

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО”**

Теплоенергетичний факультет

Кафедра автоматизації проектування енергетичних процесів і систем

До захисту допущено:

Завідувач кафедри

_____ Наталія АУШЕВА

«___» _____ 2022 р.

**Дипломна робота
на здобуття ступеня бакалавра
спеціальності 122 “Комп’ютерні науки”
освітня програма “Комп’ютерний еколого-економічний моніторинг
процесів та систем”
на тему: “Вебдодаток контролю навчального процесу”**

Виконав:

студент IV курсу, групи ТМ-81

Бабій Іван Олександрович

Керівник:

доктор технічних наук, професор

Верлань Андрій Анатолійович



Рецензент:

засновник ТОВ "АРТКАЙ"

кандидат економічних наук, доцент

Чеканов Костянтин Сергійович

Засвідчую, що у цій дипломній роботі
немає запозичень з праць інших авторів без
відповідних посилань.

Студент (-ка) _____

Київ — 2022 року

**Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”**

Факультет теплоенергетичний

Кафедра автоматизації проектування енергетичних процесів і систем

Рівень вищої освіти перший

спеціальність 122 «Комп’ютерні науки»

освітня програма «Комп’ютерний моніторинг та геометричне моделювання процесів і систем»

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____Наталія АУШЕВА

(підпис)

« ____ » _____ 2022р.

ЗАВДАННЯ

на дипломну роботу студенту

Бабію Івану Олександровичу

(прізвище, ім’я, по батькові)

1. Тема роботи Вебдодаток контролю навчального процесу

керівник роботи Верлань Андрій Анатолійович, д.т.н., професор

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

затверджена наказом вищого навчального закладу від «08» червня 2022р. № **965-С**

2. Строк подання студентом роботи 24.05.2021р.

3. Вихідні дані до роботи Автоматизована система контролю навчального процесу.

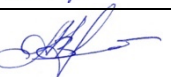
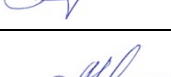
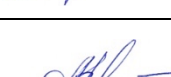
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) розробити вебдодаток для контролю навчального процесу; дати можливість завантажувати завдання з дисциплін; контролювати розклад занять протягом дня та тижня; мати доступ до контактної інформації студентів групи.

5. Перелік ілюстративного матеріалу

Вебдодаток контролю навчального процесу, Опис діаграми прецедентів, Аналіз системи E-campus КПП, оновлений E-campus КПП, основний електронний кампус КПП, Аналіз системи Edu КНТЕУ, Система контролю навчального процесу, Вхід студента у систему, Вхід викладача у систему, Інтерфейс програми для студента, Інтерфейс програми для викладача.

6. Дата видачі завдання «20» _____ лютого _____ 2021 р.

КАЛЕНДАРНИЙ ПЛАН

з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітки
1.	Затвердження теми роботи	25.05.2022	
2.	Вивчення та аналіз задачі	02.04.2022 – 08.04.2022	
3.	Розробка архітектури та загальної структури системи	19.04.2022 – 22.04.2022	
4.	Розробка структур окремих підсистем	16.04.2022 – 22.04.2022	
5.	Програмна реалізація системи	19.04.2022 – 28.04.2022	
6.	Оформлення пояснювальної записки	16.05.2022 – 28.05.2022	
7.	Захист програмного продукту	29.05.2022	
8.	Передзахист	06.06.2022 – 09.06.2022	
9.	Захист	20.06.2022	

Студент

(підпис)

Керівник роботи

(підпис)

Бабій І.О.

(прізвище та ініціали,)

Верлань А.А.

(прізвище та ініціали,)

АНОТАЦІЯ

У даній роботі було описано розробку програмного продукту, яким є вебдодаток контролю навчального процесу. Було описано предметну область продукту, а також проведено аналіз аналогічних рішень, що існують. У роботі вказано засоби розробки та архітектуру, яка використовується у програмному рішенні. На прикладі двох типів користувачів було представлено основні сценарії використання системи.

Вебдодаток складається двох частин. Front-end частина є відображенням даних, що стосуються навчального процесу, з можливістю завантажувати чи відправляти типові документи. Back-end частина відповідає за обробку даних, що містяться в базі та передаються через API у реальному часі.

Метою роботи є підвищення ефективності роботи студента під час дистанційного навчання за рахунок логіки інтерфейсу системи.

Пояснювальна записка складається з шести розділів, висновку, списку використаних джерел, а також містить 47 сторінок, 25 графічні матеріали та 3 додатки. Список використаних джерел складається з 1 бібліографічного видання та 11 електронних ресурсів.

Ключові слова: вебдодаток, навчальний процес, система контролю, розклад занять, дисципліни.

ABSTRACT

This research paper described the development of a software product, which is a web application controlling the learning process. The subject area of the product was described, as well as an analysis of similar solutions that exist. The paper indicated the development tools and the architecture that is used in the software solution. Using two types of users as examples, the main scenarios for using the system were presented. Web application is composed of two parts. Front-end part is the representation of the data, which are related to the learning process, with the possibility to upload or send typical documents. The back-end part is responsible for the data processing, which is contained in the database and is transmitted through an API in real time. The purpose of the work is to increase the efficiency of the student's work during distance learning through the logic of the system's interface. The interpretative note consists of six sections, a conclusion, a list of used sources, and includes 47 pages, 25 graphics, and 3 appendices. The list of used sources consists of 1 bibliography and 11 electronic resources. Keywords: web-application, educational process, control system, class schedule, disciplines.

ЗМІСТ

СПИСОК ТЕРМІНІВ, СКОРОЧЕНЬ ТА ПОЗНАЧЕНЬ	8
ВСТУП	10
1 ПОСТАНОВКА ЗАДАЧІ РОЗРОБКИ ВЕБДОДАТКУ КОНТРОЛЮ НАВЧАЛЬНОГО ПРОЦЕСУ	13
1.1 Опис основних задач програмного продукту	13
1.2 Призначення продукту. Цільова аудиторія	14
1.3 Висновки до розділу	15
2 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ	16
2.1 Аналіз системи E-campus КПІ	16
2.1.1 Оновлений E-campus КПІ	16
2.1.2 Основний електронний кампус КПІ	18
2.2 Аналіз системи Edu КНТЕУ на базі Moodle	19
2.3 Висновки до розділу	21
3 ВИБІР ЗАСОБІВ РОЗРОБКИ	22
3.1 Опис засобів Front-end розробки	22
3.2 Опис засобів Back-end розробки	23
3.3 Вимоги до програмно-апаратної частини	24
3.4 Висновки до розділу	25
4 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ	26
4.1 Формування системи контролю навчального процесу	26
4.2 Опис діаграми прецедентів	32
4.3 Висновок до розділу	34

5 РОБОТА КОРИСТУВАЧА З ПРОГРАМНОЮ СИСТЕМОЮ	35
5.1 Вхід студента в систему	35
5.2 Вхід викладача (адміністратора) в систему	36
5.3 Інтерфейс програми для студента	37
5.4 Інтерфейс програми для викладача (адміністратора)	42
5.5 Висновки до розділу	45
6 ПЕРСПЕКТИВИ РОЗВИТКУ ПРОГРАМНОГО ПРОДУКТУ	46
ВИСНОВКИ	47
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	48
ДОДАТОК А	49
ДОДАТОК Б	51
ДОДАТОК В	62

СПИСОК ТЕРМІНІВ, СКОРОЧЕНЬ ТА ПОЗНАЧЕНЬ

ERP-система (Enterprise Resource Planning System) — комплексна система для автоматизації бізнес-процесів компанії.

Діаграма прецедентів (UseCase Diagram) — діаграма, що відображає відношення між акторами та сценаріями використання системи.

UX (User Experience) — досвід користувача при взаємодії з тією чи іншою системою.

Фреймворк — програмне забезпечення, що дозволяє обробляти інструменти та конфігурації, необхідні для мови програмування, на базі якої вона створена, а також надавати додаткову структуру, функції та оптимізацію для цифрового продукту.

URL — унікальний ідентифікатор, що використовується для визначення місцеперебування ресурсу в мережі Інтернет.

API (Application Programming Interface) — програмний посередник, що дозволяє комунікувати додаткам чи сервісам одне між одним.

PaaS (Platform as a Service) — модель хмарних обчислень, при якому сторонній постачальник надає користувачам апаратні чи програмні інструменти через мережу Інтернет.

Webhook — легкий API, який забезпечує односторонній обмін даними, що ініціюються подіями.

HMAC (Hash-based message authentication code) — метод криптографічної аутентифікації, що використовує хеш-функцію та секретний ключ.

RSA (Rivest-Shamir-Adleman) — криптосистема з відкритим ключем, яка широко використовується для безпечної передачі даних.

ECDSA (Elliptic Curve Digital Signature Algorithm) — один з найбільш комплексних алгоритмів шифрування з відкритим ключем. Ключі генеруються за

допомогою еліптичної кривої криптографії, які менше, ніж середні ключі, що генеруються алгоритмами цифрового підпису.

SSO (Single Sign-On) — метод аутентифікації, який дозволяє користувачам безпечно авторизуватись одразу в декількох додатках чи сайтах, використовуючи один набір даних для входу.

ID — ідентифікатор.

ВСТУП

Сьогодні велика кількість компаній, підприємств, організацій є цифровізованими. Вони використовують програмні рішення, які дозволяють контролювати та керувати масивами даних, тими чи іншими об'єктами системи та взаємодією між ними. Цифрові системи контролю, зокрема, теж є одними з таких рішень.

Системи контролю бувають різної складності та деталізації. Найскладнішими в створенні та реалізації є ERP-системи, які дають змогу приймати найбільш точні рішення про планування робочих ресурсів. Проте є цифрові рішення значно простіші, які дають менше інформації щодо роботи підприємства, але виконують поставлені перед ними задачі.

Незалежно від складності системи, вони є важливою частиною функціонування підприємства, компанії чи організації, бо підвищують продуктивність її роботи.

Система контролю навчального процесу — є одним із різновидів цифрових рішень, яке використовують у більшості вищих навчальних закладах України та закордоном. Ці системи мають у собі здебільшого схожі функції: бази даних студентів, викладачів, занять, завдань, розклад навчальних предметів та ведення обліку оцінок студентів за виконання ними робіт. Складніші системи можуть містити систему оцінювання ефективності викладачів.

Ідея розробити власне цифрове рішення виникла через особистий аналіз проходження навчального процесу у Національному технічному університеті України "Київському політехнічному інституті імені Ігоря Сікорського" на Теплоенергетичному факультеті, кафедрі АПЕПС. Система, яка використовується університетом, є достатньою для розв'язання задач студентів та викладачів, якщо процес навчання проходить безпосередньо у закладі. Проте, починаючи з 2020

року, не тільки в Україні, але й в усьому світі, відбулися зміни, що частково чи повністю унеможливили навчання за старими правилами.

У березні 20220 року почалося масове розповсюдження епідемії COVID-19, що змусило перевести всі навчальні заклади на дистанційну форму здобуття освіти. З цього часу спілкування з викладачем, проведення занять, постановка задач, перевірка їх виконання та багато інших процесів — усе це стало потребувати цифрових інструментів, зокрема хмарних рішень. Е-samrus КПІ став основою для взаємодії між студентами та викладачами, проте рішення не призначене для дистанційного навчання, а тому має недоліки з точки зору інтерфейсу, функціоналу та обробкою масивів даних.

Впродовж часу ситуація була з епідемією була непередбачуваною, але студенти та викладачі мали надію, що буде змога повернутись на попередню форму навчання. За декілька років учасники процесу поступово звикли до дистанційних умов.

24 лютого 2022 року сталось широкомасштабне вторгнення Російської Федерації на територію України. Це стало ключовим фактором продовження дистанційної форми навчання, адже знаходження великого скупчення людей в одному місці, що й передбачала минула форма навчання, є фактично небезпечним для життя усіх учасників процесу.

Відтак потреба у якісних функціональних цифрових рішеннях, особливо, у системах контролю навчального процесу, стала невід'ємною. Постійне покращення наявних систем є логічним та необхідним задля підвищення ефективності взаємодії студентів та викладачів та навчальної системи в цілому.

Програмний продукт, що був розроблений мною, має на меті покращити наступні процеси:

- Використання студентами системи контролю навчального процесу за допомогою за допомогою інтерфейсу, що є побудованим логічно, та зручно відображає наявні функції системи;

- Доступ студенту до його поточних завдань з дисциплін;

- Доступ студенту до розкладу завдань у поточний робочий тиждень;
- Відображення списку студентів групи з контактною інформацією кожного з них;
- Відображення списку дисциплін студента на поточний семестр з доступом до лекцій та завдань;
- Перегляд детальної інформації щодо лекцій з дисципліни з можливістю завантажити копію презентації чи текстовий конспект;
- Аналіз інформації завдань з навчального предмета з можливістю завантажити текстовий документ завдання на хмарне сховище та відправити звіт у з готовим вирішенням завдання у систему;
- Необхідність дізнатись дедлайн лабораторної чи іншого виду роботи;
- Оцінювання викладачем лабораторної чи іншої роботи студента з дисципліни.
- Відображення оцінок за самостійні роботи з дисципліни;
- Перегляд розкладу предметів студентом;
- Зміна особистої інформації студента чи даних для входу в систему;
- Додавання викладачем чи адміністратором інформації щодо розкладу предмету, лекцій чи завдань до нього.

У цій дипломній роботі буде детальніше описано процес розробки програмного продукту — вебдодатку контролю навчального процесу. Ми зрозуміємо, які є основні задачі системи, її цільова аудиторія та призначення; побачимо як функціонально користувачі продукту взаємодіють з ним. Також буде проведено порівняння з альтернативними системами, опитування цільової аудиторії. Ми зрозуміємо які інструменти та технології використовувались при розробці, якою є програмна реалізація системи, та як відбувається робота користувача зі створеним цифровим рішенням. Наприкінці буде підбито підсумки, які навички та знання були отримані під час цієї роботи та що було зроблено.

1 ПОСТАНОВКА ЗАДАЧІ РОЗРОБКИ ВЕБДОДАТКУ КОНТРОЛЮ НАВЧАЛЬНОГО ПРОЦЕСУ

Мета роботи полягає у створенні хмарного програмного забезпечення, що допоможе контролювати навчальним процес студенту та автоматизувати його взаємодію з викладачами.

1.1 Опис основних задач програмного продукту

Вебдодаток контролю навчального процесу є комплексним рішенням, яке містить в собі спектр функцій, необхідний, щоб задовольнити більшість запитів студентів чи викладачів, що знаходяться на дистанційній формі навчання.

Основними задачами програмного продукту є:

- Відображення детального розкладу навчання для студента по днях та тижнях;
- Доступ студента до матеріалів заняття впродовж навчального семестру;
- Відображення списку студентів групи з контактною інформацією до кожного з них;
- Можливість студенту отримати завдання лабораторних чи контрольних робіт та відправити готові звіти;
- Внесення викладачем інформації в систему щодо оцінки за роботу студента.

Виконання задач, які стоять перед програмним продуктом, задовольняють основні вимоги до системи, що дозволяє студенту та викладачеві контролювати навчальний процес, а також адмініструвати дані про час проведення дисциплінарних занять та зберігати на хмарному сховищі документи, що стосуються лабораторних та іншого виду робіт.

1.2 Призначення продукту. Цільова аудиторія

Програмний продукт першочергово призначений для студентів, викладачів та адміністрації вищих навчальних закладів. Студент використовує вебдодаток як продукт контролю власного навчального прогресу протягом семестру, а також інструмент доступу до матеріалів, що стосуються дисциплін, які проходить студент. Викладач чи адміністратор використовує панель керування для додавання інформації про предмет, розклад протягом навчального семестру, інформацію про заняття, текстові чи презентації матеріали до них, а також лабораторні, контрольні чи інші види завдання з інформацією по дедлайну роботи, документи, що містять завдання до роботи та оцінку, яку отримує студент при виконанні того чи іншого завдання.

Програмний продукт може бути використаний для закладів нетрадиційної освіти як додаткова платформа, що використовуються для онлайн-курсів з вивчення різних спеціальностей чи спеціалізацій. У такому разі система ролей не змінюється: є студенти, які проходять тимчасове навчання, викладачі та адміністратори, які керують даними та матеріалами щодо занять та завдань для самостійного опрацювання. Також система може містити завдання, що використовуються для роботи у групах, при цьому задача буде універсальною для всіх студентів.

Вебдодаток контролю навчального процесу також може бути використаний компаніями, що створюють власну освітню платформу для підвищення кваліфікації своїх співробітників. У такому разі співробітник займає роль студента у системі, а викладачем може виступати Senior співробітник того чи іншого відділу, чи Team Lead.

Цифрове рішення може використовуватися у закладах середньої освіти для контролю навчального процесу учнів. Роль студента у такому випадку замінюється

учнем, а викладач замінюється вчителем. Принцип роботи системи не потребує змін.

Усі наведені приклади використання програмного продукту доцільні при дистанційній, очній та гібридній формах навчання. Використання цифрової системи підвищить ефективність навчання та проведення занять внаслідок наявності єдиної системи контролю об'єктів навчального процесу та взаємодії між ними.

1.3 Висновки до розділу

У даному розділі була описана розробка вебдодатку контролю навчального процесу. Детальніше було описано які основні задачі виконує програмний продукт, якого результату дозволяє досягти набір функцій системи. Також визначено цільове призначення продукту, його варіанти використання. Розглянуто цільові аудиторії продукту та застосування вебдодатку в альтернативних сферах використання.

2 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

Предметною областю програмного продукту є системи контролю навчального процесу у вищих навчальних закладах. Найчастіше це хмарні рішення, які передбачають зберігання та обробку даних на сервері.

Для дослідження предметної області проведемо порівняння між основним продуктом та його аналоговими рішеннями, що уже використовуються студентами та викладачами. Було обрано одну з популярних систем — E-campus, що використовується у КПІ. Для більш детального порівняльного аналізу також обрано рішення Edu КНТЕУ, яке було створено на базі платформи Moodle.

2.1 Аналіз системи E-campus КПІ

Система електронного кампуса є основним інструментом обміну інформації між студентами, викладачами та адміністрацією університету чи окремих факультетів у КПІ продовж одинадцяти років. Аналіз саме цього продуктового рішення став основою для створення вебдодатку контролю навчального процесу.

E-campus КПІ наразі має дві версії системи, які працюють одночасно та є доповненням одне одного на рівні функціональних можливостей. Розглянемо обидві.

2.1.1 Оновлений E-campus КПІ

Електронний кампус університету з оновленим дизайном був створений як оновлення та покращення старої системи з точки зору користувацького досвіду, оновлення набору функцій, які, вочевидь, були актуалізовані з урахуванням нових

потреб студентів, викладачів та адміністрації, а також на основі аналізу поведінки та запитів користувачів за час використання старої системи E-campus.

Після входу в систему, ми бачимо головну сторінку електронного кабінету кампусу (рисунок 2.1). Вона розміщує у себе широкий спектр інформації, що стосується навчального процесу: це є розділ "Оголошення" у якому вказана універсальна інформація для всіх користувачів, а також гіперпосилання, що ведуть до різноманітної інформації стосовно навчального процесу.

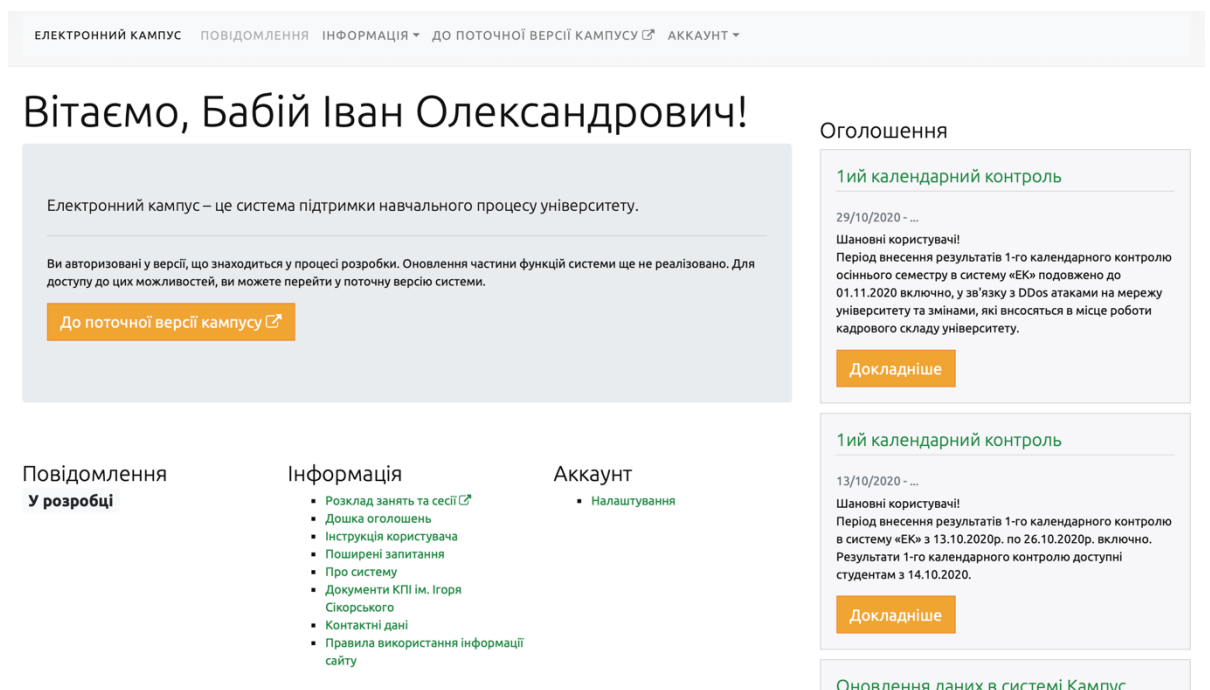


Рисунок 2.1 — Головна сторінка оновленого E-campus КПІ

Перевагою даного рішення є інформативний інтерфейс з мінімальною кількістю рівнів для отримання тієї чи іншої інформації, та інтегрованість системи з іншими продуктами університету, такими як Розклад КПІ чи старою версією кампусу.

Недоліками оновленого рішення можна назвати першочергово незавершений випуск продукту на загальну аудиторію. Електронний кампус мав стати заміною старої версії системи, проте більше ніж за чотири роки не реалізував основні функції системи, пов'язані з відображенням оцінок по тим чи іншим роботам, а

також доступ до матеріалів занять чи завдань. Іншим недоліком системи є перенавантажений інформацією графічний інтерфейс, який є неправильно побудований з точки зору UX. Наслідком останнього недоліку є збільшений час на пошук інформації, яку хоче знайти користувач, а також менша ймовірність, що очікуваний результат від системи буде отримано до виходу з неї.

Отже, можна констатувати, що оновлена версія електронного кампуса є допоміжним інструментом до старої системи кампуса, але досі не є повноцінним рішенням, у якому зосереджено всю необхідну інформацію, що стосується навчального процесу.

2.1.2 Основний електронний кампус КПІ

Електронний кампус, який є основним інструментом для отримання інформації з дисциплін, оцінок за ті чи інші роботи протягом семестру, відображені у основній версії кампуса, яка розроблялась з 2011 року.

Доступ до системи здійснюється через посилання на сторінці оновленого електронного кампуса. Авторизація користувача проходить також через нову версію. Перехід у систему відбувається на головну сторінку. (Рисунок 2.2)

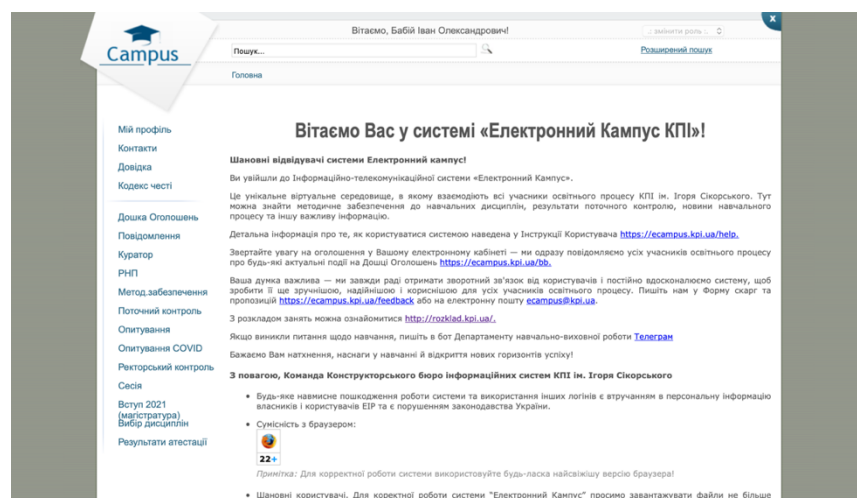


Рисунок 2.2 — Головна сторінка електронного кампуса КПІ

Перевагами даної системи є доступ до комплексної інформації, що стосується дисциплін: детальний опис навчального плану з предметів, результати атестацій, сесій. Також кампус містить детальний опис оцінювання впродовж семестру, план заходів, які передбачені, та доступ до внутрішніх та зовнішніх матеріалів, що стосуються предмету. Система містить інструмент прямої комунікації між викладачами та студентами у вигляді чату та опитування, призначені для студентів.

Недоліками системи є незручний інтерфейс користувача, який проявляється у складності знайти чи додати ту чи іншу інформацію у призначені для цього розділи. Також важливим недоліком системи є перенасичення функціоналом, що робить більш ніж половину функцій повністю або майже не використовуваними. Останньою проблемою, яка погіршує якість користування кампусом, є не актуалізована інформація щодо контактів викладачів та деталізованих даних і документів, що стосуються безпосередньо проходження навчального процесу протягом семестру, та допомагають студентами краще вивчати ту чи іншу дисципліну.

Отже, електронний кампус КПІ, який містить у собі основні функції для контролю та поліпшення якості навчального процесу, не задовольняє потреби, які стоять перед ним в умовах дистанційного навчання.

2.2 Аналіз системи Edu КНТЕУ на базі Moodle

Система Edu, розроблена КНТЕУ для внутрішнього користування студентами та викладачами, була створена у 2018 році, але найбільшого поширення отримала з початку пандемії COVID-19 для виконання задач комунікаційного типу між різними типами користувачів, а також обміну матеріалами з дисциплін та іншої інформації, що стосується навчального процесу.

Дана система створена на основі платформи Moodle і тому повторює його дизайн інтерфейсу та основні набори функцій.

Головна сторінка містить широкий спектр доступів, зокрема: доступ до курсів (дисциплін) студента, заняття, які будуть відбуватись наступний тиждень, та гіперпосилання на окремі кабінети користувача та допоміжні матеріали, які необхідні студенту (Рисунок 2.3).

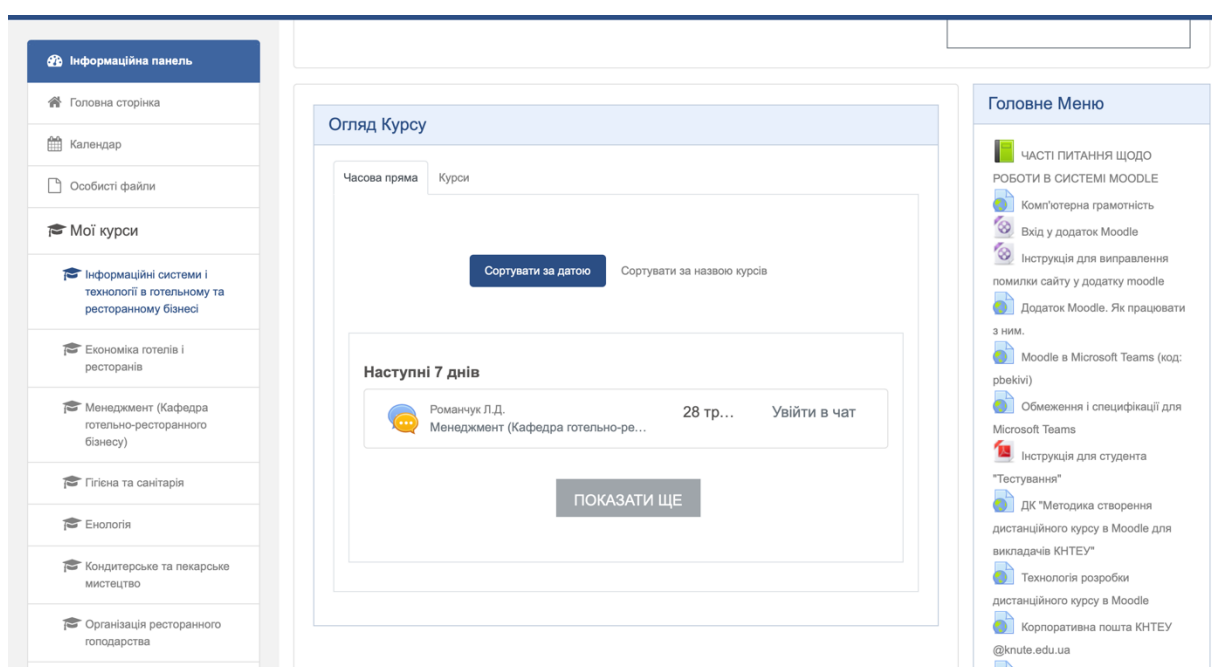


Рисунок 2.3 — Головна сторінка інформаційної панелі студента КНТЕУ

Основна перевага системи закладена в особливості платформи Moodle: вона створена для персоналізованого навчального середовища. Це допомагає користувачу працювати з інтерфейсом, адже відображає релевантну інформацію для студента чи викладача. Крім того, Edu має детальну глибинну систему роботи з кожною дисципліною окремо. Студент може завантажувати виконані роботи, переглядати завдання, заняття, власні оцінки за кожну роботу окремо чи загальну за семестр. Це дає змогу користувачу мати повну інформацію щодо свого прогресу у навчальному процесі та планувати підготовку власних задач.

Окремою перевагою можна відзначити можливість у системі створювати тести для оцінювання знань студентів. Ця функція від Moodle користується окремою популярністю серед навчальних закладів України.

Попри привілеї, система має свої недоліки. Насамперед система Moodle є універсальною платформою, а отже потребує великої кількості налаштувань, зокрема логіки відображення занять та дисциплін, завдань та тестів, що може, у певних випадках, ускладнити інтерфейс та взаємодію з ним.

Загалом система виглядає схоже до вебдодатку контролю навчального процесу, проте має свої відмінності: Edu має ширший спектр функцій, але і складнішу логіку користування.

2.3 Висновки до розділу

У цьому розділі було описано предметну область цифрового продукту, аналоги, які брались до уваги при розробці продукту. Також детальніше було проаналізовано системи, які використовують одні з найпопулярніших вищих навчальних закладів України.

Системи мають свої переваги і недоліки, які суттєво впливають на користувацький досвід та ефективність роботи з системою. Вебдодаток, на противагу обом аналогам, має краще побудований інтерфейс з точки зору UX та має перелік функцій, що є необхідними для студента для контролю, планування та роботи над завданнями з предметів протягом навчального процесу.

3 ВИБІР ЗАСОБІВ РОЗРОБКИ

Розробка продукту розподіляється на декілька етапів, відповідно до типу системи, який було обрано. Оскільки система є вебдодатком, то вона містить:

- розробку клієнтської частини продукту, саме інтерфейсу користувача, інакше іменованої як Front-end частини;
- розробку програмно-апаратної частини, яка відповідає за обробку даних та функцій; інакше називається Back-End.

3.1 Опис засобів Front-end розробки

За основу розробки Front-end частини вебдодатку було обрано NextJS, що є фреймворком мови React. Також було використано TypeScript у складі NextJS. Для обробки даних застосовувався SWR. Для управління пакетами був обраний Yarn. При Front-end розробці не використовувалася мова розмітки HTML, оскільки вона є менш доцільною при розробці вебдодатків схожого типу. Розробка відбувалась у середовищі Atom.

NextJS — гнучкий фреймворк React, який надає блоки для побудови вебдодатків. Використовується для вирішення загальних завдань додатка, таких як маршрутизація, отримання даних, інтеграція — все що покращує роботу розробників та кінцевих користувачів [1].

React — це бібліотека Javascript для побудови користувацьких інтерфейсів. Одна з найбільших переваг React є можливість створення компонентів, які схожі на елементи HTML, що багаторазово використовуються, для швидкого й ефективного створення користувацьких інтерфейсів [2].

Typescript — суворо типізована мова програмування, створена на основі JavaScript, та яка надає найкращі інструменти для будь-якого масштабування. За

рахунок суворої типізації Typescript допомагає виявити помилки у коді шляхом статичної перевірки типів [3].

CSS — мова, що створена для опису оформлення вебсторінок, включаючи колір, макет та шрифти [4].

SWR — це бібліотека React Hooks для отримання та обробки даних. Перевагами SWR є легка вага програмного коду, оновлення даних у реальному часі, пагінація, незалежність від Back-end. Завдяки SWR компоненти будуть постійно та автоматично отримувати потік оновлених даних [5].

Yarn — менеджер пакетів для коду. Пакети, що передаються, містять в собі весь код, а також файл package.json, що називається маніфестом, який описує цей пакет [6].

JetBrains WebStorm — середовище для Front-end розробки від компанії JetBrains, створена на основі платформи IntelliJ IDEA. [7].

3.2 Опис засобів Back-end розробки

Для розробки Back-end частини додатка було обрано PostgreSQL як основну мову бази даних, сервер Heroku на якому відбувається хостинг, а також Hasura, яка дозволяє використовувати API GraphQL як зручний інтерфейс для роботи з базою даних. Для автентифікації користувача використано JWT.

PostgreSQL — система управління реляційними базами даних з відкритим вихідним кодом [8].

Heroku — PaaS платформа, яка дозволяє розробникам створювати, запускати чи керувати додатком повністю у хмарному середовищі [9].

Hasura — швидкий GraphQL-сервер, який надає GraphQL API у реальному часі через PostgreSQL, з webhook тригерами для подій баз даних і віддаленими схемами для бізнес-логіки [10].

GraphQL — мова запитів для API й середовище виконання для реалізації цих запитів з використанням наявних даних. GraphQL забезпечує повний та зрозумілий опис даних в API, дає клієнтам можливість запитувати саме те, що їм потрібно, полегшує розвиток API з часом та дозволяє використовувати потужні інструменти розробника [11].

JWT — це відкритий стандарт (RFC 7519), який визначає компактний та самодостатній спосіб безпечної передачі інформації між сторонами у вигляді об'єкта JSON. Ця інформація може бути перевірена і можна довіряти, оскільки вона підписана цифровим підписом. JWT можуть бути підписані секретно (за допомогою алгоритму HMAC) або парою відкритий/закритий ключ із використанням RSA або ECDSA [12].

3.3 Вимоги до програмно-апаратної частини

Вебдодаток контролю навчального процесу працює на усіх сучасних браузерах, зокрема в Internet Explorer версії 11 і вище.

Мінімальними системними вимогами для використання програмного забезпечення є:

- Операційна система: Windows 7, Mac OS 10.5, Linux 5;
- 2GB RAM;
- Процесор Intel Core 2 Duo;
- 10GB HDD OR SSD;
- Відеокарта Radeon HD 8450;
- доступ до мережі Інтернет.

3.4 Висновки до розділу

У даному розділі пояснено, які типи розробки використовуються при створенні вебдодатку контролю навчального процесу. Серед засобів для створення Front-end частини було описано мови програмування та технології, які використовуються. Також було показано ключові переваги засобів, які стали причиною вибору тієї чи іншої технології. Back-end була поверхнево описана за допомогою яких систем, платформ, серверів чи мов програмування відбувається, і детальніше була описана кожна з них з урахуванням власних переваг.

4 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

Програмна реалізація продукту складається з формування бази даних, що відповідає за комплексну інформацію, яка охоплює усі напрямки навчального процесу з детальним описом кожного з них.

4.1 Формування системи контролю навчального процесу

Для формування системи контролю навчального процесу було розроблено базу даних, яка містить набір даних, необхідний для повного функціонування вебдодатку. Було створено наступні таблиці:

- Відвідування (Оригінальна назва Attendance);
- Заняття (Classes);
- Дисципліни (Disciplines);
- Групи (Groups);
- Студенти (Students);
- Завдання (Tasks);
- Викладачі (Teachers);
- Завантаження (Uploads).

Детальніше проаналізуємо роботу кожної з них.

Таблиця Attendance (рисунок 4.1) відповідає за взаємозв'язок між групою та заняттям. Для цього побудовано взаємозв'язок між об'єктами за принципом "один-до-одного" з базою занять за допомогою ідентифікатора, а з групою — за допомогою коду групи. Attendance існує для того, щоб відображати в інтерфейсі студента лише ті заняття, що належать до його групи, та не показувати ті, що не є релевантними.

attendance

Browse Rows Insert Row Modify Relationships Permissions

Filter **Sort**

-- column -- [eq] equals -- value -- -- column -- --

Run query Export data

		group_code	class_id	created_at	updated
		TM-81	acfd7cc-63a5-4083-b687-0635291efe6e	2022-05-27T17:07:50.369603+00:00	2022-05-27
		TM-81	71a57bc3-9519-4c1e-b3e4-b2122976ef5a	2022-05-27T17:07:53.171689+00:00	2022-05-27
		TM-81	c78c6caf-cd5e-430e-ba78-c660d27f82f2	2022-05-27T17:08:03.907426+00:00	2022-05-27
		TM-81	d6a8ae58-7605-40cb-b6ad-4c1f22114e25	2022-05-27T17:08:09.117095+00:00	2022-05-27
		TM-81	ba4ba8c4-55d6-4b3c-a1cb-64ead94c9c4d	2022-05-27T17:08:17.659074+00:00	2022-05-27
		TM-81	7318983c-f4cc-4e51-8b75-426ec7c6bac4	2022-05-27T17:08:22.007619+00:00	2022-05-27
		TM-81	c8ce271e-68d8-40e0-8b66-bf2a96ff1626	2022-05-27T17:08:26.682396+00:00	2022-05-27
		TM-81	85763187-ad42-4f75-85f7-28a7b9a6accc	2022-05-28T08:37:16.153268+00:00	2022-05-28

Рисунок 4.1 — Таблица Attendance

Наступною є таблиця Classes (рисунок 4.2).

classes

7318983c-f4cc-4e51-8b75-426ec7c6bac4 **Browse Rows** Insert Row Modify Relationships Permissions

Filter **Sort**

-- column -- [eq] equals -- value -- -- column -- --

Run query Export data

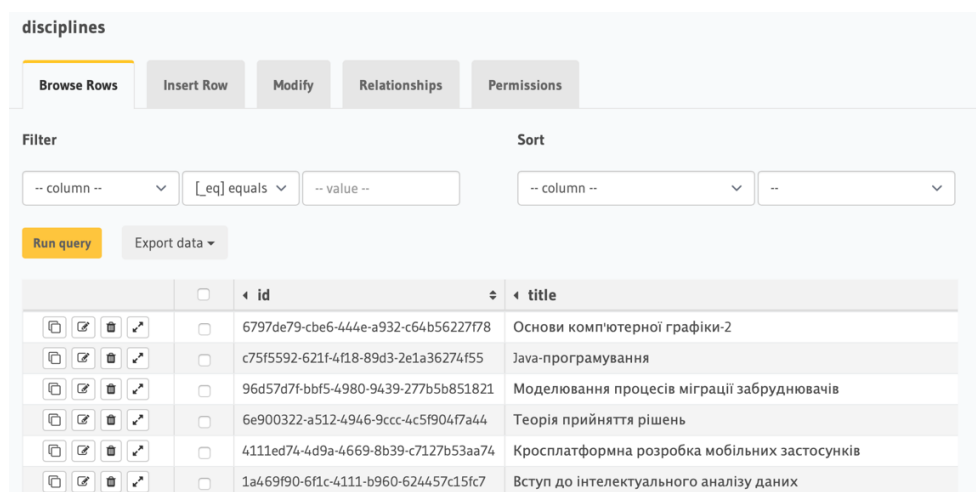
		id	title	type	start_time
		acfd7cc-63a5-4083-b687-0635291efe6e	Основи комп'ютерної графіки-2	Лекція	2022-05-29T10:00:00
		71a57bc3-9519-4c1e-b3e4-b2122976ef5a	Основи комп'ютерної графіки-2	Лекція	2022-05-29T12:00:00
		e7fd4834-5b06-4dd1-964c-f17c04e61118	Основи комп'ютерної графіки-2	Лекція	2022-05-29T14:00:00
		c78c6caf-cd5e-430e-ba78-c660d27f82f2	Основи комп'ютерної графіки-2	Лекція	2022-05-29T16:00:00
		d6a8ae58-7605-40cb-b6ad-4c1f22114e25	Java-програмування	Лекція	2022-05-28T10:00:00
		ba4ba8c4-55d6-4b3c-a1cb-64ead94c9c4d	Java-програмування	Лекція	2022-05-28T12:00:00
		7318983c-f4cc-4e51-8b75-426ec7c6bac4	Java-програмування	Лекція	2022-05-28T14:00:00
		c8ce271e-68d8-40e0-8b66-bf2a96ff1626	Java-програмування	Лекція	2022-05-28T16:00:00
		85763187-ad42-4f75-85f7-28a7b9a6accc	Основи комп'ютерної графіки-2	Лекція	2022-05-28T14:00:00

Рисунок 4.2 — Таблица Classes

Вона зберігає інформацію про заняття усіх груп, які проходять протягом усього проміжку часу. Наразі система працює так, що кожне заняття у кожній групі є унікальним, а тому потребує окремого запису в таблиці. Відповідні записи мають у собі наступні параметри:

- ID, який використовується для зв'язку з іншими таблицями за принципом "багато-до-багатьох";
- Назва предмету;
- Тип: лекція, лабораторна, практика чи інше;
- Дата та час початку заняття;
- Дата та час закінчення;
- ID дисципліни;
- Посилання на матеріали заняття (зовнішні посилання на хмарне сховище);
- Час створення та поновлення запису.

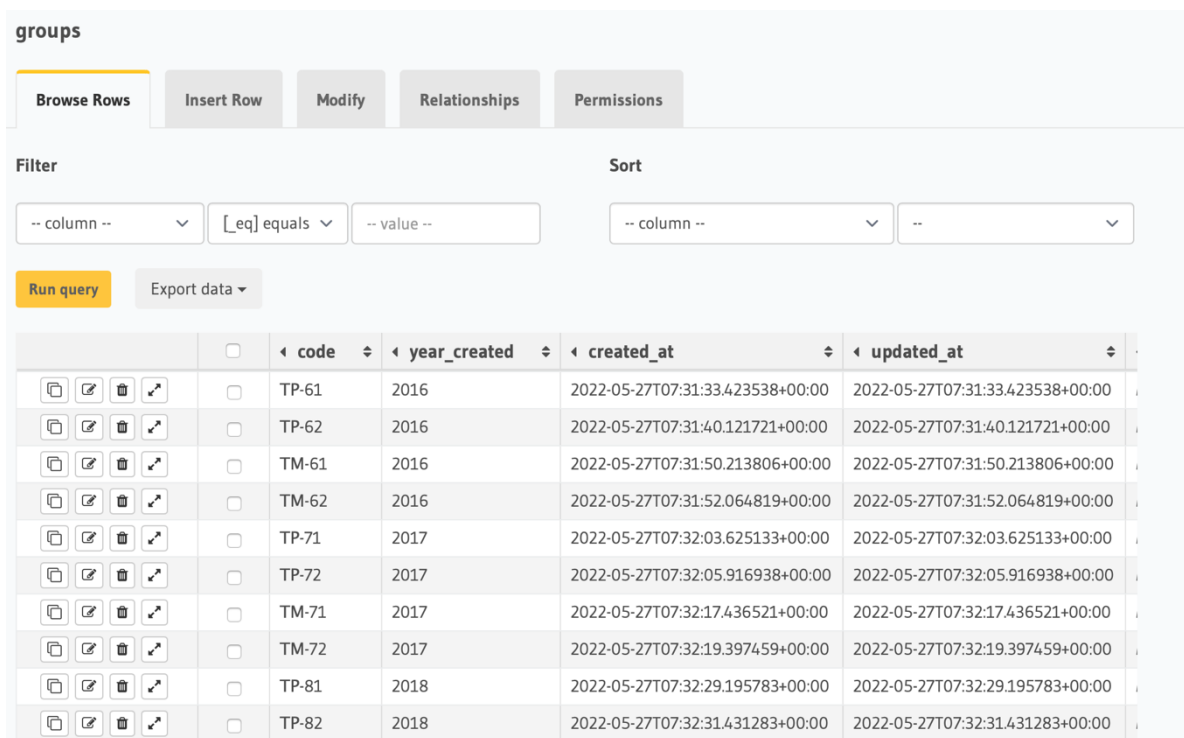
Наступною таблицею є Disciplines (рисунок 4.3). У ній зберігається інформації про увесь список навчальних дисциплін, які існують у вищому навчальному закладі. Записи містять у собі інформацію ідентифікатор дисципліни, назву та час створення і поновлення запису. Дані з таблиці передаються до Classes за принципом "багато-до-багатьох".



	id	title
	6797de79-cbe6-444e-a932-c64b56227f78	Основи комп'ютерної графіки-2
	c75f5592-621f-4f18-89d3-2e1a36274f55	Java-програмування
	96d57d7f-bbf5-4980-9439-277b5b851821	Моделювання процесів міграції забруднювачів
	6e900322-a512-4946-9ccc-4c5f904f7a44	Теорія прийняття рішень
	4111ed74-4d9a-4669-8b39-c7127b53aa74	Кросплатформна розробка мобільних застосунків
	1a469f90-6f1c-4111-b960-624457c15fc7	Вступ до інтелектуального аналізу даних

Рисунок 4.3 — Таблиця Disciplines

Таблиця Groups (рисунок 4.4) зберігає список груп, що є в навчальному закладі. Полями запису є код групи, дата створення, час створення та поновлення запису, а також ID старости та куратора для зв'язку з таблицями студентів та викладачів за принципом "один-до-одного". За принципом "багато-до-багатьох" таблиця зв'язана з таблицями відвідування та студентів.



	☐	code	year_created	created_at	updated_at
   	☐	TP-61	2016	2022-05-27T07:31:33.423538+00:00	2022-05-27T07:31:33.423538+00:00
   	☐	TP-62	2016	2022-05-27T07:31:40.121721+00:00	2022-05-27T07:31:40.121721+00:00
   	☐	TM-61	2016	2022-05-27T07:31:50.213806+00:00	2022-05-27T07:31:50.213806+00:00
   	☐	TM-62	2016	2022-05-27T07:31:52.064819+00:00	2022-05-27T07:31:52.064819+00:00
   	☐	TP-71	2017	2022-05-27T07:32:03.625133+00:00	2022-05-27T07:32:03.625133+00:00
   	☐	TP-72	2017	2022-05-27T07:32:05.916938+00:00	2022-05-27T07:32:05.916938+00:00
   	☐	TM-71	2017	2022-05-27T07:32:17.436521+00:00	2022-05-27T07:32:17.436521+00:00
   	☐	TM-72	2017	2022-05-27T07:32:19.397459+00:00	2022-05-27T07:32:19.397459+00:00
   	☐	TP-81	2018	2022-05-27T07:32:29.195783+00:00	2022-05-27T07:32:29.195783+00:00
   	☐	TP-82	2018	2022-05-27T07:32:31.431283+00:00	2022-05-27T07:32:31.431283+00:00

Рисунок 4.4 — Таблиця Groups

Students (рисунок 4.5) зберігає в собі список всіх студентів з інформацією про їхні імена, номери телефону, пошту, код групи у якій вони навчаються, а також статус студента, і пароль для входу у систему. У цьому випадку номер телефону може бути нульовим значенням, а отже відсутнім. Взаємозв'язок за об'єктом відбувається з Groups використовуючи код, а за масивом відбувається з Groups з'єднуючи ID старости, який є студентом, та з Uploads — з ID студентом.

students					
Browse Rows Insert Row Modify Relationships Permissions					
Filter			Sort		
-- column -- [_eq] equals -- value --			-- column -- --		
Run query Export data					
		id	name	phone	email
☐	☐	9530b185-13b2-47be-b3c0-021ee8167223	Йосип Андрійович Кравченко	+3805153962500	saxeizaumi
☐	☐	1dfd47e-3c74-4082-a99d-a274413c3f7b	Надія Янівна Середа	+38083314508983	wazaddauc
☐	☐	91cf125c-173e-45cb-a923-2dfceb6a8fe3	Антоненко Артем Васильович	+380711976127	xawabraya
☐	☐	22660a68-aae8-47c2-8f31-9fbc5b92505b	Ірина Йосипівна Броваренко	+3802484852284	lefujellobe
☐	☐	322308bb-7360-414f-84ec-16428d37e5b6	Валентин Борисович Василенко	+380824090124	yaproulapr
☐	☐	370bda5d-e436-4f44-a1f6-b3fe44ba9e01	Вікторія Романівна Мельниченко	+3803947971939	сахопewati
☐	☐	d58fed5b-1010-4eb4-87dc-3c275b5f71d2	Єлизавета Євгеніївна Середа	+3805810411335	ganedoido:
☐	☐	06380f9d-6725-4c99-90d6-01ed573048b5	Васильєв Марія Валентинівна	+38035853012429	ribraleuquc
☐	☐	92431db4-a907-4daf-80ce-1d1eec07ed6a	Бабій Іван Олександрович	+380507771755	i.babiy@ar

Рисунок 4.5 — Таблиця Students

Таблиця Tasks (рисунок 4.6) містить інформацію про лабораторні чи інші типи завдань, які призначені для виконання студентом. Кожен запис містить власний ID, назву роботи, ID заняття до якого воно відноситься, опис роботи, дедлайн кінцевої здачі, а також час створення та оновлення запису.

tasks					
Browse Rows Insert Row Modify Relationships Permissions					
Filter			Sort		
-- column -- [_eq] equals -- value --			-- column -- --		
Run query Export data					
		id	title	class_id	
☐	☐	10ea1c38-5434-4b89-b3df-199709589e30	Лабораторна №1	acfd7cc-63a5-4083-b687-0635291efe6e	
☐	☐	98979d2d-05ea-44a6-bfac-a7e993688238	Лабораторна №2	71a57bc3-9519-4c1e-b3e4-b2122976ef5a	
☐	☐	d24ef165-177c-48c6-b3dc-37e6dd096249	Лабораторна №3	e7fd4834-5b06-4dd1-964c-f17c04e61118	

Рисунок 4.6 — Таблиця Tasks

Tasks з'єднується з таблицею Заняття за принципом "один-до-одного", використовуючи ідентифікатор заняття. За принципом "багато-до-багатьох" таблиця пов'язується з Uploads через ідентифікатор завдання.

Таблиця Teachers містить інформацію про викладачів (рисунок 4.7). Щодо деталей інформації — у записах використовується ID, повне ім'я викладача, номер телефону та електронна пошта. Усі дані є обов'язковими для заповнення. Таблиця пов'язана за допомогою масивів з Groups по ID викладача, який є куратором тієї чи іншої групи.

	☐	id	name	phone	email
☐	☐	fd6b7126-7869-4d33-b6a6-c4c5550e549e	Шинкаренко Павло Іванович	380392895867	naseummauyinna-3
☐	☐	c90a4ba5-3bf6-46da-aa5f-c842c6dc34df	Богдан Євгенійович Лисенко	380875812446	rovaunavoxei-92346

Рисунок 4.7 — Таблиця Teachers

Останньою в системі є таблиця Uploads (рисунок 4.8). Вона містить інформацію про відправлені звіти студентів з робіт, що вони виконали. Кожен запис має дані про ID студента, ID задачі, посилання на звіт, а також оцінка за виконане завдання. Адміністратор використовує цю таблицю для швидкого доступу до посилань та виставлення оцінок за роботу. Бали відображаються у списку завдань студента з того чи іншого предмету.

uploads

[Browse Rows](#)
[Insert Row](#)
[Modify](#)
[Relationships](#)
[Permissions](#)

Filter
-- column --
[_eq] equals
-- value --

Sort
-- column --
--

[Run query](#)
[Export data](#)

	☐	◀ student_id	◀ task_id	◀ link
☐ ☐ ☐ ☐	☐	92431db4-a907-4daf-80ce-1d1eec07ed6a	98979d2d-05ea-44a6-bfac-a7e993688238	https://cloud.hasura.io/projec

Рисунок 4.8 — Таблиця Uploads.

Таблиця має відношення за принципом "один-до-одного" з таблицею Students та Tasks, використовуючи відповідні ідентифікатори.

4.2 Опис діаграми прецедентів

Для докладнішого опису взаємодії користувачів з системою у програмному продукті було створено діаграму прецедентів (рисунок 4.8).

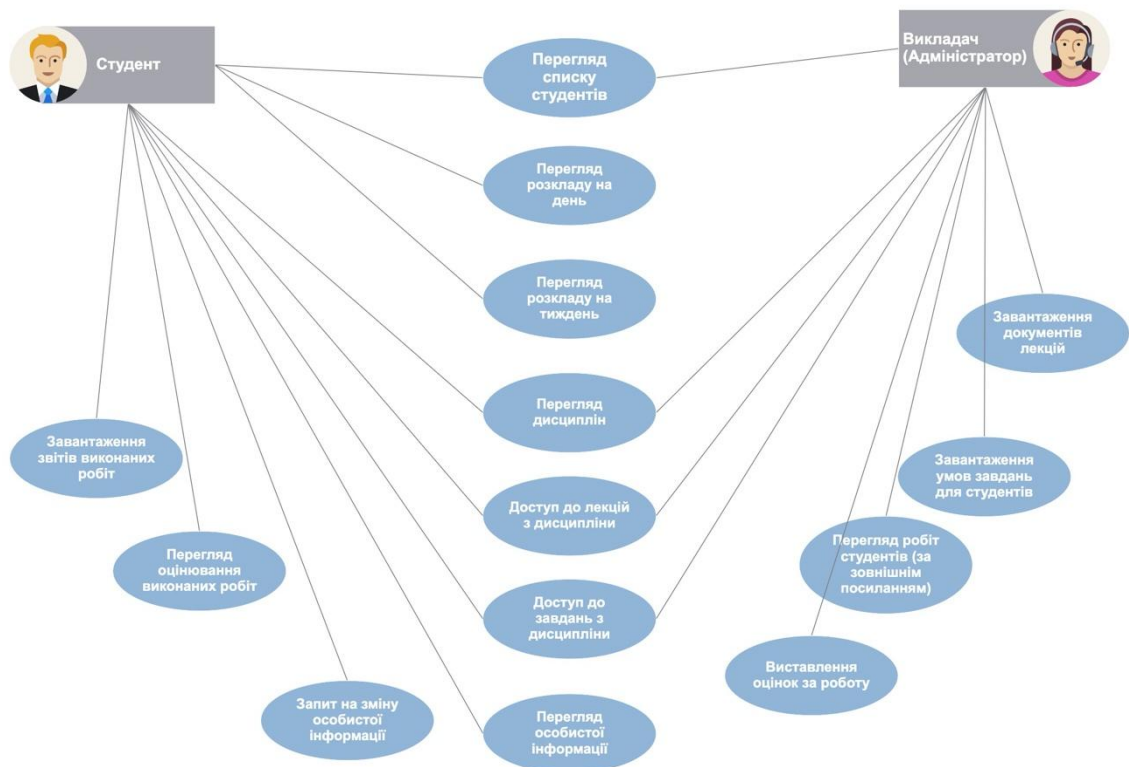


Рисунок 4.8 — Діаграма прецедентів

На схемі показано учасників системи — студента та викладача, який виступає у ролі адміністратора. Можливості продукту мають як спільні, так і унікальні сценарії користування.

Основними задачами продукту для студента є перегляд розкладу занять, взаємодія з документами, що стосуються лекцій та завдань на опрацювання, та перевірка оцінки за роботу. Викладач, у свою чергу, має загальний доступ до лекцій, матеріалів до них, умов завдань лабораторних чи інших робіт. Також викладач має можливість додавати нові заняття чи завдання, і отримувати звіти готових робіт студентів та оцінювати їх.

Перевага діаграми прецедентів у тому, що наочно можна побачити наявний набір функцій, учасників та їх взаємодію з функціями системи та спільні чи унікальні можливості використання.

Діаграма була розроблена за допомогою середовища Miro, яке є професійним інструментом для побудови схем.

4.3 Висновок до розділу

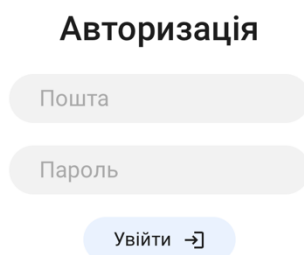
У цьому розділі було описано формування системи контролю навчального процесу та безпосередньо базу даних, яка є її основою. Було представлено кожну таблицю, яка є складової системи; пояснено які поля кожна з таблиць містить, а також описано усі відношення між таблицями, що демонструє залежності між ними. До того ж було описано діаграму прецедентів, що відображає ключових користувачів та взаємодію між ними та варіантами використання системи.

5 РОБОТА КОРИСТУВАЧА З ПРОГРАМНОЮ СИСТЕМОЮ

Вебдодаток містить декілька графічних систем, які мають різні точки входу. Вони розподіляються залежно від того, хто користується системою. Студент використовує основну графічну систему, яка була створена індивідуально для проєкту. Викладач, який виступає у ролі адміністратора, використовує систему Hasura для керування даними.

5.1 Вхід студента в систему

Точкою входу студента для роботи з системою є вікно авторизації (рисунок 5.1).



The image shows a login form with the title "Авторизація" (Authorization) in bold black text. Below the title are two input fields: "Пошта" (Email) and "Пароль" (Password), both with light gray borders and rounded corners. Below these fields is a blue button with the text "Увійти →" (Login →) in white.

Рисунок 5.1 — Авторизація у систему (для студента)

Авторизація відбувається шляхом вводу особистих даних студента, а саме пошти та паролю, які були попередньо відправлені адміністратором системи. Перевірка даних відбувається на коректність введення пошти за формою, а також повним збігом даних з тими, що наявні безпосередньо в базі.

Авторизація у систему є обов'язковою до виконання. Будь-який незалогінований користувач не має доступу до інших сторінок вебдодатку. При спробі перейти на ту чи іншу сторінку, система повертає користувача на вікно авторизації.

Повторне введення пошти та паролю для входу не є обов'язковим. Система зберігає дані про вхід користувача з того чи іншого пристрою та дозволяє авторизованому користувачу не проходити шлях логіну щоразу.

Якщо користувач бажає перезайти в систему, для цього він має натиснути на шапці сайту (рисунок 5.2) на кнопку виходу, розташовану у правій частині.

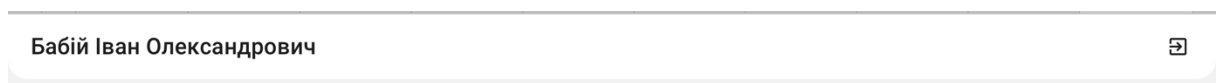


Рисунок 5.2 — Шапка сайту

Після виходу, користувач має авторизуватись повторно, використовуючи свої дані для входу.

5.2 Вхід викладача (адміністратора) в систему

Викладач та адміністратор у цьому програмному продукті займають одну роль, а тому вхід у систему буде однаковий для обох.

Авторизація у систему відбувається через сторінку серверу Nasura, що має зручний інтерфейс для користування (рисунок 5.3).

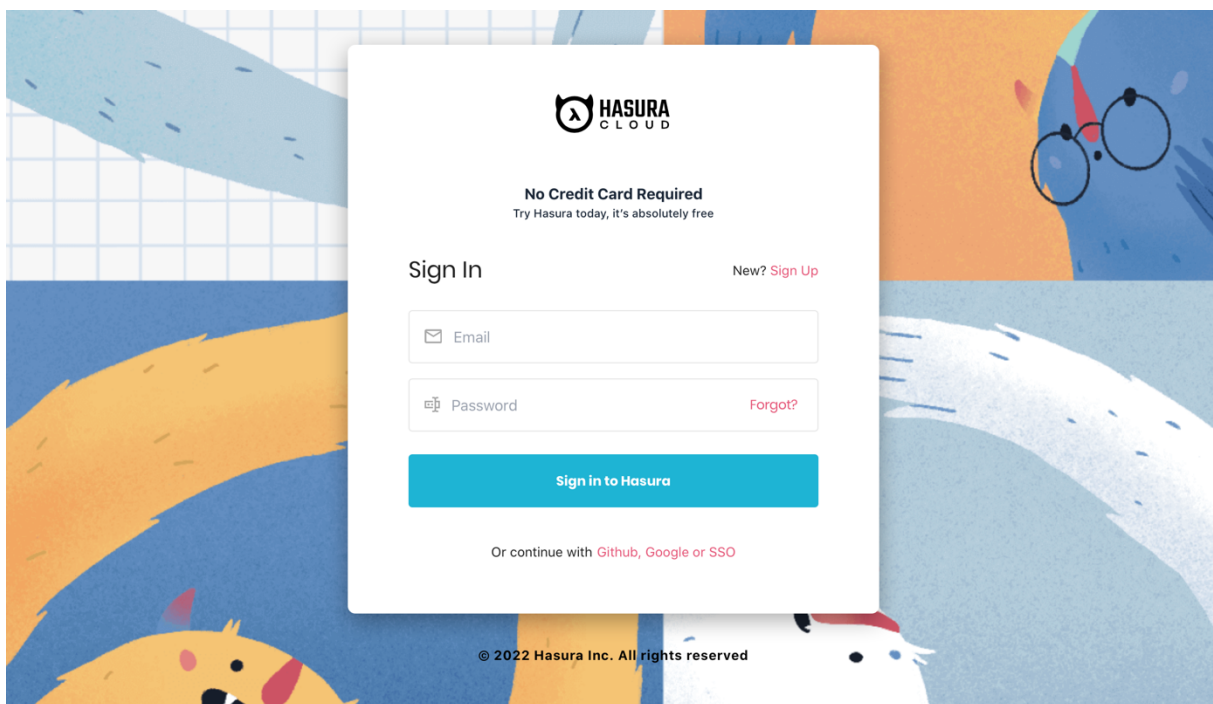


Рисунок 5.3 — Авторизація у систему (для викладача чи адміністратора)

Система передбачає вхід за допомогою облікового запису Github, Google чи за допомогою SSO. Також можна використовувати такі облікові дані як електронна пошта та пароль.

Якщо користувач втратив власний пароль, його можна відновити, використовуючи пошту. На електронну адресу прийде унікальне посилання, перейшовши на яке можна вказати новий пароль для входу.

5.3 Інтерфейс програми для студента

Після входу в систему студент потрапляє на головну сторінку (рисунок 5.4). Вона виконує роль дашборду, у якій зосереджена основна інформація, яку найчастіше потребує користувач: розклад занять на сьогодні та список завдань. Розклад містить у собі назву предмета, його тип та час проведення. Завдання, що

знаходяться у правій частині екрану, складаються з дедлайну здачі, назви роботи та дисципліни, для якої робота виконується.

У лівій частині екрану міститься меню вебдодатку, у якому розміщені наступні розділи: Дашбоард, Студенти, Дисципліни, Розклад та Налаштування. Активна сторінка підсвічена синім кольором.

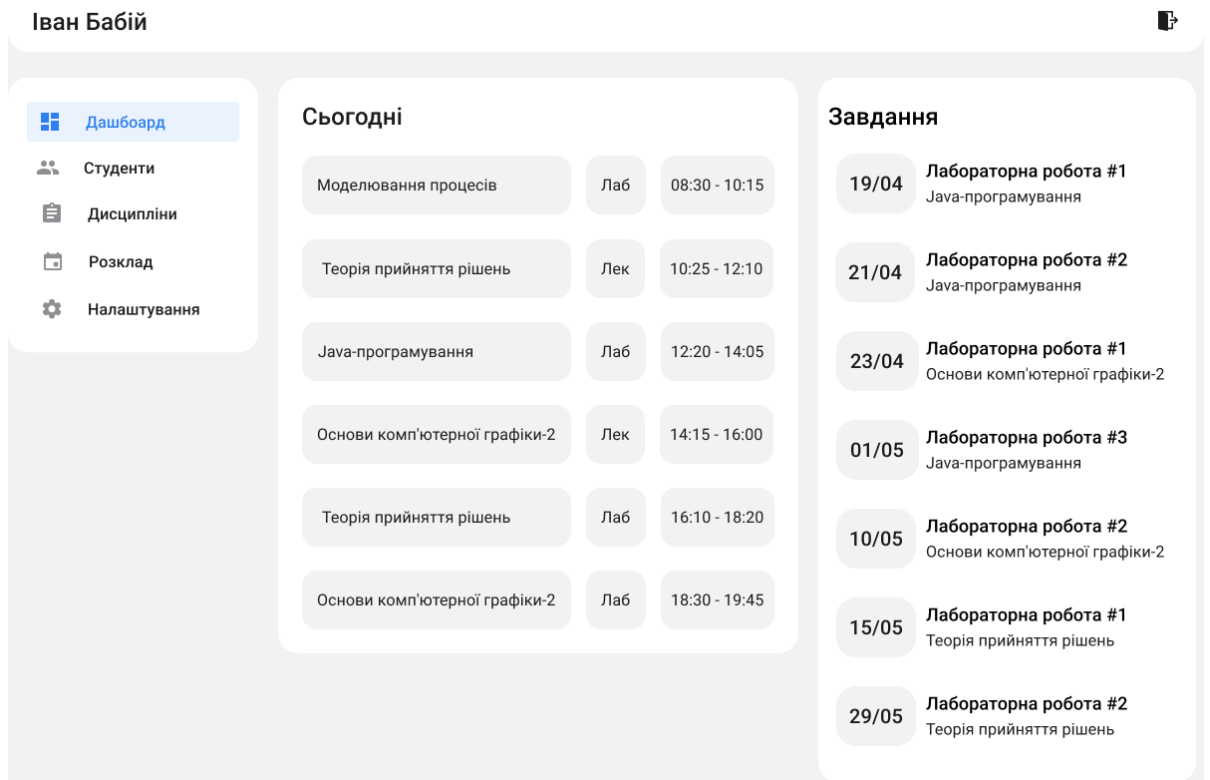


Рисунок 5.4 — Головна сторінка вебдодатку для студента

При переході у розділ Студенти (рисунок 5.5), ліва та верхня частина екрану не змінюються. У заголовку основного блоку зображається назва групи, у якій знаходиться студент. Нижче показана таблиця зі списком всіх студентів групи з їхнім статусом, номером телефону та електронною поштою.

Список студентів ТМ-81

№	ПІБ	Статус	Телефон	E-mail
1	Йосип Андрійович Кравченко	Студент	+3805153962500	saxeizaummauxa-3134@yopmail.com
2	Надія Янівна Середа	Студент	+38083314508983	wazaddauquexa-4202@yopmail.com
3	Антоненко Артем Васильович	Студент	+380711976127	xawabrayaco-7276@yopmail.com
4	Ірина Йосипівна Броваренко	Студент	+3802484852284	lefujellobe-5913@yopmail.com
5	Валентин Борисович Василенко	Студент	+380824090124	yaproulapruyu-4367@yopmail.com
6	Вікторія Романівна Мельниченко	Студент	+3803947971939	saxopewatra-6398@yopmail.com
7	Єлизавета Євгеніївна Середа	Студент	+3805810411335	ganedoidoso-4984@yopmail.com
8	Васильєв Марія Валентинівна	Студент	+38035853012429	ribraleuquoufre-8023@yopmail.com
9	Бабій Іван Олександрович	Студент	+380507771755	i.babiy@artk.ai

Наступним розділом є Дисципліни (рисунок 5.6)

Іван Бабій

■ ■ Дашборд

 Студенти

Дисципліни

Розклад

 Налаштування

Основи комп'ютерної графіки-2

Викладач: Ходаковський Олексій Володимирович

Лекції

Завдання

Теорія прийняття рішень

Викладач: Шинкаренко Павло
Іванович

Лекції

Завдання

Аналіз інтернету речей

Викладач: Богдан Євгенійович
Лисенко

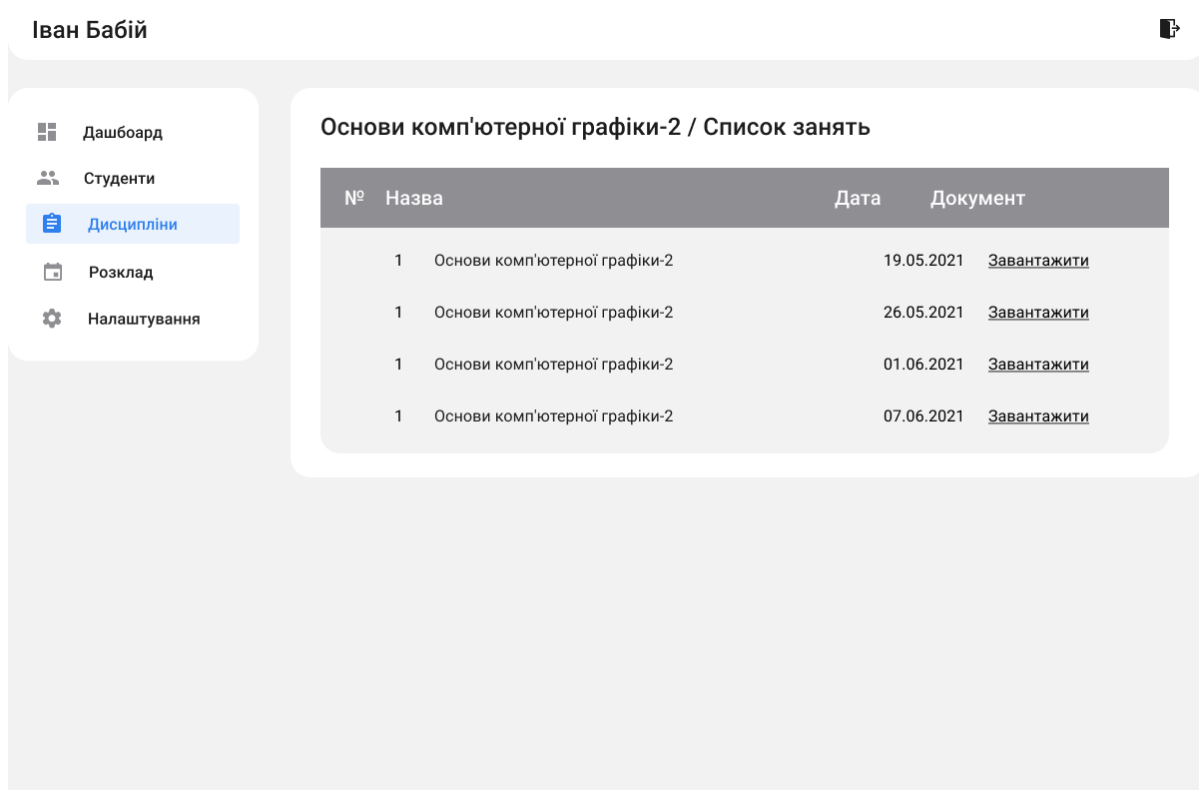
Лекції

Завдання

39

Розділ містить інформацію щодо дисциплін, які належать до групи студента. Кожна з дисциплін має підрозділи Заняття та Завдання.

На сторінці Заняття (рисунок 5.7) зображено список занять, дата проведення та посилання на завантаження допоміжних матеріалів.



Основи комп'ютерної графіки-2 / Список занять			
№	Назва	Дата	Документ
1	Основи комп'ютерної графіки-2	19.05.2021	Завантажити
1	Основи комп'ютерної графіки-2	26.05.2021	Завантажити
1	Основи комп'ютерної графіки-2	01.06.2021	Завантажити
1	Основи комп'ютерної графіки-2	07.06.2021	Завантажити

Рисунок 5.7 — Список занять

Сторінка Завдання (рисунок 5.8) містить перелік завдань, а також дедлайну та кнопку для завантаження посилання на виконаний звіт. Крім цього у списку студент може побачити власну оцінку за зроблену роботу, у випадку, коли адміністратор чи викладач внесе відповідні зміни у систему.

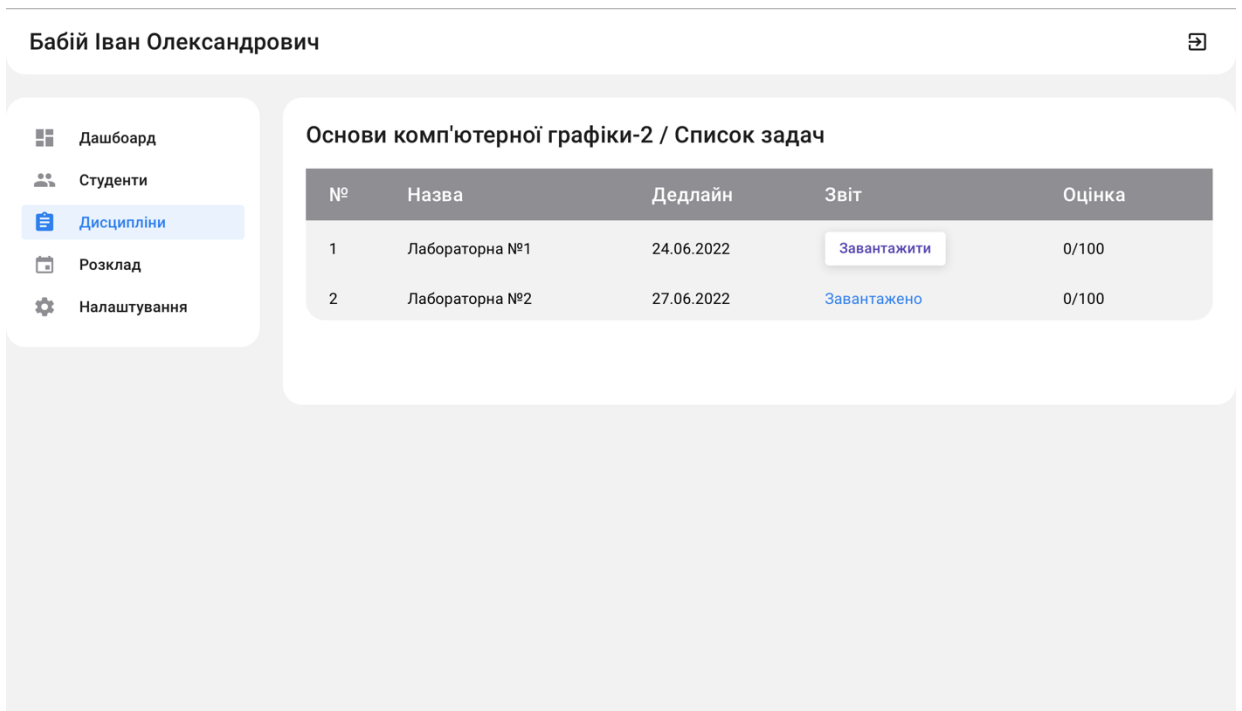


Рисунок 5.8 — Список завдань

Передостаннім розділом у системі для студента є Розклад (рисунок 5.9). У ньому зображений розклад на тиждень. Кожного дня вказано заняття та час проведення.

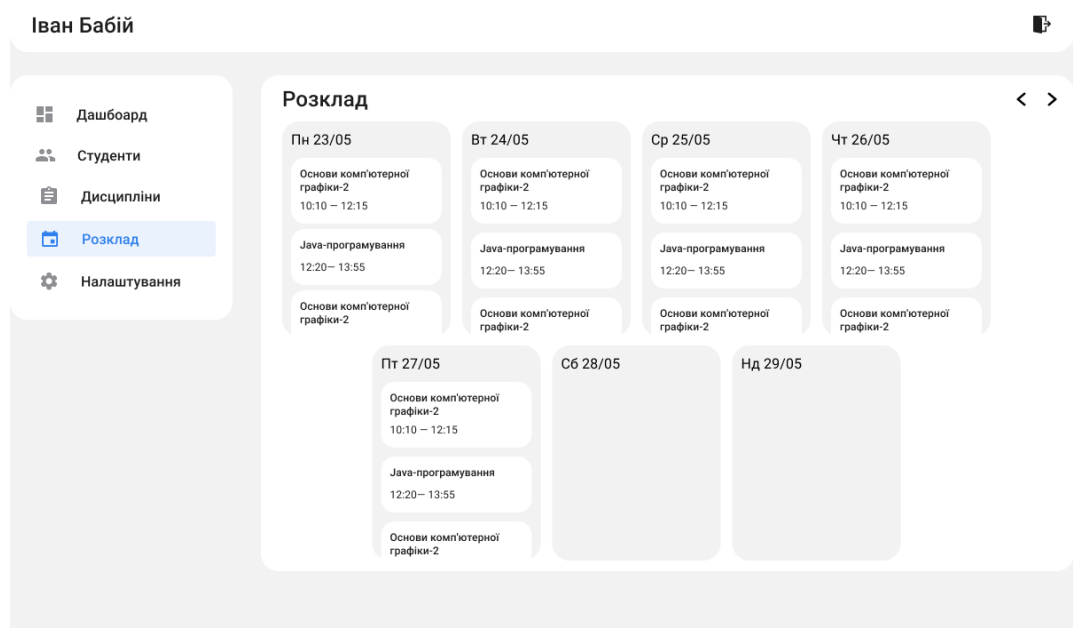


Рисунок 5.9 — Розклад

Останнім розділом є Налаштування (рисунок 5.10). У ньому вказано особисту інформацію студента і посилання на зміну особистих даних, яке веде на електронну пошту адміністратора.

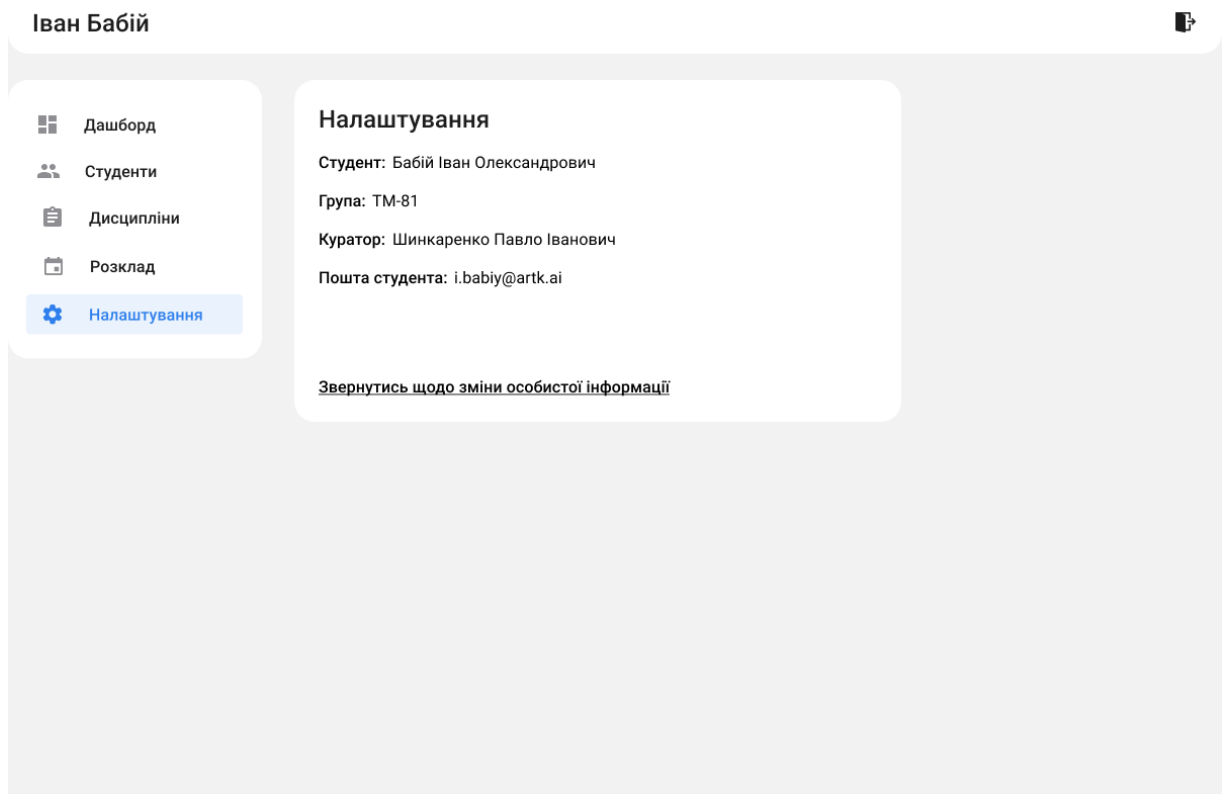


Рисунок 5.10 — Налаштування

Інтерфейс системи студента є індивідуальним рішенням, яке було створено спеціально під програмний продукт.

5.4 Інтерфейс програми для викладача (адміністратора)

Після входу в систему адміністратору відображається список проєктів, які були створені на обліковому записі (рисунок 5.11). Необхідно запустити babiy-diploma, який, власне, і є основою програмного продукту.

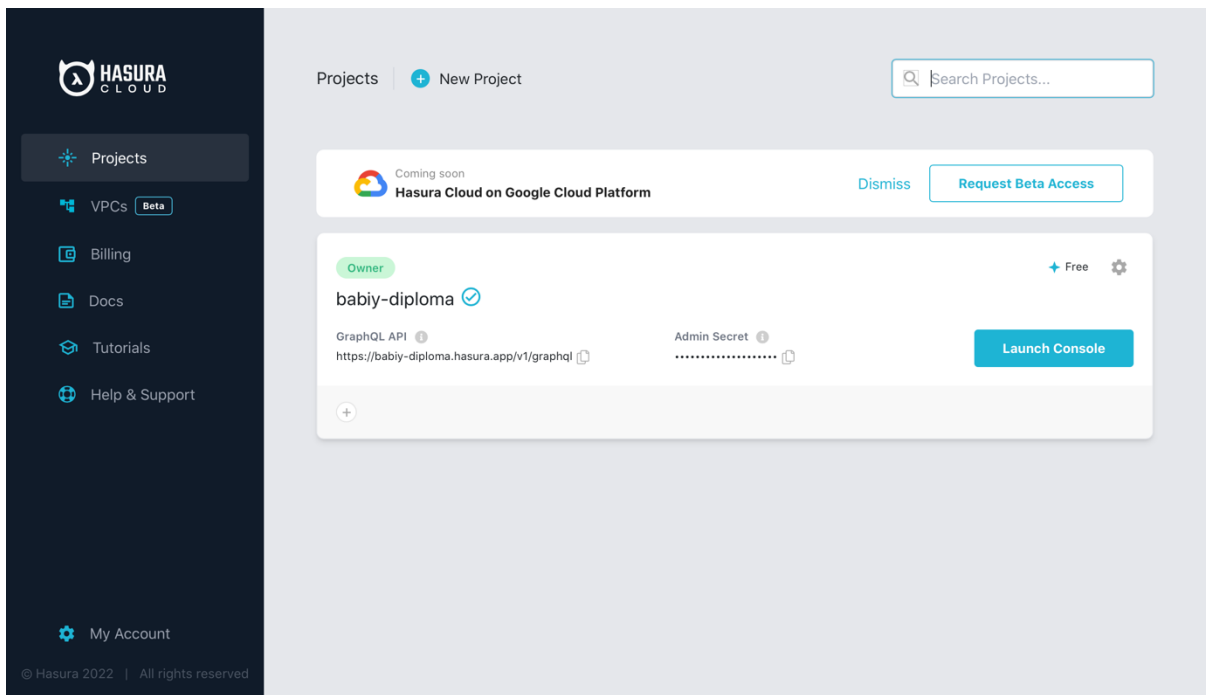


Рисунок 5.11 — Список проєктів у Hasura

Після переходу безпосередньо у проєкт, користувач відкриває доступ до графічного інтерфейсу, у якому детально виражено програмний продукт. Для адміністративного керування системою необхідно перейти у Data Manager, натиснувши на другу іконку зліва у шапці сайту. Після цього відкриється панель керування даними у вебдодатку (рисунок 5.12).

У блоці з лівої сторони сайту показано загальну структуру бази даних та таблиці, які використовуються у вебдодатку. Центральний блок показує у верхній частині шлях, де розташована таблиця, її назву та функції редагування, модифікацій, доступів та зв'язків з іншими таблицями. Основна частина центрального блоку відображає елементи таблиці з ідентифікатором та іншими параметрами.

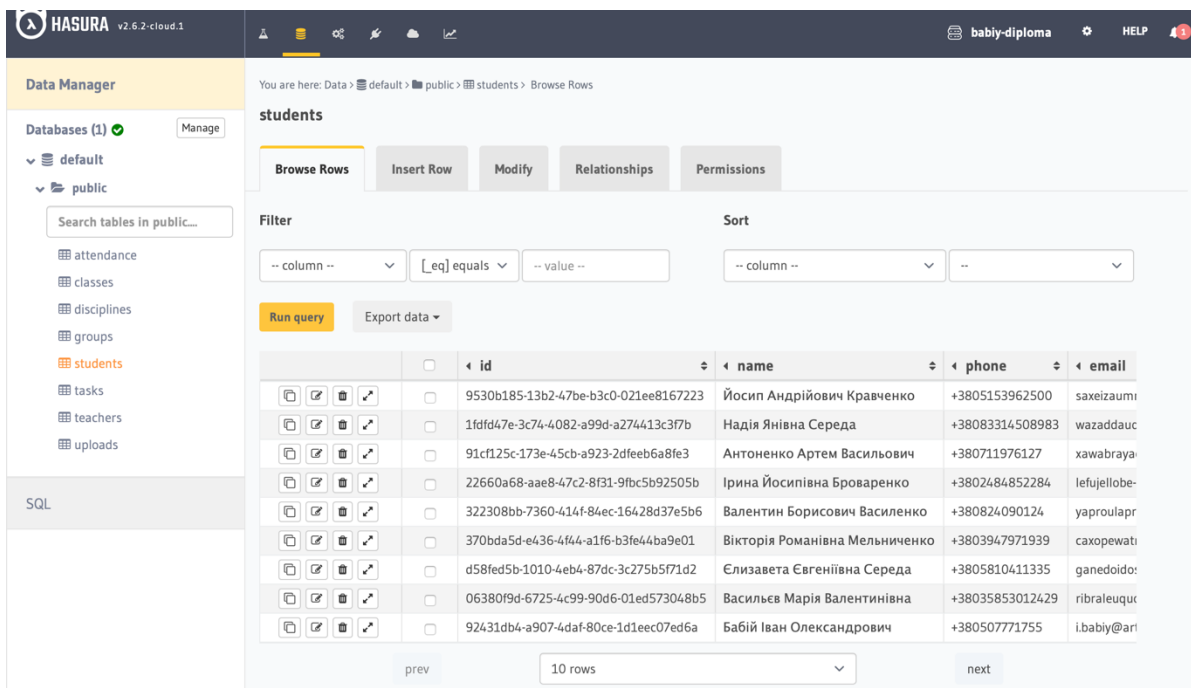


Рисунок 5.12 — Панель керування даними

Розглянемо один з основних варіантів використання інтерфейсу Hasura для адміністратора — внесення нових даних в ту чи іншу таблицю. Для програмного продукту ця функція може бути використана для додавання інформації про нові завдання для студентів, заняття з дисциплін, інформації про вчителів та інших даних, пов'язаних з навчальним процесом. На рисунку 5.13 можна побачити як незмінні блоки інтерфейсу, так і змінений центральний блок з відкритою вкладкою Insert Row. Дана функція призначена для додавання даних у таблицю, у даному випадку у Students. Блок містить поля, що використовуються для опису записів у таблиці. Інтерфейс дозволяє налаштовувати поля на нульові значення, за замовчуванням або жодну з цих налаштувань.

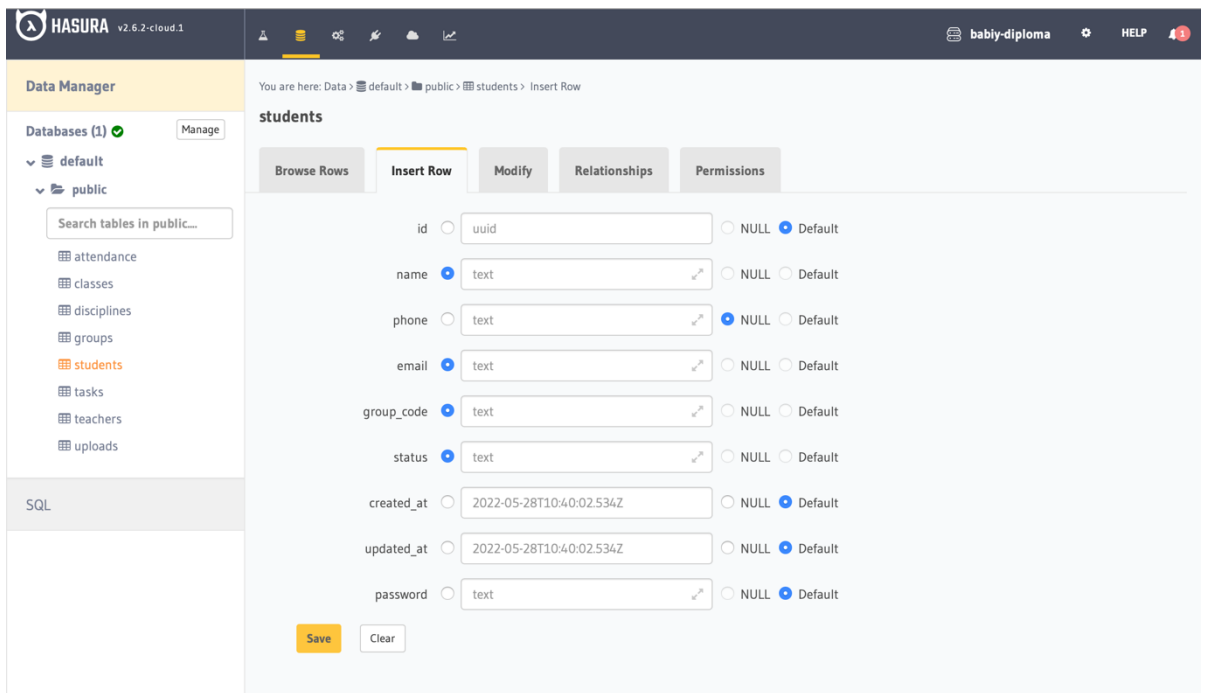


Рисунок 5.13 — Внесення даних в таблицю

Інтерфейс адміністратора має широкий набір функцій які дозволяють гнучко налаштовувати систему, покращувати її та розширяти для оновлення програмного продукту.

5.5 Висновки до розділу

У цьому розділі було детально розглянути користувацькі інтерфейси студента та викладача в ролі адміністратора. Продукт має 2 інтерфейси, залежно від ролі. Кожна з них потребує авторизації та передбачає різний набір функцій. Обидва інтерфейси надають користувачам усі основні можливості, необхідні для контролю навчального процесу.

6 ПЕРСПЕКТИВИ РОЗВИТКУ ПРОГРАМНОГО ПРОДУКТУ

Вебдодаток контролю навчального процесу, як і будь-який інший програмний продукт, має основний реліз, який вміщує у себе усі основні функції, що були закладені на етапі ідеї створення та детально доопрацьовані на етапі реалізації. Попри створений результат кожне цифрове рішення має потенціал для подальшого розвитку.

Наступним глобальним шляхом розвитку одразу після інтеграції продукту у системі вищого навчального закладу може стати створення диференційованих прав та ролей для викладачів та різних рівнів адміністрації. Доступи, які будуть створені, мають бути інтегровані в графічний інтерфейс продукту, який наразі призначений для студентів. У рамках оновлення системи необхідно буде розробити логіку взаємодії користувачів з розширеними правами з інтерфейсом, а також додати нові розділи, які допоможуть детальніше аналізувати навчальні процеси студентів та викладачів, і вимірювати ефективність роботи кожного з користувачів.

Іншою перспективою розвитку продукту є інтеграція системи з сервісами Google. Оскільки дистанційне навчання потребує онлайн-проведення занять, пропонується під'єднати GMeet, який дозволяє створювати онлайн зустрічі. Кожен дзвінок займає унікальне місце в системі, а тому можна приєднати до кожного заняття власне посилання та за допомогою інтерфейсу створити зручний доступ до онлайн заняття. Крім GMeet пропонується використати інтеграцію з сервісом Google Calendar. Це дозволить студенту, який веде список справ у власному календарі, синхронізувати свій розклад з сервісом Google, та ефективніше контролювати свої задачі.

ВИСНОВКИ

У ході розробки вебдодатку контролю навчального процесу було описано основні задачі програмного продукту, виявлено призначення даного рішення. Також вдалось проаналізувати цільову аудиторію, виявити ключові потреби.

За допомогою аналізу конкурентів вдалось визначити предметну область, виявити переваги та недоліки систем КПІ та КНТЕУ. На основі порівняння було обрано рішення, які стали реалізовані у програмному продукті.

Перед початком розробки визначено засоби, які будуть використовуватись при Front-End та Back-End проєктуванні. Також була обрана програмна реалізація продукту, якою стала комплексна база даних.

Наприкінці було зображено сценарії користування системи студентом та викладачем у ролі адміністратора за допомогою створених інтерфейсів системи. На цьому етапі було показано усі можливості системи, які є ключовими для комфортного та ефективного користування вебдодатком.

Окрім цього, описано перспективи розвитку програмного продукту: які є шляхи покращення системи задля більшої ефективності роботи з нею.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. NextJS [Електронний ресурс] — URL для доступу до ресурсу: <https://nextjs.org/learn/> (дата звернення: 20.05.2022).
2. React [Електронний ресурс] — URL для доступу до ресурсу: <https://reactjs.org/docs/> (дата звернення: 20.05.2022).
3. Typescript [Електронний ресурс] — URL для доступу до ресурсу: <https://www.typescriptlang.org> (дата звернення: 20.05.2022).
4. CSS [Електронний ресурс] — URL для доступу до ресурсу: <https://www.w3.org/standards> (дата звернення: 21.05.2022).
5. SWR [Електронний ресурс] — URL для доступу до ресурсу: <https://swr.vercel.app> (дата звернення: 22.05.2022).
6. Yarn [Електронний ресурс] — URL для доступу до ресурсу: <https://yarnpkg.com> (дата звернення: 23.05.2022).
7. Webstorm [Електронний ресурс] — URL для доступу до ресурсу: <https://www.jetbrains.com/webstorm/> (дата звернення: 23.05.2022).
8. Leo S. Hsu, Regina O. Obe PostgreSQL: Up and Running: навчальний посібник, Вид 1-е, 2012 р. 314 с.
9. Heroku [Електронний ресурс] — URL для доступу до ресурсу: <https://www.heroku.com> (дата звернення: 24.05.2022).
10. Hasura [Електронний ресурс] — URL для доступу до ресурсу: <https://hasura.io/> (дата звернення: 24.05.2022).
11. GraphQL [Електронний ресурс] — URL для доступу до ресурсу: <https://graphql.org> (дата звернення: 25.05.2022).
12. JWT [Електронний ресурс] — URL для доступу до ресурсу: <https://jwt.io> (дата звернення: 25.05.2022).

ДОДАТОК А

Вебдодаток контролю навчального процесу

Специфікація

УКР.НТУУ”КП”_ТЕФ_АПЕПС_ТМ81155_22Б

Аркушів 2

Київ — 2022

Позначення	Найменування	Примітки
Документація		
УКР.НТУУ"КПІ"_ТЕ Ф_АПЕПС_ТМ_81155_22Б	Записка.docx	Пояснювальна записка дипломної роботи
Компоненти		
УКР.НТУУ"КПІ"_ТЕ Ф_АПЕПС_ТМ_81155_22Б 12-2	Components	Компонент, що містить блоки на сторінках та зв'язок з базою даних
УКР.НТУУ"КПІ"_ТЕ Ф_АПЕПС_ТМ_81155_22Б1 2-3	Pages	Компонент, що містить стандартні елементи сторінок інтерфейсу студента
УКР.НТУУ"КПІ"_ТЕ Ф_АПЕПС_ТМ_81155_22Б1 2-4	Styles	Компонент, що містить загальні стилі для усього сайту
УКР.НТУУ"КПІ"_ТЕ Ф_АПЕПС_ТМ_81155_22Б1 2-5	Types, Utils	Компоненти, що типізують дані та отримують їх з бази даних

ДОДАТОК Б

Вебдодаток контролю навчального процесу

Текст програми

УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТМ81155_22Б 12-5

Аркушів 11

Київ — 2022

Представлення дисципліни у вигляді об'єкту:

```
import type Lecture from './lecture'
```

```
interface Discipline {  
  id: string  
  title: string
```

```
  classes?: Lecture[]  
}
```

```
export default Discipline
```

Представлення групи у вигляді об'єкту:

```
import type Student from './student'
```

```
import type Teacher from './teacher'
```

```
interface Group {  
  code: string  
  year_created: number  
  created_at: string  
  updated_at: string  
  curator_id: string  
  curator: Teacher  
  headman_id: string  
  headman?: Student  
  students?: Student[]  
}
```

```
export default Group
```

Представлення лекції у вигляді об'єкту:

```
import type Teacher from './teacher'
```

```
interface Lecture {  
  id: string  
  title: string  
  type: string  
  start_time: string  
  end_time: string  
  link?: string  
  teacher?: Teacher  
}  
  
export default Lecture
```

Представлення студента у вигляді об'єкту:

```
import type Group from './group'  
  
interface Student {  
  id: string  
  name: string  
  phone: string  
  email: string  
  group_code: string  
  group?: Group  
  status: string  
  created_at: string  
  updated_at: string  
}  
  
export default Student
```

Представлення задачі у вигляді об'єкту:

```
interface Task {  
  id: string  
  title: string  
  description: string  
  deadline: string
```

```

uploads?: {
  link: string
  mark: string
}[]
class?: {
  id: string
  title: string
  discipline?: {
    id: string
    title: string
  }
}
}
}

export default Task

```

Представлення викладача у вигляді об'єкту:

```

interface Teacher {
  id: string
  name: string
  phone: string
  email: string
}

export default Teacher

```

Представлення користувача у вигляді об'єкту:

```

import type Group from './group'

type User = {
  token?: string
  id?: string
  name?: string
  group?: Group
  curator?: string
}

```

```
  email?: string
}

export default User
```

Отримання та вивід даних про заняття:

```
import useSWR from 'swr'
import axios from 'axios'
import type Discipline from 'types/discipline'

const getDisciplineLectures = async (id: string, groupCode: string) => {
  if (!id || !groupCode) return null
  const res = await axios.get<Discipline>(`/api/disciplineLectures?id=${id}&code=${groupCode}`)
  return res?.data
}

const useDisciplineLectures = (id: string, code: string) => {
  const { data, ...hookProps } = useSWR<Discipline>(['lectures', id, code], () =>
    getDisciplineLectures(id, code))

  return { ...hookProps, data, discipline: data, lectures: data?.classes }
}

export default useDisciplineLectures
```

Отримання та вивід даних про завдання:

```
import useSWR from 'swr'
import axios from 'axios'
import type Discipline from 'types/discipline'
import type Task from 'types/task'

interface QueryData {
  discipline: Discipline
  tasks: Task[]
}
```

```

const getDisciplineTasks = async (id: string, groupCode: string) => {
  if (!id || !groupCode) return null
  const res = await axios.get<QueryData>(`/api/disciplineTasks?id=${id}&code=${groupCode}`)
  return res?.data
}

const useDisciplineTasks = (id: string, code: string) => {
  const { data, ...hookProps } = useSWR<QueryData>(['tasks', id, code], () => getDisciplineTasks(id, code))

  return { ...hookProps, data, discipline: data?.discipline, tasks: data?.tasks }
}

export default useDisciplineTasks

```

Отримання та вивід даних про дисципліну:

```

import useSWR from 'swr'
import axios from 'axios'
import type Discipline from 'types/discipline'

const getDisciplines = async (groupCode: string) => {
  if (!groupCode) return null
  const res = await axios.get<Discipline[]>(`/api/disciplines?code=${groupCode}`)
  return res?.data
}

const useDisciplines = (groupCode: string) => {
  const { data, ...hookProps } = useSWR<Discipline[]>(['disciplines', groupCode], () => getDisciplines(groupCode))

  return { ...hookProps, data, disciplines: data }
}

export default useDisciplines

```


Отримання та вивід даних про групу:

```
import useSWR from 'swr'
import axios from 'axios'
import type Group from 'types/group'

const getGroup = async (groupCode: string) => {
  if (!groupCode) return null
  const res = await axios.get<Group>(`/api/group?code=${groupCode}`)
  return res?.data
}

const useGroup = (groupCode: string) => {
  const { data, ...hookProps } = useSWR<Group>(['group', groupCode], () => getGroup(groupCode))
  return { ...hookProps, data, group: data }
}

export default useGroup
```

Отримання даних про загальний розклад:

```
import { useMemo } from 'react'
import useSWR from 'swr'
import axios from 'axios'
import type Lecture from 'types/lecture'

const getScheduleData = async (groupCode: string, start: string, end: string) => {
  if (!groupCode) return null
  const res = await
  axios.get<Lecture[]>(`/api/schedule?code=${groupCode}&start=${start}&end=${end}`)
  return res?.data
}

function addDays(date: string | Date, days: number) {
  const result = new Date(date)
  if (!days) return result
  result.setDate(result.getDate() + days)
  return result
}
```

```

function getDateByWeek(week: number, year: number) {
  // Create a date for 1 Jan in required year
  const d = new Date(year, 0)
  // Get day of week number, sun = 0, mon = 1, etc.
  const dayNum = d.getDay()
  // Get days to add
  let requiredDate = --week * 7

  // For ISO week numbering
  // If 1 Jan is Friday to Sunday, go to next week
  if (dayNum > 0 || dayNum > 4) {
    requiredDate += 7
  }

  // Add required number of days
  d.setDate(2 - d.getDay() + ++requiredDate)
  d.setHours(0, 0, 0, 0)

  return d
}

export interface Day {
  date: string
  classes: Lecture[]
}

interface useScheduleProps {
  code: string
  week: number
  year: number
}

const useSchedule = ({ code, week, year }: useScheduleProps) => {
  const date = getDateByWeek(week, year)
  const start = date.toISOString()
  const end = addDays(date, 7).toISOString()

  const { data, ...hookProps } = useSWR<Lecture[]>(['students', code, start, end], () =>
    getScheduleData(code, start, end))

  const schedule = useMemo(() => {
    if (!data) return null

```

```

const days: Day[] = [...Array(7).keys()].map((i) => ({
  date: addDays(start, i).toISOString().slice(0, 10),
  classes: [],
})))

data.forEach?.((lecture) => days.find((day) =>
lecture?.start_time?.startsWith?.(day.date))?.classes.push(lecture))

// @ts-ignore
days.forEach((day) => day.classes.sort((lecture1, lecture2) => new Date(lecture1.start_time) - new
Date(lecture2.start_time)))

return days
}, [data, start])

return { ...hookProps, data, schedule }
}

export default useSchedule

```

Отримання даних про завдання:

```

import useSWR from 'swr'
import axios from 'axios'
import type Task from 'types/task'

const getTasks = async (id: string) => {
  if (!id) return null
  const res = await axios.get<Task[]>(`/api/tasks?id=${id}`)
  return res?.data
}

const useTasks = (userId: string) => {
  const { data, ...hookProps } = useSWR<Task[]>(['tasks', userId], () => getTasks(userId))

  return { ...hookProps, data, tasks: data }
}

export default useTasks

```

Отримання даних про розклад на сьогодні:

```
import useSWR from 'swr'
import axios from 'axios'
import type Lecture from 'types/lecture'

const getTodaySchedule = async (groupCode: string) => {
  if (!groupCode) return null
  const res = await axios.get<Lecture[]>(`/api/today?code=${groupCode}`)
  return res?.data
}

const useToday = (groupCode: string) => {
  const { data, ...hookProps } = useSWR<Lecture[]>(['today', groupCode], () =>
    getTodaySchedule(groupCode))

  return { ...hookProps, data, lectures: data }
}

export default useToday
```

Перевірка на авторизацію користувача:

```
import { useEffect, useCallback } from 'react'
import Router from 'next/router'
import useSWR from 'swr'
import axios from 'axios'
import type User from 'types/user'

function useUser({ redirectTo = "", redirectIfFound = false } = {}) {
  const { data: user, mutate: mutateUser, error } = useSWR<User>('/api/user')

  const isLoggedIn = !!user?.token

  useEffect(() => {
    // if no redirect needed, just return (example: already on /dashboard)
    // if user data not yet there (fetch in progress, logged in or not) then don't do anything yet
    if (!redirectTo || !user) return
```

```

    if (
      // If redirectTo is set, redirect if the user was not found.
      (redirectTo && !redirectIfFound && !isLoggedIn) ||
      // If redirectIfFound is also set, redirect if the user was found
      (redirectIfFound && isLoggedIn)
    ) {
      Router.push(redirectTo)
    }
  }, [user, redirectIfFound, redirectTo, isLoggedIn])

  const logOut = useCallback(async () => mutateUser(await axios.post('/api/logout', false)),
    [mutateUser])

  const hasError = !!error
  useEffect(() => {
    if (hasError) {
      localStorage.clear()
      sessionStorage.clear()
      logOut()
    }
  }, [hasError, logOut])

  return { user, isLoggedIn, mutateUser, logOut }
}

export default useUser

```

ДОДАТОК В

Вебдодаток контролю навчального процесу

Опис програми

УКР.НТУУ”КП”_ТЕФ_АПЕПС_ТМ81155_22Б

Аркушів 9

Київ — 2022

АНОТАЦІЯ

Додаток описує програмний продукт вебдодаток контролю навчального процесу, що працює з наступними даними:

- заняття протягом навчального семестру та дисципліни, які має студент
- звіти завдань, які відправляють студенти на оцінювання;
- розклад занять;
- списки груп та викладачів;
- контактна інформація студентів.

Back-end частина була написана з використанням NodeJS та GraphQL API для роботи з базою даних.

ЗМІСТ

1 ЗАГАЛЬНІ ВІДОМОСТІ	65
2 ФУНКЦІОНАЛЬНЕ ПРИЗНАЧЕННЯ	66
3 ОПИС ЛОГІЧНОЇ СТРУКТУРИ.....	67
4 НЕОБХІДНІ ТЕХНІЧНІ ЗАСОБИ	68
5 ВХІДНІ ДАНІ.....	69
6 ВИХІДНІ ДАНІ.....	70

1 ЗАГАЛЬНІ ВІДОМОСТІ

Додаток описує основні компоненти вебдодатку контролю навчального процесу. Додаток В містить наступні розділи: функціональне призначення, логічну структуру, необхідні технічні засоби для користування продуктом, а також вхідні та вихідні дані. Додаток Б містить програмний код компонентів, що використовуються.

Мінімальними системними вимогами для використання програмного продукту є: операційна система Windows 7, Mac OS 10.5 чи Linux 5, 2GB RAM, процесор Intel Core 2 Duo, 10GB HDD чи SSD, відеокарта Radeon HD 8450 та доступ до мережі Інтернет.

Back-end вебдодатку розроблений на NextJS з використанням GraphQL для зв'язку з базою даних.

2 ФУНКЦІОНАЛЬНЕ ПРИЗНАЧЕННЯ

Вебдодаток підтримує для користування наступні функції:

- перегляд розкладу занять на день;
- перегляд списку невиконаних завдань;
- перегляд списку студентів групи;
- перегляд списку дисциплін у навчальному семестрі;
- перегляд списку лекцій з можливістю завантажити матеріали до них;
- перегляд списку завдань з можливістю відправити звіт завдання та переглянути оцінку;
- перегляд розкладу на тиждень;
- перегляд особистої інформації студентом.

3 ОПИС ЛОГІЧНОЇ СТРУКТУРИ

Архітектура програмного забезпечення поділяється на обробку даних, які описані в базі даних на сервері Hasura та, відповідно, зображенням даних в інтерфейсі користувача.

Система влаштована таким чином, що обробкою даних є отримання інформації з бази та її структуризація для подальшої роботи. Передаються дані щодо дисциплін, занять, завдань, викладачів, студентів, груп, а також розкладу.

Відображення інформації відбувається за допомогою отримання даних з бази за допомогою GraphQL API у реальному часі.

4 НЕОБХІДНІ ТЕХНІЧНІ ЗАСОБИ

Для користування вебдодатком контролю навчального процесу пристрою в першу чергу необхідний доступ до мережі Інтернет. Стосовно операційної системи, для коректної роботи мінімальні версії мають бути Windows 7, Mac OS 10.5 чи Linux 5. Щодо технічного забезпечення, пристрій повинен мати якнайменше процесор Intel Core 2 Duo, 10GB HDD чи SSD та відеокарту Radeon HD 8450.

Програмний продукт підтримується усіма останніми версіями браузерів, а також Internet Explorer версії 11+. Продукт підтримує ноутбуки та комп'ютери з екранним розширенням вище 1200 пікселів.

5 ВХІДНІ ДАНІ

Вхідна інформація системи контролю навчального процесу для студента:

- ID дисципліни;
- ID заняття;
- ID завдання;
- ID групи;
- ID студента;
- ID викладача;
- Назва дисципліни;
- Назва заняття;
- Назва завдання;
- Назва групи;
- Ім'я студента;
- Ім'я викладача;
- Тип заняття;
- Дата та час початку заняття;
- Дата та час закінчення заняття;
- Кінцева дата здачі завдання;
- Рік створення групи;
- Номер телефону студента;
- Електронна пошта студента;
- Статус студента;
- Оцінка з завдання;
- Посилання на матеріали заняття;
- Тип завантаження звіту завдання;
- Посилання на звіт завдання.

6 ВИХІДНІ ДАНІ

Вихідна інформація системи контролю навчального процесу для студента:

- Назва дисципліни;
- Назва заняття;
- Назва завдання;
- Назва групи;
- Ім'я студента;
- Ім'я викладача;
- Тип заняття;
- Час початку заняття;
- Час закінчення заняття;
- Дата проведення заняття;
- Номер телефону студента;
- Електронна пошта студента;
- Статус студента;
- Ім'я куратора;
- Посилання на матеріали заняття;
- Дедлайн завдання;
- Оцінка з завдання.