not request.user.is_active:
            return render(
                request,
                "programtutor/error.html",
                {
                    "level": "Пропозиція щодо навчання за проблемою
учня",
                    "message": "Необхідно авторизуватись для
створення пропозиції щодо навчання за проблемою учня в галузі
програмування.",
                },
            )
        if request.method == "POST":
            problem_offer_form =
ProgramproblemofferForm(request.POST)
            if problem_offer_form.is_valid() and
request.user.is_active:
                problem_offer =
problem_offer_form.save(commit=False)
                problem_offer.tutor = course_tutor
                problem_offer.status = 0
                problem_offer.save()
                return
HttpResponseRedirect(reverse("programtutor:programproblem", args=(
problem_offer.programproblem.id,)))
            else:
                problem_offer_form = ProgramproblemofferForm()
                return render(request, "programtutor/make_offer.html",
{"offer_form": problem_offer_form})

```

ДОДАТОК В
Слайди презентації

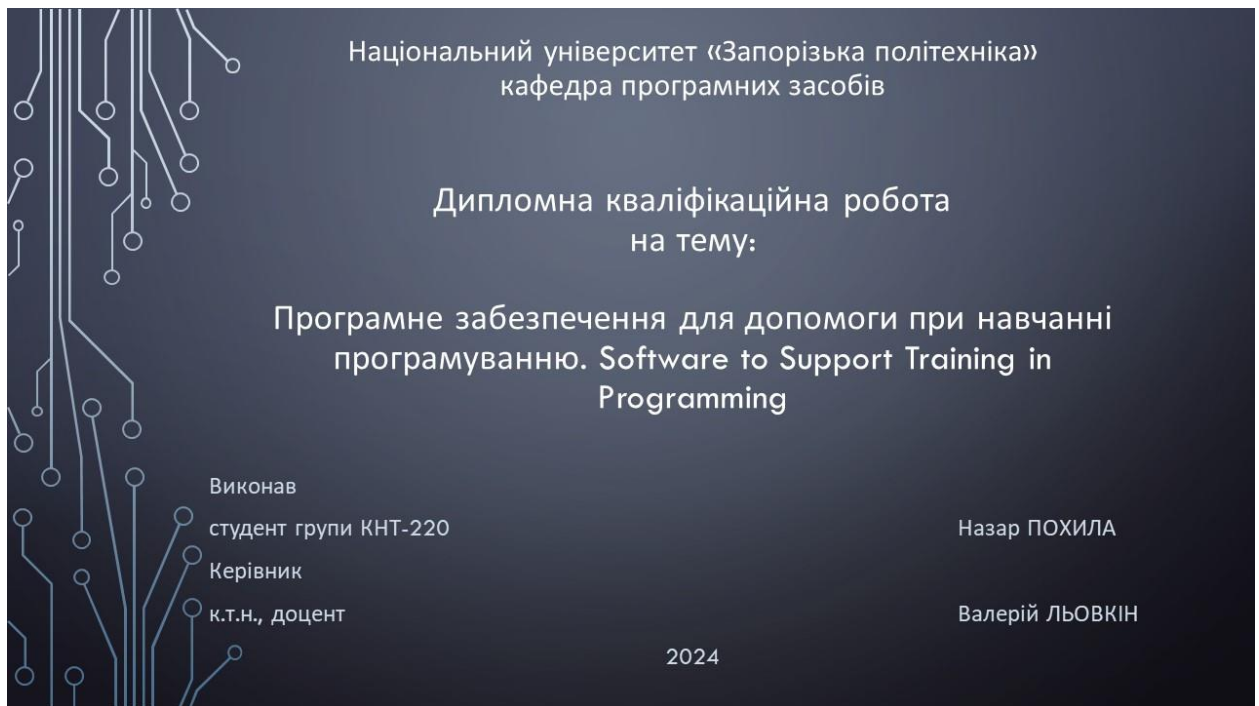


Рисунок В.1 – Початковий слайд

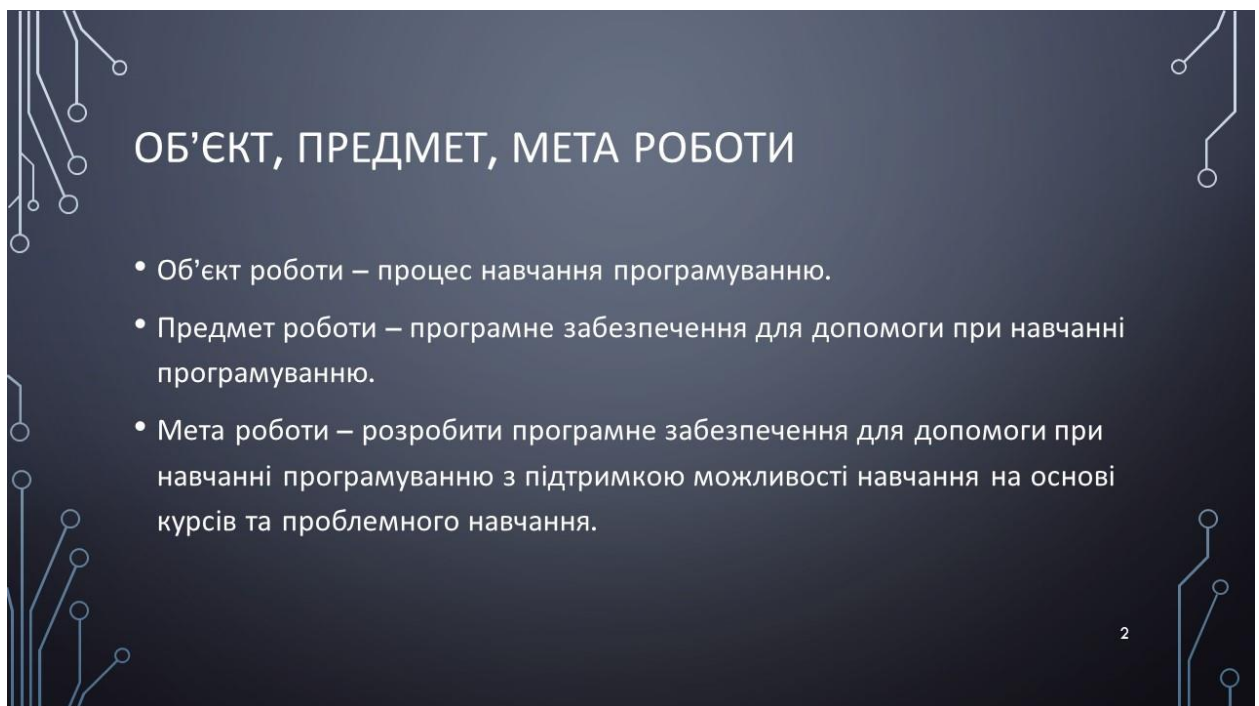


Рисунок В.2 – Об'єкт, предмет, мета роботи

ЗАВДАННЯ РОБОТИ

- проаналізувати проблему розробки програми для допомоги при навчанні програмуванню;
- спроектувати програмне забезпечення для допомоги при навчанні програмуванню;
- реалізувати програмне забезпечення для допомоги при навчанні програмуванню та виконати його тестування.

3

Рисунок В.3 – Завдання роботи

ПОРІВНЯННЯ ІСНУЮЧИХ АНАЛОГІВ ТА РОЗРОБЛЮВАНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ДОПОМОГИ ПРИ НАВЧАННІ ПРОГРАМУВАННЮ

Характеристика	Coursera	Udemy	Programming Hub	Виконувана розробка
Навчання за готовими курсами з програмування	+	+	+	+
Пошук курсів з програмування	+	+	–	+
Потенційні автори курсів	Обмежене коло	Без обмежень	Обмежене коло	Без обмежень, але з наданням права
Визначення зацікавленості в курсі	+	+	–	+
Оцінювання курсів з програмування	+	+	–	+
Пошук експертів для навчання за проблемами учня	–	–	–	+
Оцінювання експертів	–	–	–	+

4

Рисунок В.4 – Порівняння існуючих аналогів та розроблюваного програмного забезпечення для допомоги при навчанні програмуванню

ДИДАКТИЧНІ ВАРІАНТИ ВИКОРИСТАННЯ ПРОГРАМУВАННЯ ЗАБЕЗПЕЧЕННЯ ДЛЯ ДОПОМОГИ ПРИ НАВЧАННІ ПРОГРАМУВАННЮ УЧНЕМ ТА МОДЕРАТОРОМ

Рисунок В.6 – Діаграми варіантів використання програмного забезпечення для допомоги при навчанні програмуванню учнем та модератором

[illegible]

Рисунок В.8 – Структура бази даних програмного забезпечення для допомоги при навчанні програмуванню



Рисунок В.9 – Схема функціонування програмного забезпечення для допомоги навчання програмування

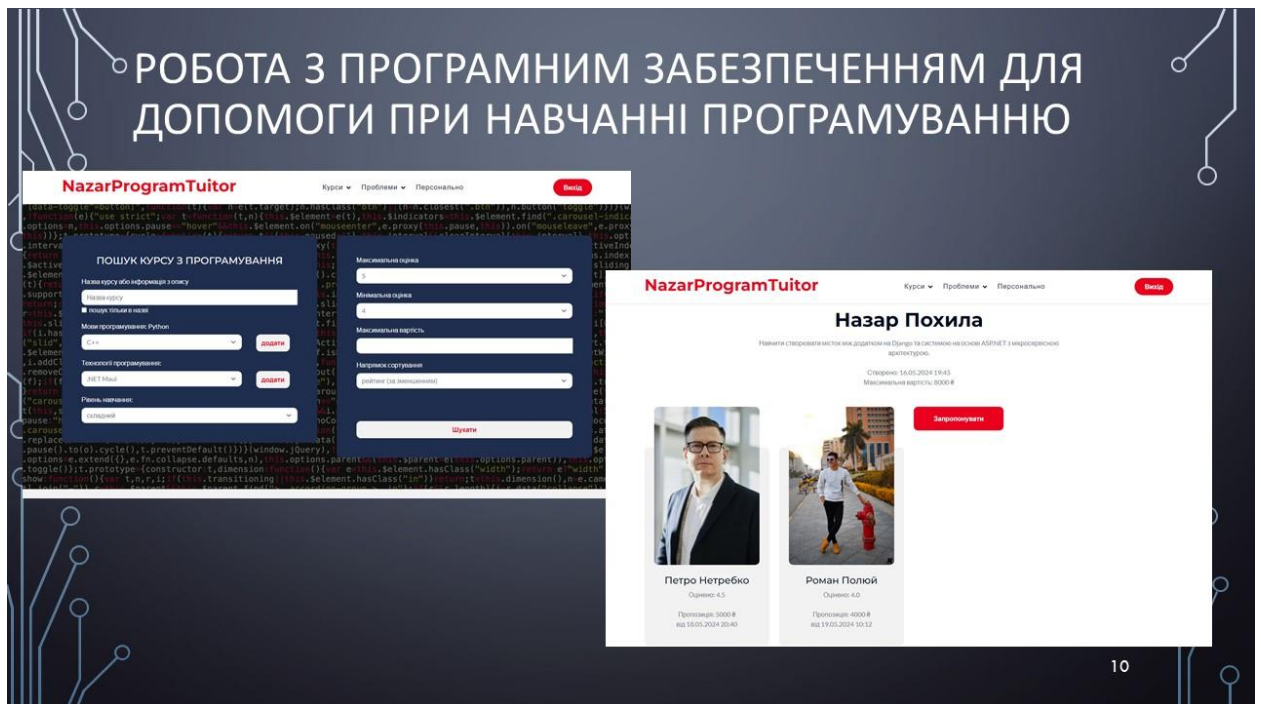


Рисунок В.10 – Робота з програмним забезпеченням для допомоги при навчанні програмуванню

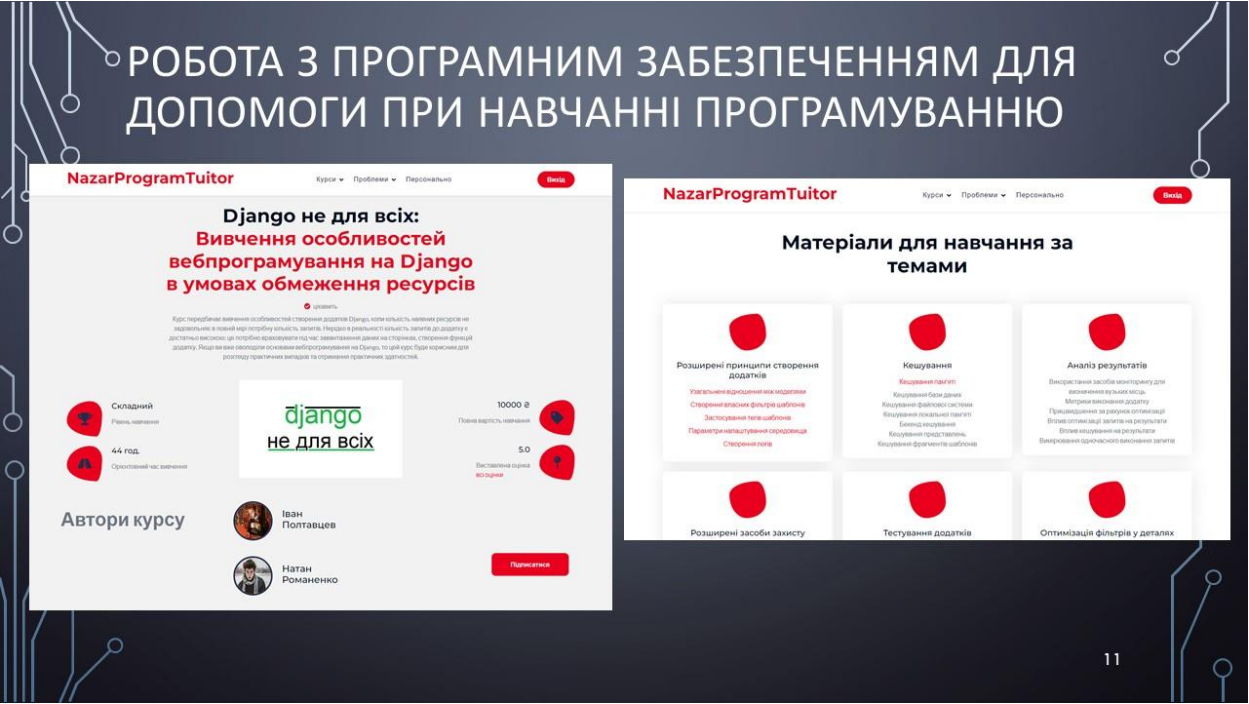


Рисунок В.11 – Робота з програмним забезпеченням для допомоги при навчанні програмуванню

ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ НАВЧАННЯ ПРОГРАМУВАННЮ	
Функціональна можливість	Результат
Пошук потрібних курсів з програмування	+
Пошук серед проблем учнів, за якими потрібно провести навчання	+
Перегляд проблем учня, за якими він потребує навчання	+
Перегляд пропозицій від експертів на проблему учня	+
Перегляд переліку пропозицій на всі проблеми учня	+
Реєстрація учня на вивчення курсу з програмування	+
Визначення зацікавлення учня у вивченні курсу з програмування	+
Деактивація зацікавлення учня у вивченні курсу з програмування	+
Перегляд курсів з програмування, у вивченні яких зацікавлений користувач	+
Перегляд переліку курсів з програмування, на вивчення яких зареєструвався учень	+
Перегляд тем за курсом з програмування	+
Перегляд курсів з програмування за заданою мовою програмування	+
Перегляд курсів з програмування за заданою технологією	+
Перегляд курсів з програмування за експертом	+
Перегляд пропозицій експерта з навчання за проблемами в галузі програмування	+

Функціональна можливість	Результат
Редагування запитання для перевірки знань за курсом з програмування	+
Вилучення запитання для перевірки знань за курсом з програмування	+
Редагування відповіді на запитання для перевірки знань за курсом з програмування	+
Вилучення відповіді на запитання для перевірки знань за курсом з програмування	+
Перегляд запитань для перевірки знань за курсом з програмування	+
Перегляд виставлених оцінок за курс з програмування	+
Вилучення курсу з програмування	+
Перегляд навчань з програмування, на які зареєструвався учень	+
Перегляд учнів зареєстрованих на курс з програмування	+
Перегляд виставлених експерту оцінок за результатами індивідуального навчання	+

Рисунок В.12 – Тестування програмного забезпечення для навчання програмуванню

ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ НАВЧАННЯ ПРОГРАМУВАННЮ	
Функціональна можливість	Результат
Перегляд матеріалу за курсом з програмування або навчанням за проблемою	+
Проходження тесту для перевірки засвоєння курсу з програмування	+
Створення курсу з програмування	+
Додавання матеріалу до курсу з програмування або навчання за проблемою	+
Редагування даних курсу з програмування	+
Редагування матеріалу курсу з програмування або навчання за проблемою	+
Створення пропозиції про навчання за проблемою	+
Скасування пропозиції про навчання за проблемою	+
Підтвердження пропозиції про навчання за проблемою	+
Авторизація	+
Пошук експертів з навчання програмуванню	+
Визнання експерта авторизованим автором	+
Реєстрація	+
Скасування визнання експерта авторизованим автором	+
Перегляд сторінки експерта	+
Перегляд сторінки учня	+
Перегляд сторінки курсу з програмування	+
Внесення проблеми в галузі програмування, за якою є потреба в навчанні	+
Редагування проблеми в галузі програмування, за якою є потреба в навчанні	+
Створення запитань для перевірки знань за курсом з програмування	+

13

Рисунок В.13 – Тестування програмного забезпечення для навчання програмуванню

ВИСНОВКИ
Через виконання завдань дипломної кваліфікаційної роботи сукупно було досягнуто мету, а успішну розробку програмного забезпечення для допомоги при навчанні програмуванню з підтримкою можливості навчання на основі курсів та проблемного навчання було перевірено за допомогою проведення тестування програмного забезпечення за сценаріями, що мали ним виконуватись. Сценарії визначались для учня, експерта, який може додатково бути також авторизованим автором курсів, а також для модератора. Для забезпечення підтримки визначених сценаріїв було детально опрацьовано структуру бази даних, що мало прямим результатом визначення моделей додатку, оскільки розроблене програмне забезпечення у підсумку є Django-додатком. Створена програма призначена для використання її експертами та учнями. Учнями програма може використовуватись для проходження курсів, підведення підсумків навчання та визначення власних проблемних питань, за якими потрібно виконати індивідуальне навчання, вибору потрібного варіанту серед тих, які запропонували експерти. Експертами програма може використовуватись для роботи над створенням власних курсів, на які може записуватися одразу багато учнів, та для формування власних пропозицій стосовно індивідуальної підготовки учнів, які потребують навчання за наявними проблемними питаннями (проблемного навчання).

14

Рисунок В.14 – Висновки