МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХЕРСОНСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
Факультет комп'ютерних наук, фізики та математики
Кафедра комп'ютерних наук та програмної інженерії

СТВОРЕННЯ ПРОГРАМНОГО ЗАСОБУ НАВЧАЛЬНОГО ПРИЗНАЧЕННЯ «МАТЕМАТИКА ТА ПРОГРАМУВАННЯ»

Кваліфікаційна робота
на здобуття ступеня вищої освіти «бакалавр»

Виконав: студент 4 курсу 441 групи
Спеціальності: 121 Інженерія програмного забезпечення
Освітньо-професійної програми: Інженерія програмного забезпечення
Березовський Н. В.
Керівник: проф. Львов М. С.
Співкерівник: Співаковський О. В.
Рецензент: Кузьміч В. І., професор кафедри алгебри, геометрії та математичного аналізу ХДУ

Херсон – 2022

ЗМІСТ

ВСТУП

Електронний навчальний курс – це освітнє електроне видання або інформаційна система комплексного призначення для реалізації дидактичних можливостей засобів інформаційно-комунікаційних технологій та підтримки навчального процесу в закладах загальної, спеціальної, професійної освіти, а також для самоосвіти у межах навчальних програм, у тому числі спрямованих на безперервну освіту. ЕНК є частиною електронної освіти. [1]

Мета дослідження: сконструювати веб-сервіс «програмування математичних задач»

Об’єкт дослідження: електронний збірник задач

Предмет дослідження: веб-сервіс «програмування математичних задач»

Завдання дослідження:

1. Вибір засобів розробки
2. Розробка веб-сервісу «програмування математичних задач»

Методи дослідження: вивчення навчальної літератури та інтернет ресурсів, аналіз методів розробки та створення власного методу вирішення різних проблем.

РОЗДІЛ 1
АНАЛІЗ ЗБІРНИКІВ ЗАДАЧ

1.1. Призначення збірника задач

Система управління навчанням (LMS) — це програмне забезпечення, розроблене спеціально для створення, розповсюдження та керування доставкою освітнього вмісту. LMS може розміщуватися як окремий продукт на сервері компанії, або це може бути хмарна платформа, яку розміщує фірма, що займається програмним забезпеченням.

Подумайте про систему управління навчанням як про технологію, яка може покращити навчання, зробити його швидшим, продуктивним, економічно ефективним і, що ще важливіше, – відстежуваним. Найпростіший LMS містить основну функціональну платформу, яка дозволяє адміністраторам завантажувати навчальний контент, проводити уроки учням, надсилати сповіщення та обмінюватися даними з авторизованими користувачами. LMS найчастіше працює всередині веб-браузера, за безпечним процесом входу. Це дає всім студентам і викладачам легкий доступ до курсів на ходу, а адміністратори та керівники можуть відстежувати прогрес студентів і вносити покращення. Існує кілька типів систем управління навчанням, які організації можуть вибрати, перевірте список порівняння:

- Підприємство проти особи
- Безкоштовні проти комерційних
- SaaS проти ліцензованого продукту
- Хмарний або локальний
- Інтегрований
- Вбудовані інструменти розробки або попередньо написана CMS

Підприємство проти особи. Рішення для навчання корпоративного рівня: на найвищому рівні рішення для навчання підприємства, як правило, налаштовується для великої компанії з 500 або більше співробітниками. Він дуже масштабований і може бути розроблений для зростання разом

з компанією та в міру додавання нових курсів. Часто корпоративний LMS має величезний список вбудованих функцій для задоволення будь-яких потреб, які можуть знадобитися.

Індивідуальний LMS: індивідуальний або невеликий організаційний LMS може пропонувати декілька або всі доступні функції, але обмежує їх тим, що може знадобитися окремому виробнику курсу. Однак це може послужити невеликій організації або окремій людині для виготовлення різноманітних навчальних матеріалів. Часто постачальники LMS пропонують різні плани підписки для задоволення різних потреб, тому невелика організація може використовувати LMS для своїх потреб і за відповідною ціною.

Безкоштовні проти комерційних. Безкоштовний LMS: існують також десятки безкоштовних продуктів електронного навчання з відкритим кодом, щоб окремі особи могли створювати курси для своїх учнів. Будь-хто, хто має знання, може застосувати їх і створити курс, доступ до якого можна отримати через середовище електронного навчання. Багато безкоштовних LMS також пропонують розширені рівні та великий список доступних функцій. Основна проблема безкоштовної LMS полягає в тому, що вони не мають того рівня підтримки, до якого звикли деякі користувачі. Ви повинні знати, як його розгорнути, або найняти для цього спеціаліста. Оновлення платформи вимагатимуть вашої уваги та часу й не завжди проходять гладко.

Комерційний: комерційний продукт призначений для використання будь-якою організацією, коледжем чи університетом чи іншою первинною освітньою організацією. Як правило, його легко розгорнути, і він підтримує цілодобову підтримку. Він також поставляється з цінником в залежності від рівня необхідних функцій. Переважно комерційні платформи потребують частого оновлення, але вони також забезпечують зростаючий список доступних функцій, тому ви завжди

будете використовувати найсучасніше високотехнологічне рішення для електронного навчання.

SaaS проти ліцензованого продукту. SaaS LMS: Мабуть, найпопулярнішою моделлю для LMS є модель сервісу як програмного забезпечення (SaaS). Це готове рішення, яке легко розгорнути. Зазвичай він базується на хмарі та включає часті безкоштовні оновлення. Хоча це може бути дещо обмежено в налаштуванні, SaaS LMS ідеально підходить для організації, яка розвивається, оскільки вона масштабується з часом. Технічна підтримка зазвичай включається на весь період використання.

Ліцензований продукт: ліцензована система управління навчанням може запропонувати користувачам найкращі налаштування, підтримку клієнтів та надійність. Він більш гнучкий і розроблений відповідно до специфікацій кожного клієнта, що може зайняти деякий час. Його можна встановити на хмарному сервері або локально. Запровадження може відбутися за бажанням організації, але оновлення може зайняти більше часу. Найбільша вартість – ліцензія користувача, однак ліцензовані продукти виграють на великій відстані, особливо якщо вони реалізовані як хмарне рішення. Це робить вартість володіння та вартість на одного користувача значно нижчими в порівнянні з рішеннями SaaS. Підтримка може бути обмежена часом або включена як платне доповнення.

Хмарний або локальний. Хмарний LMS: хмарний LMS розміщується на захищеному сервері поза сайтом і дозволяє користувачам отримувати доступ до продукту за допомогою різноманітних комп’ютерів і мобільних пристроїв, навіть у автономному режимі за допомогою програми. Він може включати багато найпопулярніших функцій, які очікують учні. Хмарна LMS набагато більш масштабована, ніж встановлена версія.

Внутрішня LMS: створена як окремий продукт, локальна LMS є ліцензованим продуктом, створеним і встановленим відповідно до

специфікацій організації на сервері за їх вибором. Це може бути обмеженим, у великих масштабах вам доведеться використовувати кілька серверів. Хмарна установка в деяких випадках може бути більш економічно ефективною. Внутрішня інсталяція підтримує користувацькі та корпоративні дані на основі високої безпеки.

Інтегрований: сучасна LMS враховує наявне програмне забезпечення та системи, які є в організації, а потім поєднується з ними для безперебійної роботи користувача. Інтеграція передбачає підключення LMS з іншими сумісними продуктами для покращення досвіду учня. Звичайно, організація може вирішити залишити систему управління навчанням повністю окремою від усіх інших систем і продуктів. Однак це вимикає деякі можливості, які може мати на увазі ця інтеграція.

Вбудовані інструменти розробки або попередньо написана CMS. Вбудовані інструменти розробки: більшість сучасних LMS включає інструменти для створення вмісту або підтримує стандарт Experience API/xAPI (або SCORM) і дозволяє розробникам завантажувати вже створені курси та уроки з інструментів розробки або LCMS/CMS. Загалом, розробники курсів вважають за краще мати доступ до вбудованих інструментів розробки, які дають змогу повноцінно створювати та переглядати курси. У більшості випадків можна включити навіть попередньо створений вміст і інтегрувати нові модулі, вправи, навчальні документи, оцінки тощо.

Попередньо написана CMS: систему керування вмістом (CMS) часто плутають із LMS, це інший продукт. Однак CMS можна використовувати для розміщення деяких навчальних матеріалів та проектної документації. Це також може бути хорошим місцем для команд L&D, щоб розробити дизайн курсу задовго до початку запуску реальних курсів. Однак він не замінить LMS. Попередньо написані уроки курсу можна розробити тут і імпортувати в LMS пізніше.

LMS повинна мати можливість:

- Надавати досвід навчання, адаптований до окремих учнів
- Робити так, щоб викладачі могли легко робити нотатки та вносити зміни
- Давати викладачам і студентам можливість співпраці в Інтернеті
- Інтегрувати звичайні інструменти, такі як календарі, текстові процесори тощо
- Створювати фірмову присутність з урахуванням корпоративної культури для учнів
- Додавати статистику прогресу користувачів за допомогою вбудованої аналітики
- Надавати можливість глобального масштабування в міру зростання організації

Сучасна система управління навчанням часто має вбудовані інструменти та ресурси, які допомагають адміністраторам розробляти уроки, заходи та оцінювання курсу. Адміністратори можуть призначати нові облікові дані користувача та планувати проходження курсів. Вони також можуть відстежувати прогрес учнів за допомогою функцій звітності.

Хороший LMS допомагає зробити навчання цікавим, захоплюючим, щоб вони брали більш активну роль у власному розвитку. LMS має бути простим у доступі та використанні, щоб заохочувати учнів до участі. Дизайн LMS повинен бути дружнім за зовнішнім виглядом і функціональністю - на основі вимог користувача. [2]

1.2. Аналіз інших систем електронного навчання

Херсонський віртуальний університет (DLS) – програмне забезпечення, запроваджене у Херсонському Державному Університеті, призначений для адміністрування навчальних курсів у рамках дистанційного навчання.

РОЗДІЛ 2.
КОНСТРУЮВАННЯ ВЕБ-СЕРВІСУ «ПРОГРАМУВАННЯ МАТЕМАТИЧНИХ ЗАДАЧ»

2.1. Опис проекту

Мета поліпшення якості знань студентів вищих навчальних закладів у області науки визначає необхідність створення інтерактивних електронних освітніх ресурсів для вирішення завдань з математичної моделі програмування. Цей проект було задумано для збільшення якості підготовки студентів вищих навчальних закладів у області математики та програмування. В часності, його можна використовувати як підручник на факультативних заняттях та в клубах молодих програмістів, так само як у процесі індивідуальної праці, аби тренувати учнів шкіл та студентів для олімпіад із програмування.

Вміст електронного підручника заснований на наборі математичних задач. Кожна задача містить сам текст задачі, інструкції з її розв'язання та алгоритм для цього, виражений у формі програмного коду на мовах програмування Pascal, C/C++ та Python.

Програма електронного підручника – це веб-додаток, побудований на клієнт-серверній архітектурі. Основою веб-додатку є програмний модуль «підручник», який містить набір математичних задач із алгоритмом розв'язку від автора. Веб-додаток містить модуль для розширення вмісту електронного підручника у форму вбудованого редактора для математичних задач. Уповноважений користувач системи має можливість додавати нові задачі до вже існуючого підручника з описом алгоритму розв'язку. Автоматична система перевірки запропонованого алгоритму висвітлює основну експертну думку стосовно додавання нової задачі до бібліотеки для подальшого використання.

Перевірка веб-додатку в освітньому процесі здійснювалась в загальноосвітніх навчальних закладах міста Херсона, та отримав позитивний відгук від учнів та вчителів.

Задачі проекту:

- Зробити вміст та рівень учнів у математиці більш наближеним до рівня університету
- Підготовити учнів до рівня прикладної математики
- Значно підвищити компонент алгоритму учнів у програмуванні
- Заохочувати вивчанню мов програмування, англійська як мова професійної комунікації для IT спеціалістів.

Як бачимо, ЕТМ побудовано за принципом сукупності завдань. Одним із основних недоліків існуючих збірників завдань є те, що вони не часто містять інструкцій до розв'язання, ані, тим більше, правильних (з точки зору авторів) рішень у вигляді вихідних кодів. Такі збірники завдань не дуже підходять для самостійної роботи учнів та студентів. Наш досвід навчання основам алгоритмізації та програмування молодших школярів, старшокласників, а також слухачів курсів підвищення кваліфікації вчителів інформатики свідчить, що без «правильних підказок» розв'язування простих завдань програмування неефективне.

Вирішувати завдання олімпіадного рівня складності в процесі початкового навчання основам алгоритмізації та програмування без підказок просто неможливо. Більше того, часто навіть вказівок на рішення на рівні змістовного представлення алгоритмів недостатньо. Відомо, що важливими аспектами ефективного рішення є структури даних та адекватні їм структури керування, представлення яких можливе або псевдомовою, або мовою програмування.

Програмне забезпечення ЕТМ – це веб-додаток, який має архітектуру клієнт-сервер. Серверна частина програми працює під керуванням Web-сервера і забезпечує користувача Інтернетом за допомогою веб-браузера.

Технологічні вимоги до програмного забезпечення ЕТМ зводяться до вимог до створення програмного середовища, що забезпечує вирішення завдань алгоритмів програмування розв'язування математичних задач:

- авторизований доступ до освітньої інформації;
- системне адміністрування;
- ефективний розвиток інформаційних ресурсів користувачами.

При цьому ЕТМ відповідає основним вимогам щодо організації дистанційного навчання в Інтернеті, а саме:

- забезпечення доступу до навчальних матеріалів за допомогою Інтернету;
- забезпечення авторських прав на електронні навчальні ресурси, створені користувачами;
- накопичення в базі даних системи навчальних інформаційних ресурсів (математичних задач, інструкцій до розв’язування, алгоритмів розв’язування математичних задач у вигляді програмного коду тощо) у форматі стандарту IMS, їх імпорт та експорт;
- забезпечення групової роботи в мережі;
- інформування користувачів про хід і результати навчального процесу;
- поширення (розповсюдження) навчального матеріалу.

Існують основні вимоги до програмного та апаратного забезпечення для проектування та розробки програмного забезпечення ЕТМ.

1. Вимоги до функціональних характеристик.

Програмний засіб «Програмування математичних завдань» повністю забезпечує адміністрування системи, авторизацію користувачів, створення авторських електронних освітніх ресурсів (ЕОР), навчальних груп, управління навчальним процесом, збереження та статистичну обробку результатів навчання.

2. Вимоги до окремих модулів.

Програмний засіб «Програмування математичних завдань» розроблено в архітектурі клієнт-сервер з використанням технології об'єктно-орієнтованого проектування і складається з окремих програмних модулів.

Програмний модуль – система авторизації та безпеки – забезпечує реєстрацію користувачів, надання їм прав доступу до ЕТМ із призначенням ролей та прав адміністратора, репетитора та студента, захист системних даних від несанкціонованого доступу.

Програмний продукт – база даних Microsoft SQL Server – забезпечує зберігання даних про користувачів, навчальні групи, авторські ЕЕР, результати навчального процесу, а також усі інші необхідні документи системи.

Програмний модуль – спеціалізований rich-редактор для розробки EER – забезпечує розробку авторських математичних завдань, які складаються з постановки задачі, інструкції до розв'язання, алгоритму розв'язання задачі у вигляді програмного коду.

Програмний модуль – бібліотека математичних завдань – забезпечує зберігання авторських ЕНВ у базі даних, зберігання інформації про автора та використання їх у завданнях навчального процесу.

Програмний модуль – система управління навчальним процесом – забезпечує створення навчальних груп, створення індивідуального навчального завдання з використанням бібліотеки авторського ЕОР, виконання навчального завдання студентом у групі, збереження результати навчання кожного учня.

Програмний модуль – автоматизована система перевірки та верифікації нових авторських EER – забезпечує перевірку правильності поданих нових авторських EER перед збереженням їх у бібліотеці математичних завдань.

Вимоги до програмного забезпечення EPE використовуються при розробці програмного забезпечення для веб-додатку «Програмування математичних завдань».

На рис. 1 показана схема архітектури програмного забезпечення EER.

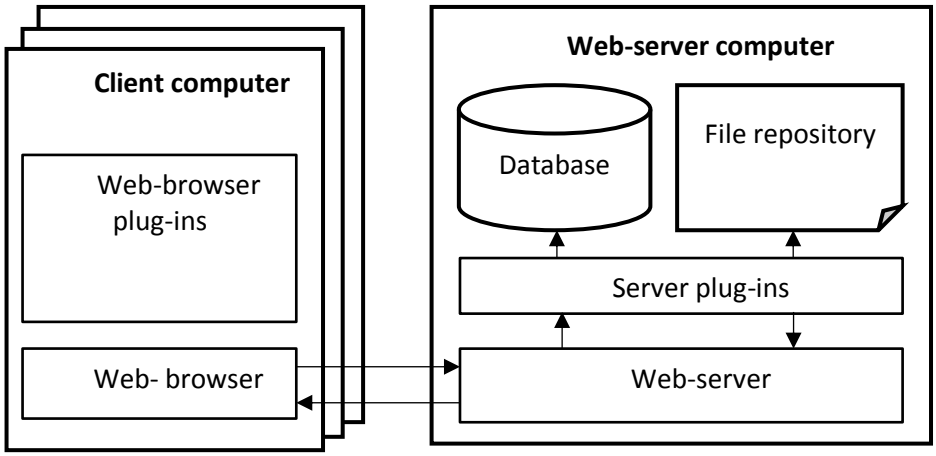


Рис. 1. Архітектура Web-додатка «Програмування математичних задач»

Плагіни для браузера забезпечують керування вмістом сторінок сайту у веб-браузері. Ці програмні модулі включають:

- Підручник у гіпертекстовому форматі.
- Багатий редактор для введення та редагування умов математичних задач, інструкцій до розв'язування завдань та алгоритмів у вигляді програмного коду.

Серверні плагіни обробляють клієнтські запити та взаємодію з базою даних і сховищем файлів. Ці програмні модулі включають:

- Парсер обробки клієнтських запитів.
- Модуль для формування запитів до бази даних, обробки отриманих даних з бази.
- Модуль управління даними зберігання файлів.
- Модуль перевірки коду алгоритму розв'язування математичних завдань, створеного користувачем.

Програмне забезпечення веб-сервера обробляє запити клієнта HTTP і генерує дані для інтерфейсу користувача.

Користувач електронного навчального посібника для роботи з ним використовує веб-браузер, доступ до якого здійснюється через модуль авторизації. Плагіни веб-браузера реалізовані в JavaScript за допомогою бібліотеки Vue JS. Як формат обміну текстовими даними використовувалися JSON (об’єктна нотація JavaScript). На рис. 2 показано принципову схему домашньої сторінки ЕТМ.

Вибір мови навчання визначає мову програмного інтерфейсу, в тому числі підручника та редактора завдань з математики.

Вибір мови програмування визначає мову програмного коду в алгоритмах розв'язування математичних завдань. Натиснувши посилання «Підручник», користувач переходить на сторінку відображення EER (рис. 3).

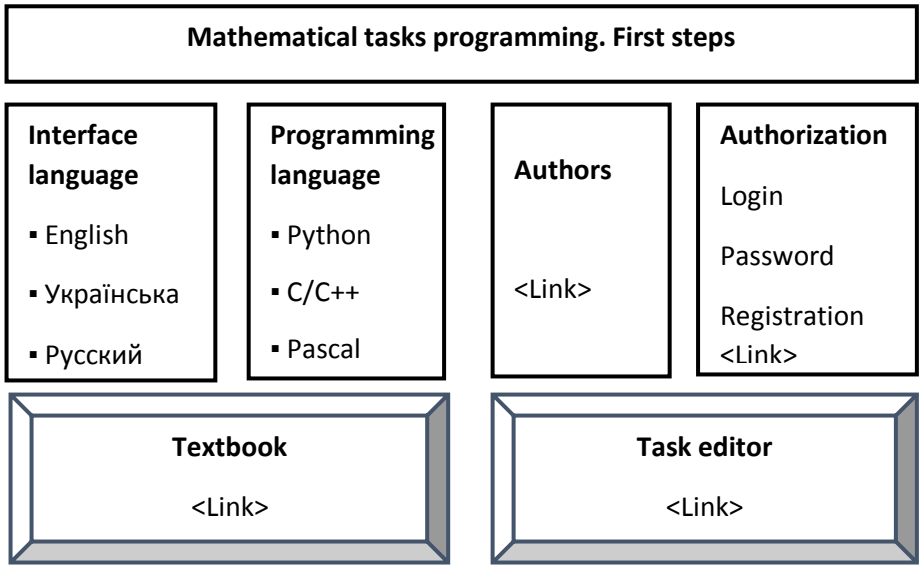


Рис. 2. Принципова схема домашньої сторінки ЕТМ

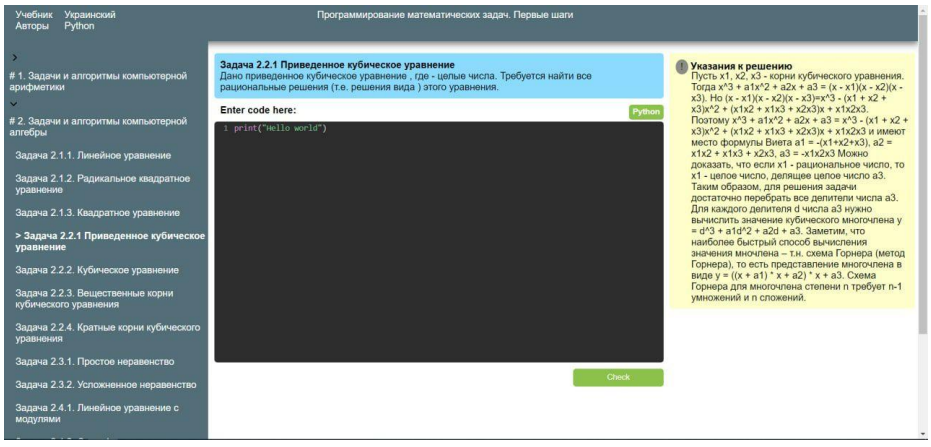


Рис. 3. Сторінка електронного підручника

Інтерфейс EER дозволяє користувачеві вибрати математичну задачу для її алгоритмічного вирішення у форматі програмного коду обраною мовою програмування. Користувач має можливість самостійно розв'язати обрану задачу з подальшою перевіркою правильності рішення. Також користувач може викликати авторське рішення задачі та порівняти його з власним рішенням.

За посиланням «Редактор завдань» викладач має можливість розробити власний авторський EEP у шаблоні назви та формулювання математичної задачі, інструкції до розв'язування задач та алгоритмів розв'язування задач у вигляді програмного коду. У цьому випадку використовується спеціалізований багатий редактор, адаптований до синтаксису мов програмування Pascal, C/C++, Python.

Експертний метод застосовувався для оцінки перспектив Web-додатку з використанням «Програмування математичних завдань» у навчальному процесі. Опитано 12 досвідчених вчителів математики загальноосвітніх шкіл м. Херсона. Вони відповіли на анкету. Для оцінювання було обрано п'ятибальну систему Лайкерта. У таблиці 1 наведено результати оцінки EEP за показниками її використання.

Таблиця 1. Експертна оцінка якості та перспективи використання в навчальному процесі Web-додатку «Програмування математичних завдань».

Таблиця 1

Експертна оцінка якості та перспектив використання Web-додатку «Програмування математичних завдань» у навчальному процесі.

№	Варіанти оцінки	Оцінка
1	Відповідність програмі навчання	4.1
2	Наукова обґрунтованість використання освітніх ресурсів	4.4

3	Дотримання єдиної методології	3.8
4	Доступність використання освітніх ресурсів	2.9
5	Оптимальність технологічних якостей навчального ресурсу	4.0
6	Важливість інтерактивності освітніх ресурсів	4.2
7	Простота використання освітніх ресурсів	3.3
8	Мотивація учнів	4.5
9	Використання в класі	3.6
10	Використання в самостійній роботі	4.7
11	Інтерес вчителів у використанні освітніх ресурсів	3.9
12	Планування використовувати такий освітній ресурс	3.7

Експертна оцінка якості та перспектив використання освітніх ресурсів вважається досить достовірною при хорошому узгодженні експертних оцінок. Тому статистична обробка результатів експертних оцінок включала аналіз консенсусу експертів. Для оцінки ступеня згоди експертів щодо варіантів оцінки використовувався метод конкордації. Гіпотезу про узгодженість експертних оцінок підтвердив критерій Пірсона.

Аналіз навчальної літератури показав, що тематичних збірників завдань із такими характеристиками практично немає:

- Збірник завдань представлено як Інтернет-ресурс;
- Збірник завдань присвячений програмуванню математичних завдань і є елементарним вступом до комп'ютерної алгебри;
- Завдання основних розділів збірника представлені так:
- Завдання – Інструкція до розв’язання – Вихідний код програми рішення;
- Збірник завдань підтримує найпоширеніші мови програмування в спортивному програмуванні Pascal, C/C++, Python.

- У збірці завдань використовуються три мови інтерфейсу користувача: українська, англійська, російська.

Розроблено методичні та програмно-технічні вимоги до програмного забезпечення ЕТМ. Програмне забезпечення ЕТМ є веб-додатком і побудовано на архітектурі клієнт-сервер.

З нашої точки зору, навчально-методичні засоби такого типу стануть у пригоді в навчальному процесі шкіл та університетів, що працюють відповідно до концепції STEM-освіти. [3]

2.2. Модель запитів до API

Реєстрація

Роут: /register - POST

Користувач передає ПІБ, ім’я користувача, пароль та електрону пошту, ці дані записуються до бази даних і далі відправляється п’яти значний код підтвердження на пошту. Повертає згенерований id користувача.

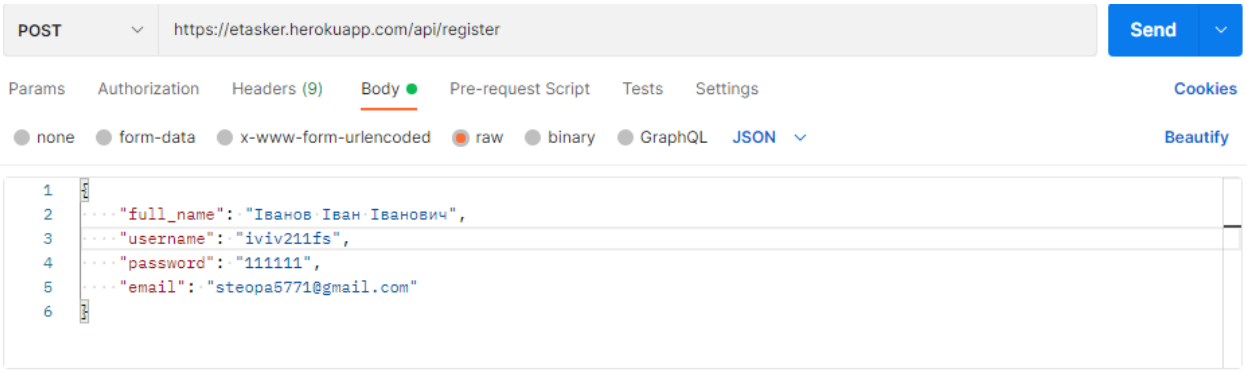


Рис. 4. Модель запиту для реєстрації

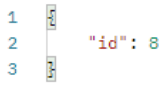


Рис. 5. Відповідь сервера про успішну реєстрацію

Підтвердження електронної пошти

Роут: /email/confirm - POST

Приймає згенерований id користувача і код, який був відправлений на електрону пошту. Повертає id, ім’я користувача та id ролі користувача, при реєстрації встановлюється роль «студент».

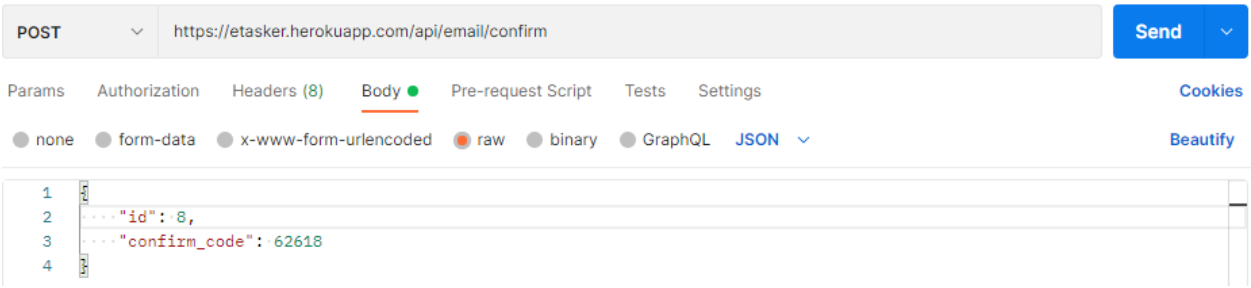


Рис. 6. Модель запиту для підтвердження e-mail

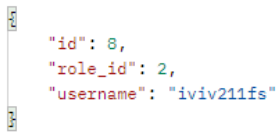


Рис. 7. Відповідь сервера про успішно проведену операцію

Повторне відправлення коду

Роут: /email/resend - POST

Приймає id користувача, генерує новий код та відправляє його на пошту, яка була вказана при реєстрації. Повертає статус операції.

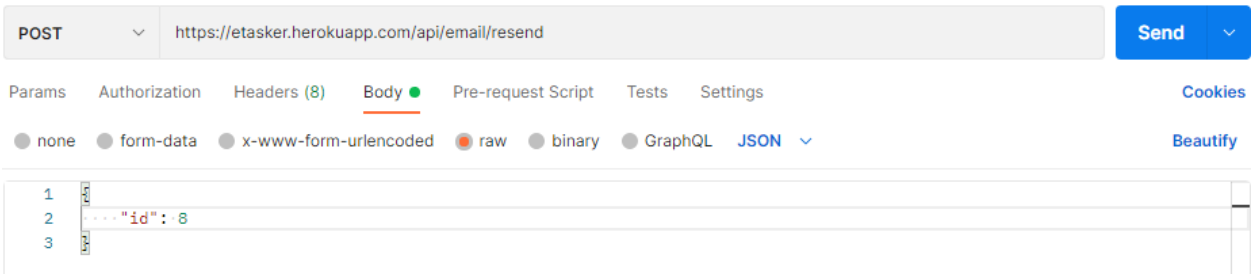


Рис. 8. Модель запиту для повторного відправлення коду підтвердження

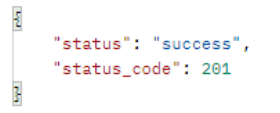


Рис. 9. Відповідь сервера про успішне відправлення

Авторизація користувача

Роут: /login - POST

Приймає ім’я користувача і пароль, порівнює дані з бази даних. Якщо дані правильні, повертає id, ім’я користувача та id його ролі.

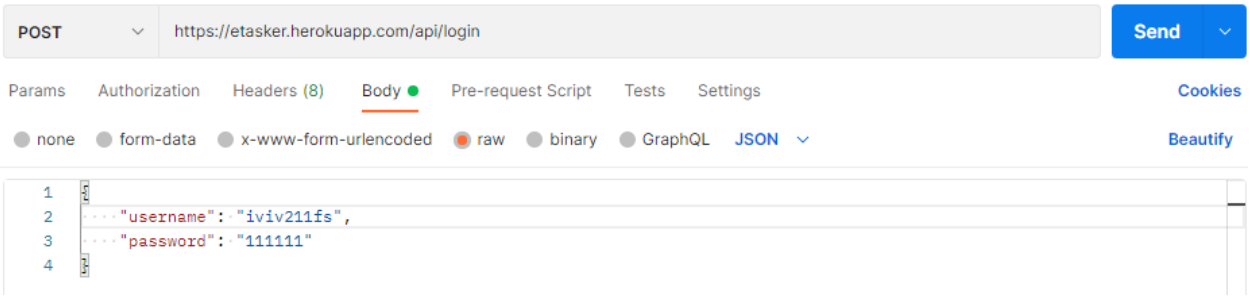


Рис. 10. Модель запиту для авторизації

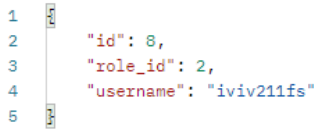


Рис. 11. Успішна авторизація

Відновлення пароля

Роут: /password/recover - POST

Приймає електрону пошту та посилання з клієнтської частини типу https://example.com/recover, яке приймає параметр /id. Отримується з бази даних id користувача, якому належить переказаний e-mail, і відправляється на пошту посилання https://example.com/recover/id. Повертає статус операції.

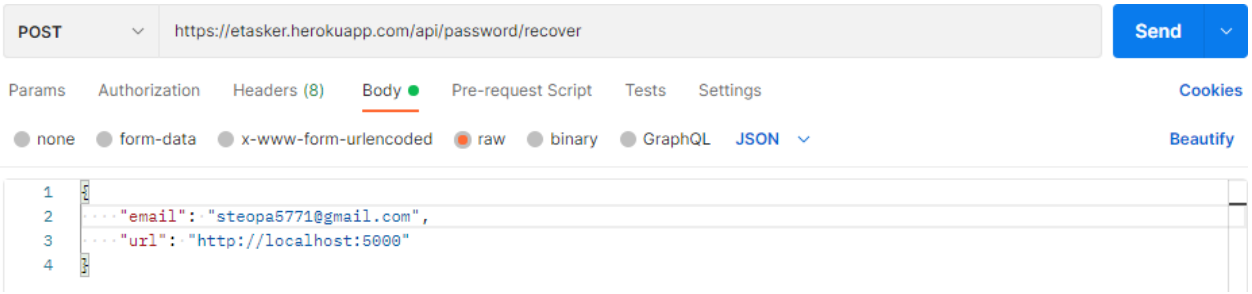


Рис. 12. Модель запиту для відправлення посилання для відновлення пароля

Роут: /password/recover - PUT

Коли користувач перейшов за посиланням, йому відкривається форма для введення нового пароля. Після введення пароля відправляється запит, куди передається id користувача та новий пароль. Повертає статус операції.

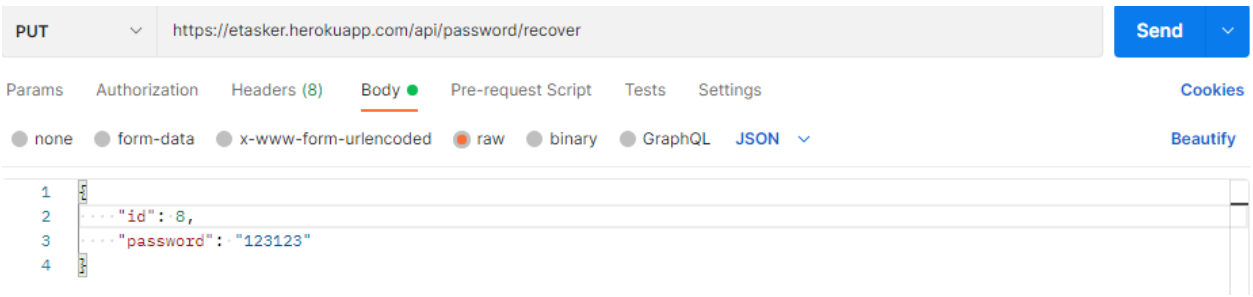


Рис. 13. Модель запиту для відновлення пароля

Зміна паролю

Роут: /password/change - PUT

Приймає id користувача, старий пароль і новий. Якщо поточний пароль із бази даних збігається зі старим паролем, який було передано, замінює пароль в базі даних на новий. Повертає статус операції.

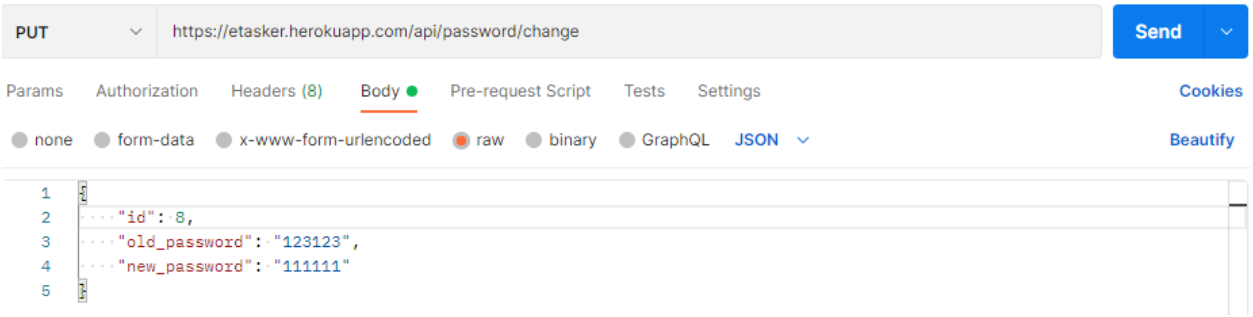


Рис. 14. Модель запиту для зміни пароля

Отримання ролі за її id

Роут: /role/{id} - GET

Приймає id ролі та повертає її id і назву

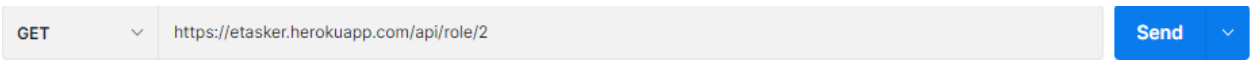


Рис. 15. Модель запиту для отримання ролі

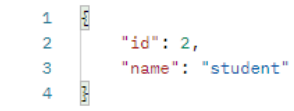


Рис. 16. Відповідь сервера з необхідною роллю

Створення нової ролі

Роут: /role - POST

Приймає назву, створює новий запис у базі даних та повертає згенерований id та назву.

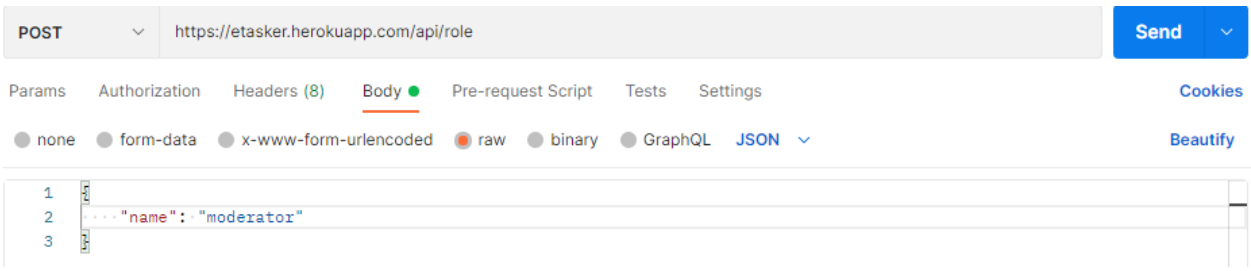


Рис. 17. Модель запиту для створення нової ролі

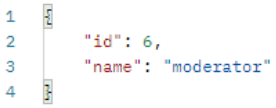


Рис. 18. Відповідь зі створеною роллю

Видалення ролі

Роут: /role - DELETE

Приймає id ролі та видаляє її дані з бази даних. Повертає статус операції.

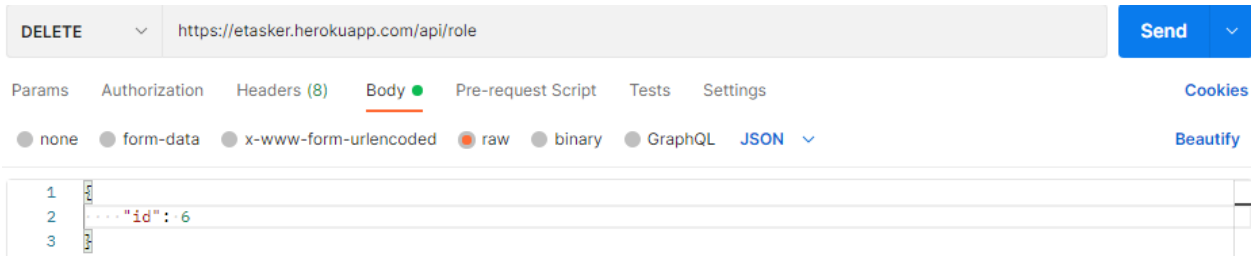


Рис. 19. Модель запиту для видалення ролі

Список ролей

Роут: /role/list

Повертає масив об’єктів - id та назва ролі.

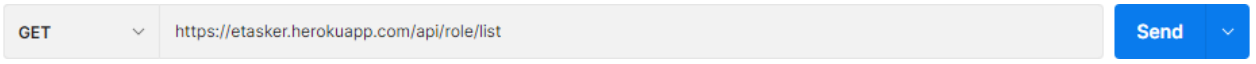


Рис. 20. Модель запиту для отримання списку ролей

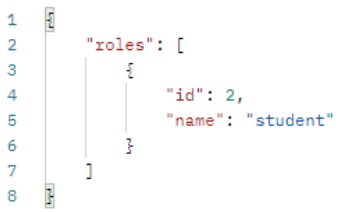


Рис. 21. Відповідь сервера зі списком ролей

Отримання даних конкретного розділу

Роут: /chapter/{id} - GET

Приймає id розділу та повертає заголовок на трьох мовах: російська, українська та англійська, автора, дату створення та редагування і дані першого підрозділу (задачі).


```
{
    "author": "rcher",
    "created": "29-01-2022",
    "id": 18,
    "subchapter_created": null,
    "subchapter_id": null,
    "subchapter_solution_suggestion": null,
    "subchapter_solution_suggestion_title": null,
    "subchapter_task": null,
    "subchapter_title": null,
    "subchapter_updated": null,
    "title": {
        "en": "Chapter 123",
        "ru": "Глава 123",
        "ua": "Розділ 123"
    },
    "updated": null
```


Рис. 22. Відповідь сервера з необхідним розділом

Створення нового розділу

Роут: /chapter - POST

Приймає заголовок та id автора і створює новий розділ. Повертає створений розділ і порожні дані про підрозділ.

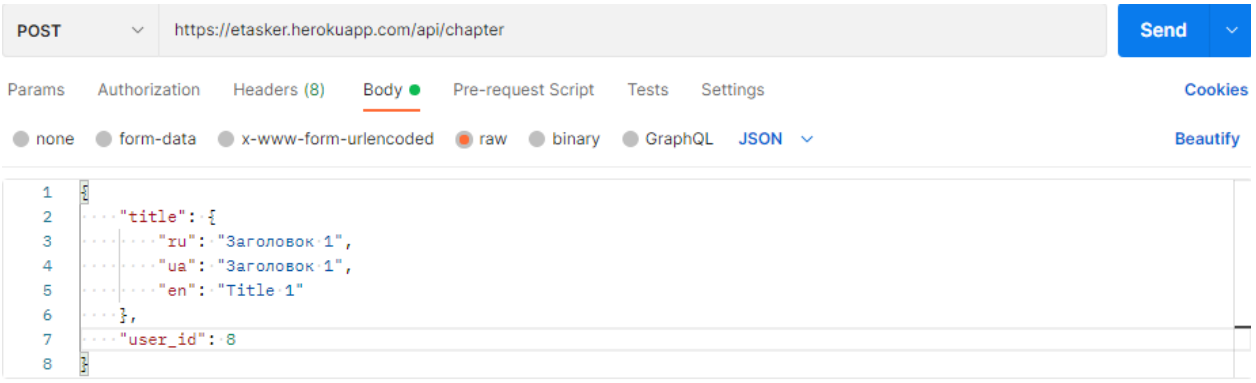


Рис. 23. Модель запиту для створення нового розділу

Редагування існуючого розділу

Роут: /chapter - PUT

Приймає id розділу та новий заголовок. Повертає відредагований розділ.

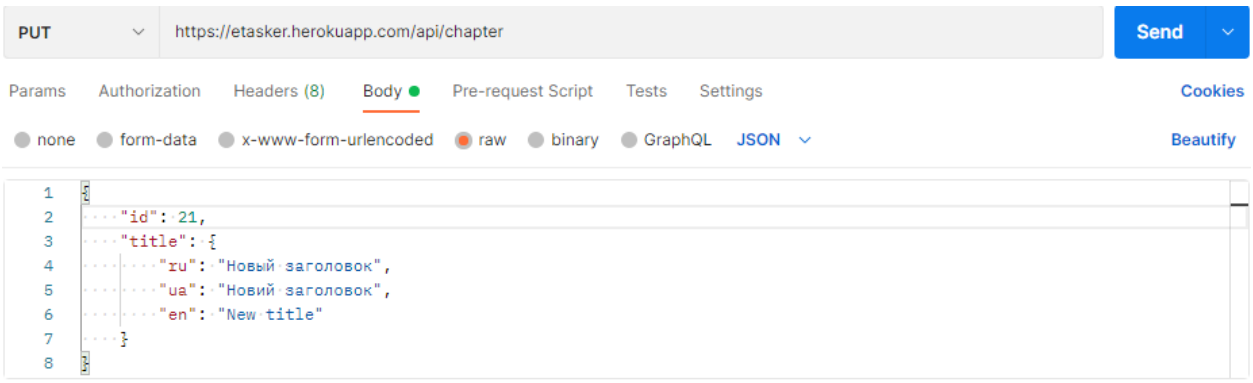


Рис. 24. Модель запиту для редагування розділу

Видалення розділу

Роут: /chapter/{id} - DELETE

Приймає id розділу і видаляє всі його дані. Повертає статус операції.

Отримання списку всіх розділів

Роут: /chapter/list - GET

Повертає масив об’єктів: id розділу, заголовок на трьох мовах, автора, дату створення та останнього редагування.

Рис. 25. Відповідь сервера зі списком усіх існуючих розділів

Отримання конкретного підрозділу (задачі)

Роут: /subchapter/{id} - GET

Приймає id підрозділу. Повертає id підрозділу, id розділу, до якого задача належить, заголовок, текст задачі, заголовок вказівок до розв’язання (може бути порожнім), текст вказівок до розв’язання, дату створення і редагування. Всі текстові елементи повертаються на трьох мовах.

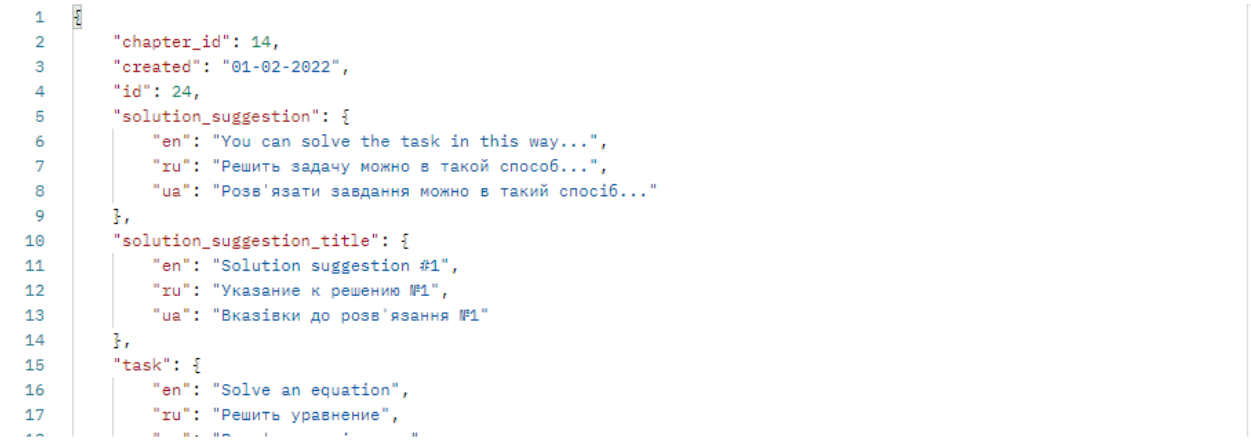


Рис. 26. Відповідь сервера з отриманим підрозділом

Створення підрозділу

Роут: /subchapter - POST

Приймає id розділу, до якого задача належить, заголовок, текст задачі, заголовок вказівок до розв’язання (може бути порожнім), текст вказівок до розв’язання, еталон коду на трьох мовах програмування (Python, Pascal, C) та очікуваний результат. Всі текстові елементи передаються на трьох мовах. Повертає створений підрозділ.

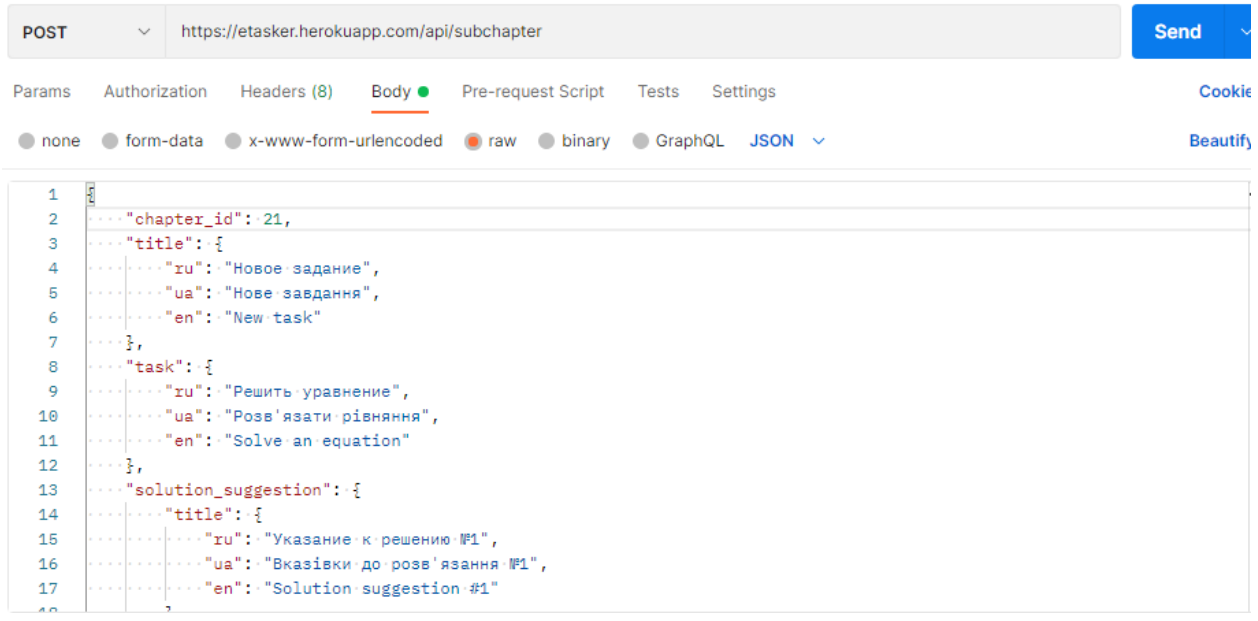


Рис. 27. Модель запиту для створення нового підрозділу

Редагування підрозділу

Роут: /subchapter - PUT

Приймає ті ж самі параметри, що при створенні підрозділу, але всі параметри, крім id підрозділу, необов’язкові. Повертає відредагований підрозділ.

Видалення підрозділу

Роут: /subchapter/{id} - DELETE

Приймає id підрозділу та видаляє всі його дані з бази даних.

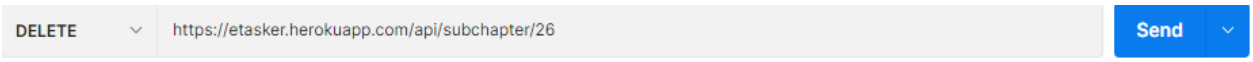


Рис. 28. Модель запиту для видалення підрозділу

Отримання списку підрозділів, які належать конкретному розділу

Роут: /subchapter/list/{chapter_id} - GET

Приймає id розділу та повертає масив підрозділів, які належать одному розділу.


```
{
    "subchapters": [
        {
            "chapter_id": 14,
            "created": "01-02-2022",
            "id": 24,
            "solution_suggestion": {
                "en": "You can solve the task in this way...",
                "ru": "Решить задачу можно в такой способ...",
                "ua": "Розв'язати завдання можно в такий спосіб..."
            },
            "solution_suggestion_title": {
                "en": "Solution suggestion #1",
                "ru": "Указание к решению №1",
                "ua": "Вказівки до розв'язання №1"
            },
            "task": {
```


Рис. 29. Відповідь сервера зі списком підрозділів із конкретного розділу

Отримання списку усіх існуючих підрозділів

Роут: /subchapter/list/all - GET

Повертає масив усіх існуючих підрозділів із усіх розділів.

```
        },
        "title": {
            "en": "New task",
            "ru": "Новое задание",
            "ua": "Нове завдання"
        },
        "updated": null
    },
    {
        "chapter_id": 14,
        "created": "01-02-2022",
        "id": 25,
        "solution_suggestion": {
            "en": "You can solve the task in this way... 1",
            "ru": "Решить задачу можно в такой способ... 1",
            "ua": "Розв'язати завдання можно в такий спосіб... 1"
        },
```


Рис. 30. Відповідь сервера зі списком усіх існуючих підрозділів

Встановлення нової ролі користувача

Роут: /account/role - POST

Встановлює нову роль користувача. Приймає id користувача та необхідної ролі і повертає дані користувача, ті ж, що при авторизації.

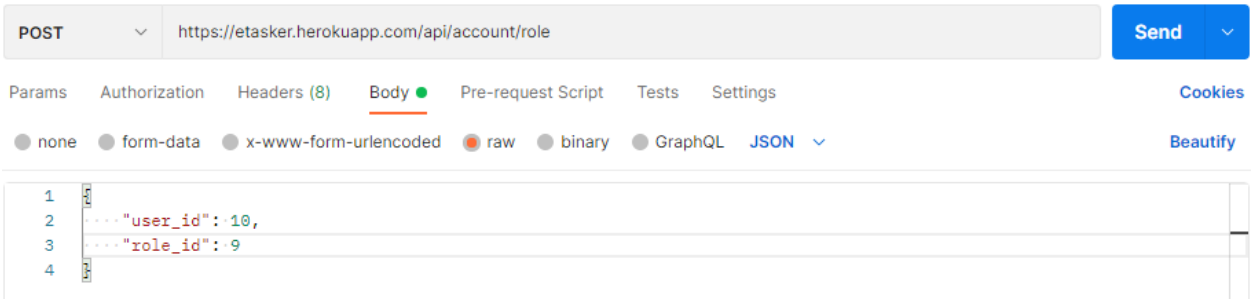


Рис. 31. Модель запиту для встановлення нової ролі користувача

Отримання даних відправленого завдання на перевірку

Роут: /code/submit/{subchapter_id} - GET

Повертає результат відправленого коду конкретного користувача для конкретного завдання (subchapter_id). Повертає id сабміту, ім’я користувача, його id, id підрозділу, його заголовок на трьох мовах, відправлений код, еталон коду, результат, очікуваний результат, мову програмування та дату відправлення.

Рис. 32. Відповідь сервера з даними сабміту

Отримання даних всіх відправлених завдань для вчителя (перевіряючого)

Роут: /code/submit/teacher/{teacher_id} - GET

Повертає останній результат усіх відправлених завдань, які були відправлені будь-яким користувачем для будь-якого завдання. teacher_id необхідний для того, щоб перевірити користувача на рівень доступу до поточної функції та повернути всі оцінки до кожного сабміту, які були виставлені цим вчителем.

Отримання даних всі відправлених завдань конкретного учня

Роут: /code/submit/student/{student_id} - GET

Повертає останній результат усіх відправлених завдань, які були відправлені конкретним учнем для будь-якого завдання.

Відправлення завдання на перевірку

Роут: /code/submit - POST

Відправляє код на компіляцію. Приймає id підрозділу, id користувача, код та мову програмування. Повертає дані цього сабміту разом із результатом компіляції коду.

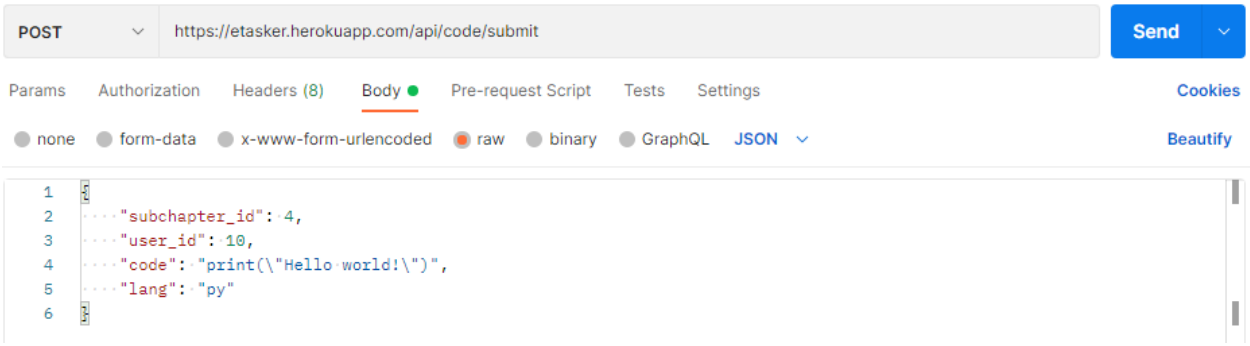


Рис. 33. Модель даних відправлення нового сабміту

Отримання оцінки за виконане завдання

Роут: /score/{id} - GET

Повертає оцінку за її id. Повертає id вчителя, його ім'я користувача та повне ім'я, id учня, ім'я користувача та повне ім'я, id сабміту коду, оцінку, коментарі до коду, дату створення та останнього редагування оцінки.

Рис. 34. Відповідь сервера з оцінкою

Виставлення оцінки

Роут: /score - POST

Виставлення оцінки конкретному сабміту коду. Приймає id вчителя, id сабміту та оцінку, повертає дані оцінки.

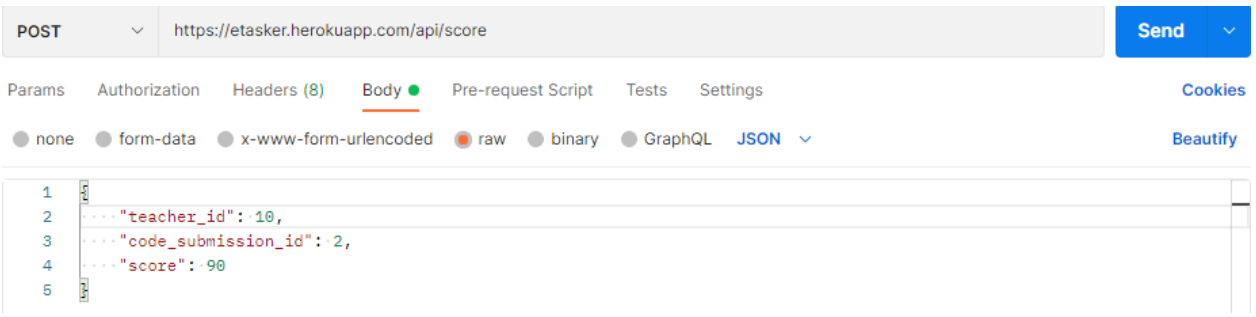


Рис. 35. Модель запиту до створення оцінки

Зміна оцінки

Роут: /score - PUT

Зміна оцінки. Приймає id оцінки, нову оцінку та id користувача, який намагається змінити оцінку, щоб порівняти його з id користувача, який виставив оцінку.

Видалення оцінки

Роут: /score/{id} - DELETE

Видаляє оцінку. Приймає id користувача, який намагається видалити оцінку, щоб порівняти з id користувача, який виставив оцінку.

Отримання списку усіх оцінок, які були виставлені конкретному сабміту

Роут: /score/list/{code_submission_id} - GET

Повертає усі оцінки, виставлені одному конкретному сабміту.

Отримання усіх оцінок, виставлених одному учню

Роут: /score/list/student/{student_id} - GET

Приймає id учня та повертає усі його оцінки.


```
{
    "scores": [
        {
            "code_submission_id": 1,
            "comments": [
                {
                    "created": "11-04-2022",
                    "id": 1,
                    "line": 1,
                    "text": "New comment 1",
                    "updated": "11-04-2022"
                },
                {
                    "created": "11-04-2022",
                    "id": 2,
                    "line": 2,
                    "text": "Comment 3",
                    "updated": "11-04-2022"
                }
```


Рис. 36. Фрагмент відповіді сервера зі списком оцінок

Отримання коментарів до конкретної оцінки

Роут: /score/comment/{score_id} - GET

Повертає список коментарів до коду учня, прив’язані до конкретної оцінки.

```
{
    "comments": [
        {
            "created": "11-04-2022",
            "id": 1,
            "line": 1,
            "text": "New comment 1",
            "updated": "11-04-2022"
        },
        {
            "created": "11-04-2022",
            "id": 2,
            "line": 2,
            "text": "Comment 3",
            "updated": "11-04-2022"
        }
    ],
    "score_id": 2
}
```


Рис. 37. Відповідь сервера з коментарями

Додавання коментарів

Роут: /score/comment/{score_id} - POST

Додає нові коментарі до оцінки. Приймає масив коментарів, який містить об’єкт із номером рядка та текстом коментаря. Повертає новоутворений масив коментарів.

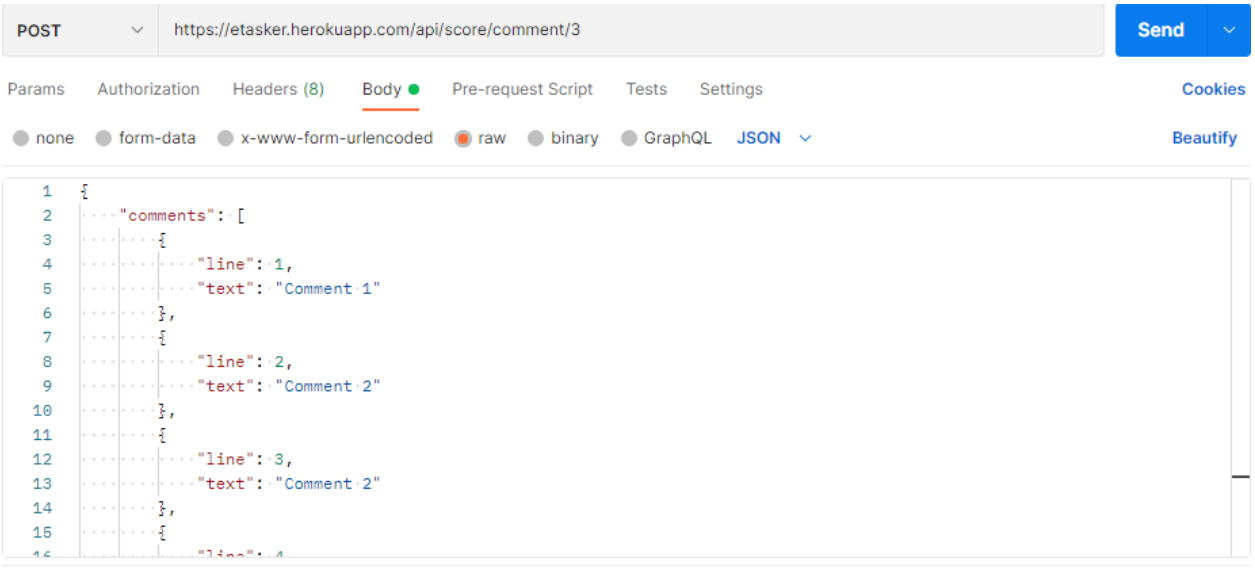


Рис. 38. Модель запиту для створення коментаря

Редагування коментарів

Роут: /score/comment/{score_id} - PUT

Змінює необхідні коментарі.

Видалення коментарів

Роут: /score/comment/{score_id} - DELETE

Видаляє коментарі. Приймає масив коментарів, який містить об’єкти з їх id. Цей параметр може бути порожнім, якщо потрібно видалити абсолютно всі коментарі, які були додані до поточної оцінки.

2.3. Конструювання веб-сервісу «програмування математичних задач»

Вся серверна частина проекту була написана за допомогою мови програмування python, мікрофреймворку flask і flask_restful та бази даних PostgreSQL. Всі роути мають вигляд класу, який наслідується від головного класу Resource із пакету flask_restful, та під’єднується у головному файлі.

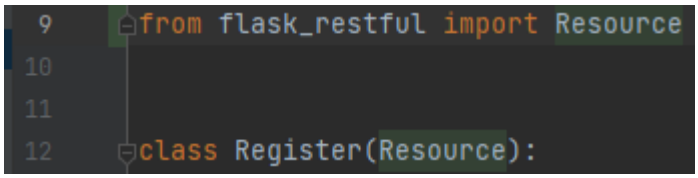


Рис. 39. Приклад створення роуту

У кожному такому класі є функції з такою ж назвою, як http метод, наприклад get, post, put і т.д. У кожному запиті на початку відбувається перевірка щодо наявності усіх необхідних параметрів, наприклад у разі реєстрації перевіряється чиє наявні поля ім’я користувача, пароль, повне ім’я, електрона пошта і т.д. Якщо не вистачає хоча б одного обов’язкового поля, сервер поверне повідомлення про невистачаючі дані.

```python
class Register(Resource):
    def post(self):
        json = request.json

        error = []

        if "full_name" not in json:
            error.append("Full name is empty")
        if "username" not in json:
            error.append("Username is empty")
        if "username" in json and len(json["username"]) < 4:
            error.append("Username must contain at least 4 digits")
        if "password" not in json:
            error.append("Password is not empty")
        if "password" in json and len(json["password"]) < 6:
            error.append("Password must contain at least 6 characters")
        if "email" not in json:
            error.append("Email is empty")
        if "email" in json and len(str.split(json["email"], "@")) < 2:
            error.append("Wrong email")

        if len(error) > 0:
            return create_response(error, 400)
```

Рис. 40. Приклад перевірки наявності необхідних даних

Далі виконується перевірка на наявність такого ім’я користувача або електронної пошти у базі даних, якщо є співпадіння, повертається повідомлення про те, що користувач з таким ім’ям користувача або емейлом вже зареєстрований. Якщо жодних помилок або співпадінь не знайдено, генерується п’ятизначний код, який буде відправлений на вказану електрону пошту, користувачу встановлюється його роль із нулевим рівнем доступу, всі передані дані записуються в базу даних та повертається згенерований id нового користувача.

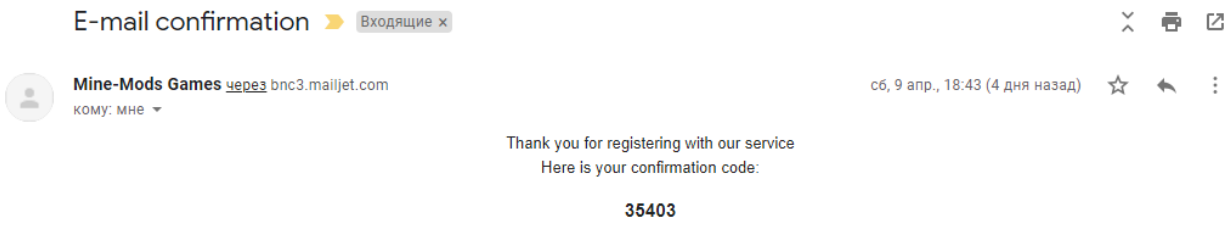


Рис. 41. Відправлений код на пошту

Якщо повідомлення не прийшло на пошту, його можна перевідправити. Приймається тільки згенерований id користувача, генерується новий код та відправляється. Якщо код прийшов, його необхідно ввести, щоб підтвердити емейл. На сервері перевіряється наявність id користувача та

коду, перевіряється на співпадіння з базой даних, отримується запис, в якому є переданий id та код, якщо співпадінь немає, повертається відповідне повідомлення. Якщо співпадіння є, в тому записі встановлюється, що електрона пошта підтверджена, та повертаються дані користувача.

```
    record = select_one("select id, username, role_id, confirm_code from users "
                        "where id=%s and confirm_code=%s", (id, confirm_code))
    if record is None:
        return create_response("Invalid code", 400)

    if update("update users set email_confirmed=true, confirm_code=NULL, date=%s where id=%s",
              (date.today().strftime("%d-%m-%Y"), id)) == 0:
        return create_response("Something went wrong", 400)

    response = {
        "id": id,
        "username": record[1],
        "role_id": record[2]
    }

    return create_response(response, 200)
```

Рис. 42. Підтвердження електронної пошти

Коли електрона пошта підтверджена, можна авторизуватися за допомогою логіну або пошти та паролю. Бувають випадки, коли забув пароль, його можна відновити. Для цього на початку передається електрона пошта користувача та посилання на сторінку сайту, де є можливість встановити новий пароль, яка приймає id користувача. Із бази даних отримується id за яким закріплений даний емейл та відправляється посилання на пошту.

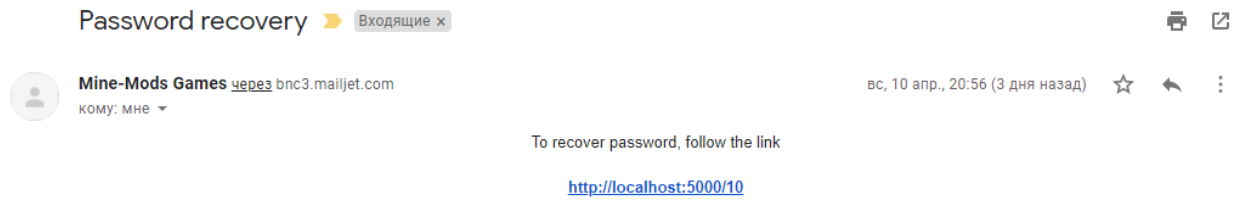
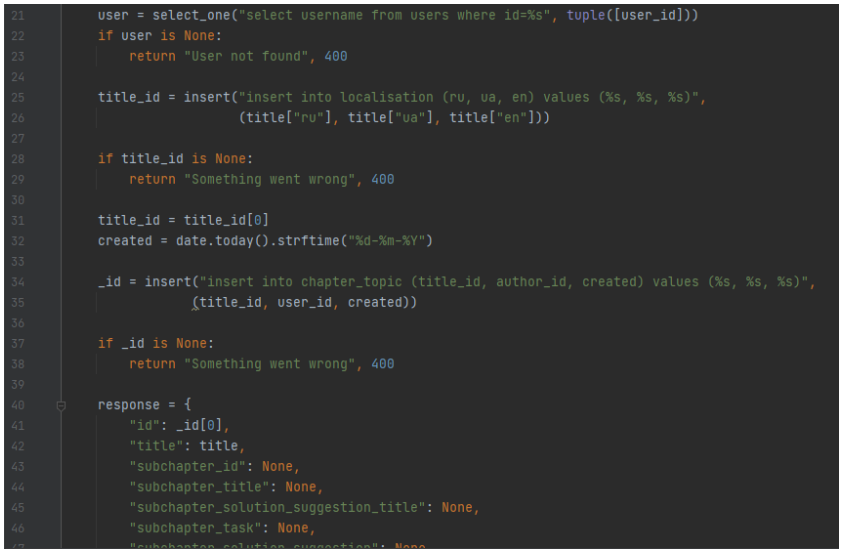


Рис. 43. Відправлене повідомлення з посиланням

Коли користувач перейшов за посиланням, ввів новий пароль, він відправляється на сервер та записується до бази даних.

Користувач з роллю вчитель або вище може створити та редагувати розділи. Для цього йому потрібно передати заголовок на трьох мовах: російська, українська та англійська, і id користувача, який є автором цього розділу. Для редагування теж потрібно передавати заголовок і ще id потрібного розділу. Щоб видалити розділ, необхідно передати його id, видалиться весь розділ, включаючи усі його підрозділи (задачі). Отримати інформацію про розділ може будь-який користувач, для цього треба передати id розділу, поверне інформацію про розділ та його перший підрозділ. Ще можна отримати список усіх підрозділів.



```
    user = select_one("select username from users where id=%s", tuple([user_id]))
    if user is None:
        return "User not found", 400

    title_id = insert("insert into localisation (ru, ua, en) values (%s, %s, %s)",
                      (title["ru"], title["ua"], title["en"]))

    if title_id is None:
        return "Something went wrong", 400

    title_id = title_id[0]
    created = date.today().strftime("%d-%m-%Y")

    _id = insert("insert into chapter_topic (title_id, author_id, created) values (%s, %s, %s)",
                 (title_id, user_id, created))

    if _id is None:
        return "Something went wrong", 400

    response = {
        "id": _id[0],
        "title": title,
        "subchapter_id": None,
        "subchapter_title": None,
        "subchapter_solution_suggestion_title": None,
        "subchapter_task": None,
```

Рис. 44. Фрагмент коду створення розділу

Коли користувач створив розділ, можна далі створити підрозділ, іншими словами – додати саму задачу. Для цього необхідно передати id розділу, заголовок, текст задачі, вказівки до розв’язання: заголовок є необов’язковим і їх вміст треба передавати на трьох мовах, як і всі інші подібні текстові параметри, еталон коду – код, який на думку автора є правильним розв’язанням завдання та очікуваний результат виконання коду. В першу чергу перевіряється, чи існує розділ із таким id, якщо ні, повертає відповідне повідомлення. Всі параметри, які мають 3 мовні версії, записуються до таблиці localisation (локалізація) та зберігаються їх id, які далі записуються в іншій таблиці task, solution_suggestion. Еталон коду зберігається теж в окремій таблиці perfect_code у трьох

мовах програмування: C, Python та Pascal. Всі ці id (посилання) зберігаються у головній таблиці subchapter. Користувач, подібно до розділу, може редагувати завдання, змінюючи будь-які параметри. Може видалити завдання за його id та будь-який користувач може отримати дані окремого завдання, може отримати список усіх підрозділів, які належать окремому розділу, та список абсолютно усіх підрозділів.

id	topic_id	title_id	task_id	solution_suggestion_id	perfect_code_id	predicted_result	created	updated
24	14	121	26	25	24	c=49	01-02-2022	*NULL*
25	14	127	27	26	25	c=49	01-02-2022	01-02-2022

Рис. 45. Таблиця subchapter

Учень може виконати завдання та відправити «сабміт» на сервер для перевірки. Передається id підрозділу, id учня, його код та мова програмування, яка була використана. На початку перевіряється чи існує ця задача та чи існує користувач з таким id. Потім компілюється код учня та повертається результат. Всі ці дані записуються до бази даних та повертається користувачу результат виконання коду.

```
    username = select_one("select username from users where id=%s limit 1", tuple([user_id]))

    result = compile_code(code, lang)
    created = date.today().strftime("%d-%m-%Y")

    id = insert("insert into code_submission (subchapter_id, user_id, code, result, lang, created)"
                " values (%s, %s, %s, %s, %s, %s)",
                (subchapter_id, user_id, code, result, lang, created))

    response = {
        "id": id[0],
        "subchapter_title": None if subchapter_title is None else {
            "ru": subchapter_title[1],
            "ua": subchapter_title[2],
            "en": subchapter_title[3]
        },
        "username": None if username is None else username[0],
        "user_id": user_id,
        "code": code,
        "result": result,
        "lang": lang,
        "created": created
    }
    return 200, response
```

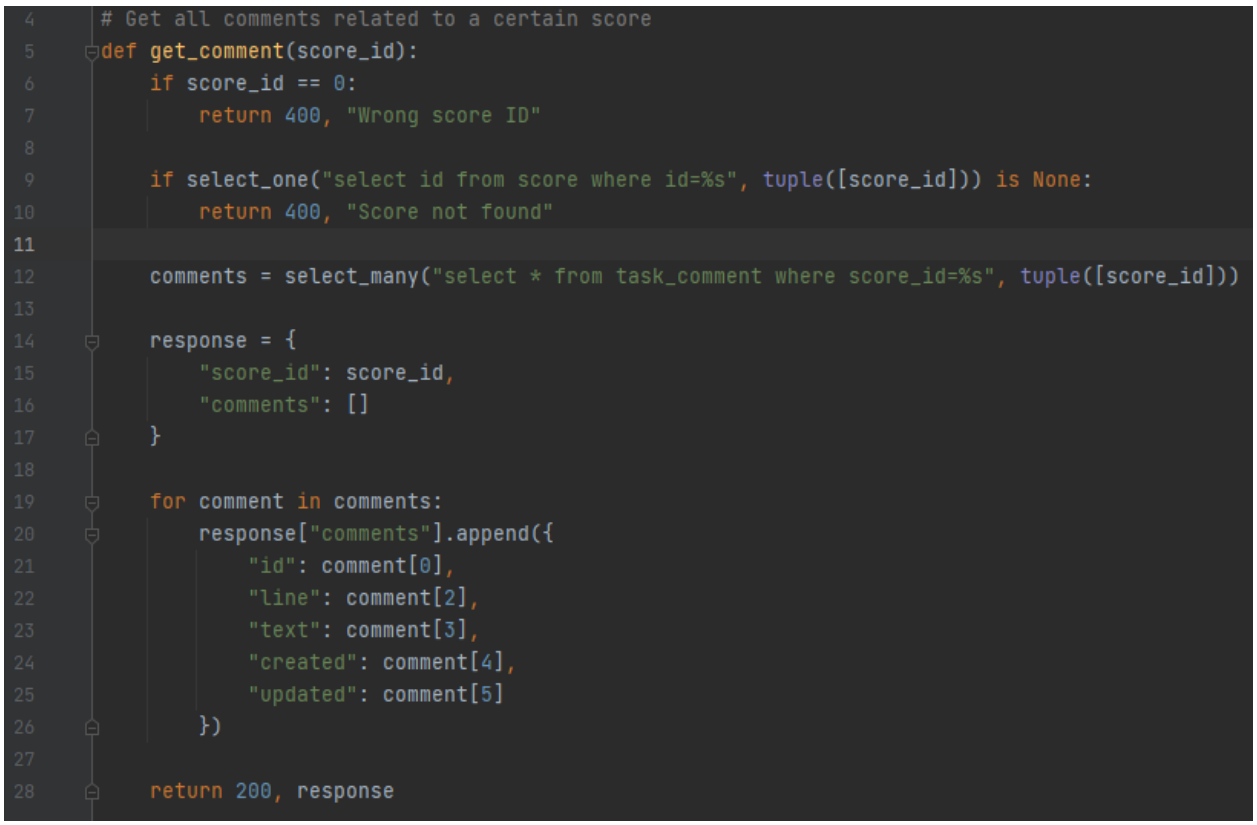
Рис. 46. Фрагмент коду сабміту

Можна отримати останній сабміт користувача, який був відправлений для конкретної задачі конкретним користувачем за їх id. Учень може отримати список усіх своїх сабмітів з оцінками. Будь-який вчитель може отримати усі такі сабміти для всіх завдань зі своєю оцінкою та виставити або змінити оцінку, куди передається id вчителя, id сабміту та оцінку за 100 бальною шкалою. Вчитель може виставити тільки одну оцінку для одного сабміту, якщо вже наявна оцінка від цього вчителя до шуканого сабміту, поверне відповідне повідомлення. Вчитель може видалити свою оцінку. Кожен може отримати її дані. Можна отримати список усіх оцінок, які були виставлені одному сабміту або одному конкретному учню на всі сабміти.

```
25      if select_one("select id from score where teacher_id=%s and code_submission_id=%s",
26                    (teacher_id, code_submission_id)) is not None:
27          return 400, "You can't add another score"
28
29      created = date.today().strftime("%d-%m-%Y")
30
31      if select_one("select id from code_submission where id=%s limit 1",
32                    tuple([code_submission_id])) is None:
33          return 400, "Wrong code submission ID"
34
35      if select_one("select username, full_name from users where id=%s limit 1",
36                    tuple([teacher_id])) is None:
37          return 400, "Teacher not found"
38
39      id = insert("insert into score (teacher_id, code_submission_id, score, created) "
40                  "values (%s, %s, %s, %s)",
41                  (teacher_id, code_submission_id, score, created))
42      return get_score(id[0])
```

Рис. 47. Фрагмент коду виставлення оцінки

До кожної оцінки вчитель може додати коментарі до кожного рядка коду передавши номер рядка та текст коментаря. Вчитель може змінити окремі коментарі передавши їх id та новий текст. Можна видалити коментарі за їх id або абсолютно всі коментарі до оцінки та можна отримати всі коментарі до однієї оцінки за її id.



```python
# Get all comments related to a certain score
def get_comment(score_id):
    if score_id == 0:
        return 400, "Wrong score ID"

    if select_one("select id from score where id=%s", tuple([score_id])) is None:
        return 400, "Score not found"

    comments = select_many("select * from task_comment where score_id=%s", tuple([score_id]))

    response = {
        "score_id": score_id,
        "comments": []
    }

    for comment in comments:
        response["comments"].append({
            "id": comment[0],
            "line": comment[2],
            "text": comment[3],
            "created": comment[4],
            "updated": comment[5]
        })

    return 200, response
```

Рис. 48. Код отримання коментарів

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Learning Python, Fifth Edition by Mark Lutz
2. SQL для простых смертных - Грабер Мартин
3. Разработка веб-приложений с использованием Flask на языке Python - Мигель Гринберг
4. The New And Improved Flask Mega-Tutorial by Miguel Grinberg
5. Flask Framework Cookbook: Over 80 proven recipes and techniques for Python web development with Flask, 2nd Edition
6. PostgreSQL Up and Running (3rd Edition) by Regina Obe and Leo Hsu
7. The Art of PostgreSQL by Dimitri Fontaine
8. About Electronic Textbook “Mathematical Tasks Programming. First Steps” – М. С. Львов, Г. М. Кравцов, Л. С. Шишко, О. О. Гнєдкова, І. Є. Черненко, Є. О. Козловський
9. https://www.valamis.com/hub/what-is-an-lms - LMS
10. https://www.talentlms.com/what-is-an-lms
11. https://www.adamenfroy.com/learning-management-system
12. https://e-student.org/what-is-lms/
13. https://ru.wikipedia.org/wiki/Система_управления_обучением
14. https://flask-restful.readthedocs.io/en/latest/
15. https://pythonbasics.org/what-is-flask-python/
16. https://www.fullstackpython.com/flask.html
17. https://www.python.org/
18. https://www.postgresql.org/
19. https://www.udemy.com/course/rest-api-flask-and-python/?ranMID=39197&ranEAID=R7BSs79ua1Y&ranSiteID=R7BSs79ua1Y-mIVuW70n3f5hcVK2jxeAYA&LSNPUBID=R7BSs79ua1Y&utm_source=aff-campaign&utm_medium=udemyads - REST APIs with Flask and Python
20. https://etasker.herokuapp.com/ - Документація до API