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

ПРОЄКТУВАННЯ ТА РОЗРОБЛЕННЯ СЕРВІСНОЇ АРХІТЕКТУРИ УПРАВЛІННЯ БІЗНЕС-ПРОЦЕСАМИ УНІВЕРСИТЕТУ. СЕРВІС "ДЕКАНАТ"

Кваліфікаційна робота (проєкт)
на здобуття ступеня вищої освіти "магістр"

Виконав: студент 2 курсу
Спеціальності 126 Інформаційні системи та технології
Освітньо-професійної програми
Інформаційні системи та технологій другого (магістерського) рівня вищої освіти
Попов Сергій Андрійович
Керівник: доктор педагогічних наук,
професор Круглик Владислав Сергійович,
Кандидат фізико-математичних наук,
доцент Єрмолаєв Вадим Анатолійович
Рецензент: кандидат фізико-математиних наук, доцент
Плоткін Яків Давидович

Херсон – 2020

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА ТЕРМІНІВ

1. REST— підхід до органі-зації API, передача репрезентативного стану системи (Representational state transfe).
2. SQL— мова запитів до реляційних баз даних (Structured Query Language).
3. JSON— текстовий формат обміну даними на основі синтаксису ECMAScript (Javascript object notation).
4. JWT—коротке повідомлення з криптографічними перетвореннями (JSON Web Token).
5. API— програмний інтерфейс додатку (application programming interface)
6. HTML— мова розмітки документів (Hypertext Markup Language).
7. HTTPS— захищений протокол передачі даних прикладного рівня (Hypertext Transfer Protocol Secure).
8. UX— враження які отримав користувач при використанні інтерфейсу (User Experience).
9. UI— який вигляд має система, та її властивості (User Interface).
10. Авторизація— підтвердження користувачем права на виконання тих чи інших дій.

ВСТУП

Актуальність роботи полягає в тому що в останні роки потреба у власній системі для управління бізнес-процесами університету набула потреби як на території України, так і в усьому світі.

Все це пов’язано із розвитком інформаційно-комунікаційних технологій, система університету починає відігравати надзвичайно важливу роль в житті вищого навчального закладу. До таких мереж залучено дуже велику кількість студентів.

Об’єктом дослідження є бізнес-процес управління деканату Херсонського державного університету

Предметом дослідження є сервісна архітектура управління бізнес-процесами університету. Сервіс "Деканат", мікросервіси “Розклад”, “Журнал”, “Залікова книжка”, “Відомість” та “Індивідуальний навчальний план успішності студента”.

Метою дослідження є проєктування сервісної архітектури управління бізнес-процесами університету та розробка сервісу “Деканат”

Завдання:

- Дати аналіз комерційних додатків та систем, які використовуються в інших університетах;
- Дати аналіз структури деканату;
- Спроектувати сервісну архітектуру управління бізнес-процесами університету;
- Поставити завдання на проектування модулів мікросервісів для сервісу “Деканат”;
- Розробити сервіс “Деканат” для системи управління бізнес-процесами університету;

Розроблений сервіс може використовуватись по прямому призначенню;

Методи дослідження: під час роботи було використано теоретичні та емпіричні методи дослідження. Використовувались такі теоретичні методи дослідження:

- аналіз,
- порівняння,
- узагальнення,
- моделювання.

Наукова новизна одержаних результатів: результати показали, що розроблений сервіс може повноцінно використовуватися в освісньому процесі в ХДУ, але сервіс потребує подальшого вдосконалення.

Практичне значення одержаних результатів:

1. На основі визначеного функціоналу спроектовано сервіс “Деканат” для сайта управління бізнес-процесами університету “ХДУ24”.
2. Розроблено мікрсервіси “Розклад навчальних занять”, “Електронний журнал”, “Елктронна залікова книжка”, “Електрона відомість” для сайта “ХДУ24”.

Розроблені мікросервіси можуть використовуватись по прямому призначенню.

Апробація результатів досліджень: Сервіс “Деканат” знаходиться на серверах університету ХДУ [17] на вкладці “Розклад навчальних занять”. Планується публікація статті до видавництва Springer в 2021 році.

РОЗДІЛ 1
Загальні відомості про організацію сервісів для управління бізнес-процесами університета.

1.1 Огляд наявних сервісів для управління бізнес-процесами університета

SmartGate - це IT-рішення для формування єдиної екосистеми освітніх сервісів університету який був розроблений компанією IBS. Даний сервіс дозволяє підвищити ефективність навчального процесу за допомогою мобільних технологій. Технологія дає можливість інтегрувати всі існуючі в вузі електронні системи і сервіси, а також доставляти необхідну інформацію на пристрої користувачів в зручному форматі та інше.

Вартість розгортання даної системи понад 400 тис. грн. Час розгортання системи від 6 місяців до 2 років.

WSB University - це сервіс для управління бізнес-процесами університету який був розроблений польським університетом WSB що знаходиться в місті Хожув.

Дана система не є комерційною.

IStudent - це мобільний додаток для управління процесами навчання з простим інтерфейсом.

Дана система не є комерційною.

Таблиця 1.1

Порівняння програмних компонентів сервісів.

	WSB UNIVERSITY	**IStudent**
Frameworks	1. ASP.NET Ajax 2. Telerik Controls	1. CodeIgniter 2. Bootstrap 3.3.7

Продовження таблицы 1.1

developed by	1. Java Script	1. PHP 5.5.9
Libraries and Functions	1. jQuery 2. jQuery UI	1. Moment.js 2. jQuery
Web Server	1. IIS 7	1. Nginx 1.4.6
Operating System	1. ---	1. Ubuntu

Service functionality	1. Student
	  1. General info
	  2. University status
	  3. Student’s record
	  4. List of teachers
	  5. Schedule
	  6. Exam dates
	2. Student office
	  1. Financial data
	  2. Amount paid
	3. Messages
	  1. Email to dean's office
	4. E-learning
	5. Questionnaires and Exams
	  1. Questionnaire
	  2. Exams
		1. General info
		2. Schedule
		3. Student’s record
		4. Financial data

iGradePlus - це система для управління школою, розроблене для

індивідуальних викладачів, шкіл і коледжів. Основні функції системи є управління учнями, класами та управління школами. Система була розроблена компанією Capterra.

Вартість розгортання даної системи понад 700 тис. грн. Час розгортання системи від 1 до 2 років.

CampusAnyware - це інформаційна система для студентів, що охоплює весь життєвий цикл учня, та повністю розміщена в хмарному середовищі.

Вартість розгортання даної системи 100 тис. грн.

	Управління навчальними матеріалами	Управління оцінкою	Управління бібліотекою	Управління кампусом	Управління навчальними програмами	Управління фінансовою допомогою	Спеціальна освіта	Онлайн-платежі	Управління доступом	Фінансовий менеджмент	Портал для батьків / учнів	Управління випускниками	Факультет / Управління персоналом	Управління збору коштів	Планування	Інформація для студентів / записи	Студентський портал	Інструменти зв'язку	Календар подій	Інтерактивне навчання	Опитування / Голосування	Управління безпекою	Управління студентською групою	
WSB	*	*	*	-	*	*	*	*	*	*	-	*	*	*	*	*	-	*	*	-	*	*	-	
IStudent	-	*	-	-	-	*	-	-	-	-	-	-	-	-	-	*	-	-	*	*	-	-	-	
SmartGate	*	*	*	*	*	-	-	-	*	*	-	*	*	-	-	*	*	-	-	*	-	-	*	
iGradePlus	-	*	*	*	-	-	*	-	-	*	-	-	*	*	-	-	-	*	-	-	-	*	*	
CampusAnyware	*	-	*	*	*	*	*	*	-	*	*	*	-	*	*	*	*	*	*	*	*	*	*	

Рисунок 1.1 - Матриця порівняння існуючих систем

Проаналізувавши дані системи ми дійшли до висновоку, що система WSB та IStudent не задовольняє повний спектор вимог які потрібні в університеті. Система SmartGate, IGeadePlus та CampusAnyware не задовольняє нас через високої вартості розгортання цих систем, та через неповний спектр функціоналу який необхідний для університету.

Тому нами було прийнято рішення в необхідності розробки власної системи для управління бізнес-процесами університета.

1.2 Порівняння програмних засобів для розробки системи

Найпоширенішою мовою програмування веб-додатків є “PHP”. PHP використовують близько 78% всіх сайтів. Мова з'явився в 1995 році, коли було не так багато можливостей для створення динамічних веб-сторінок. Велика різноманітність функцій PHP дає можливість уникати написання багаторядкових функцій, призначених для користувача. Мова PHP здаватиметься знайомою програмістам, що працюють в різних областях. Багато конструкцій мови запозичені з C, Perl. Код PHP дуже схожий на той, який зустрічається в типових програмах.

Компанії які використовують PHP:

Facebook, Lyft, Mint, Hootsuite, Viber, Buffer, DocuSign.

Java Script - мова програмування, котру можна використовувати як для фронтенда, так і для бекенд. Ця мова підійде для початківців, оскільки в ній мало налаштувань, і можна почати писати код прямо в браузері.

Значно розширює функціонал JS програмна платформа Node.js. З її допомогою код, написаний на JS, можна запускати без браузера на бекенд. А наявність величезної кількості готових рішень npm дозволяє розробнику не витрачати час на створення більшості типових рішень.

З JavaScript створюються веб-додатки і програмне забезпечення, а саме:

- Фронтенд і бекенд веб-сайтів;
- Мобільні додатки;
- Веб-сервери;

Компанії які використовують JavaScript:

Airbnb, Codecademy, HotelTonight, eBay, Square, Asana.

Python - популярна мова програмування. Він використовується як для веб-розробки, так і для створення настільних додатків. У Мережі можна знайти величезну кількість навчальних сайтів, навчальних посібників по Python.

Синтаксис Python простий і легкий для розуміння в порівнянні з іншими мовами. Python підтримує об'єктно-орієнтоване, функціональне і аспектно-орієнтоване програмування, а також це динамічно типізований мову з відкритим вихідним кодом.

Застосування Python:

- Кросплатформні shell-скрипти;
- Веб розробка;
- Data Science, Machine Learning

Компанії які використовують Python:

Uber, Pinterest, Mozilla, Spotify, Quora, Pandora, Netflix, Asana.

GO (Golang) - компільована мова програмування, розроблена всередині компанії Google. Мова Go розроблялася для створення високоефективних програм, що працюють на сучасних розподілених системах і багатоядерних процесорах.

Go - мова з суворою статичною типізацією. Має досить лаконічний і простий синтаксис, заснований на основі мови C, але суттєво доопрацьована.

GO дозволяє:

- Розробляти ПЗ для розподілених систем;
- Писати ПЗ, котрий може обробляти великі обсяги інформації.

Компанії які використовують GO:

Dropbox, SoundCloud, Docker, CloudFlare, Alibaba, Xiaomi, Domino’s Pizza, Gett, Avito, Mail.ru, Tinkoff, 2GIS.

Ruby + Ruby on Rails - це веб-фреймворк на мові програмування Ruby. Ruby on Rails має набір готових інструментів, які дозволяють швидко виконувати базові завдання.

Ruby досить лаконічний і не вимагає багато коду для бекенд, що дозволяє розробникам швидко створювати і запускати прототипи. Популярність Ruby виросла напочатку 2000 років, але з тих пір помітно

знизилася.

Ruby - мова з відкритим вихідним кодом, а значить її можна модифікувати і доповнювати.

Ruby дозволяє:

- Автоматизувати повторювані завдання;
- Створювати веб-додатки;
- Писати мобільні додатки та ігри;
- Створювати прототипи.

Компанії які використовують Ruby + Ruby on Rails

Airbnb, Codecademy, TaskRabbit, Kickstarter, RapGenius, Scribd, AngelList.

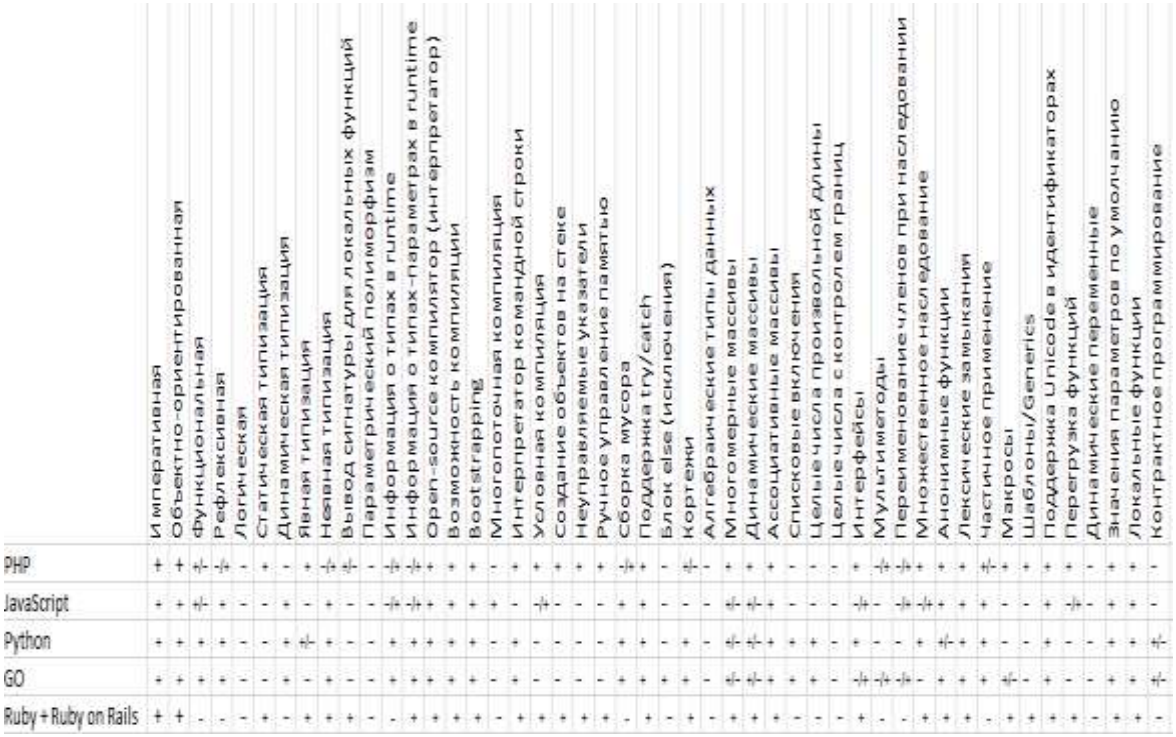

Рисунок 1.2 - Матриця порівняння програмних засобів для розробки системи

Проаналізувавши програмні засоби, ми дійшли до висновку, що для розробки нашої системи ми будемо використовувати мову програмування Python, через те що вона задовольняє поставленим завданням для реалізації системи університету.

1.3 Обґрунтування функціоналу сервісу

Проаналізувавши існуючі додатки, ми прийшли до висновків щодо потреби розробки власної системи управління бізнес-процесами університету.

Ми спроектували функціональні потреби для власного сервісу:

1. Особистий кабінет студента.
2. Сервіс “Деканат”.
   - 2.1. Мікросервіс “Розклад навчальних занять”.
     - 2.1.1. Формування розкладу занять.
     - 2.1.2. Формування розкладу контрольних занять (іспитів, диференційованих заліків, заліків).
     - 2.1.3. Можливість додавання занять у ZOOM.
     - 2.1.4. Редагування розкладу навчальних занять (аудиторій, викладачів).
   - 2.2. Мікросервіс “Електронний журнал”.
   - 2.3. Мікросервіс “Електронна залікова книжка”.
   - 2.4. Мікросервіс “Електронна відомість”.
     - 2.4.1. Форма контролю.
   - 2.5. Мікросервіс “Індивідуальний навчальний план успішності студента”.
     - 2.5.1. Статистичні результати успішності студента.
3. Сервіс “Відділ кадрів”.
   - 3.1. Облік інформації про кількісний та якісний склад кадрів університету.
   - 3.2. Облік звітно-облікової документації (у тому числі наказів з особових питань).
   - 3.3. Облік, заповнення і зберігання трудових книжок.
   - 3.4. Облік, заповнення і зберігання особових справ і особових карток працівників.
   - 3.5. Надання державної статистичної звітності з кадрових питань.

3.6. Надання довідок про поточну та попередню трудову діяльність працівників.

3.7. Обчислення трудового стажу працівників.

3.8. Контроль за встановленням доплат (надбавок) за вислугу років (якщо вони передбачені положенням про оплату праці).

3.9. Облік відпусток працівників.

3.10. Облік особових справ студентів усіх форм навчання.

3.11. Табельний облік працівників відділу кадрів, ректорату.

3.12. Підготовка для Міністерства освіти і науки України штатного формуляра ХДУ.

3.13. Робота в Інформаційно-аналітичній системі університету «Персонал» та «Контингент».

3.14. Оформлення листків непрацездатності.

3.15. Оформлення особових справ студентів, які переводяться до ХДУ з інших вищих навчальних закладів.

3.16. Робота з особовими справами студентів отримані від приймальної комісії.

3.17. Здійснення організаційних заходів щодо своєчасного щорічного подання відповідними посадовими особами ХДУ декларацій про майно, доходи, витрати і зобов'язання фінансового характеру.

4. Сервіс "Навчальний відділ".

4.1. Участь у розробці та затвердженні навчальних та робочих навчальних планів.

4.2. Контроль за складанням розкладів аудиторних занять, організації самостійної роботи студентів, заліків, семестрових та державних екзаменів.

4.3. Аналіз організації та здійснення навчального процесу, результатів комплексної перевірки знань студентів з предметів та проходження практик, самостійної роботи, виконання курсових та

дипломних робіт.

4.4. Складання графіків навчального процесу.

4.5. Контроль за плануванням та виконанням навчального навантаження професорсько-викладацького складу згідно навчальних планів денної, заочної, екстернатної форм навчання.

4.6. Підготовка, замовлення, отримання, видача студентських квитків та дублікатів студентських квитків.

4.7. Підготовка, замовлення, отримання, видача документів про освіту (дипломів) та дублікатів дипломів.

4.8. Підготовка, замовлення, отримання, видача додатків до дипломів та дублікатів додатків до дипломів.

4.9. Видача академічних довідок.

4.10. Робота з Єдиною державною електронною базою з питань освіти.

4.11. Підготовка акредитаційних та ліцензійних справ спільно з випусковими кафедрами.

4.12. Облік навчальної документації згідно з вимогами наказу Міністерства освіти і науки, молоді та спорту України від 29.03.2012 року № 384.

4.13. Контроль за виконанням індивідуальних планів викладачів.

4.14. Складання статистичних звітних даних за стандартними формами.

4.15. Контроль за дотриманням вимог чинного законодавства з питань відрахування, поновлення, надання академічної відпустки, індивідуального графіку навчання, отримання документів про навчання в ХДУ.

4.16. Ведення документації щодо руху контингенту студентів.

4.17. Спільно з відділом кадрів підготовка документації та отримання і видача документів про освіту.

4.18. Аналіз кадрових паспортів кафедр.

5. Сервіс "ВЗЯВО"
   5.1.Головна сторінка сервісу.
   5.2. Сервіс “Анкетування”.
      5.2.1. Створення анкетувань.
      5.2.2. Створення анонімних анкетувань.
      5.2.3. Редагування анкетувань.
      5.2.4. Перегляд поточних результатів.
   5.3.Результати.
      5.3.1. Перелік результатів опитування.
      5.3.2. Статистичні результати.
      5.3.3. Якість опитування.
6. Відділ безпеки сервісу.
7. Система логування.
8. Сервіс “імпорту-експорту”.
9. Розробка фронтенд частини сервісу.
10. API.
    10.1. Реалізація serializers.
    10.2. Swagger специфікація.
    10.3. Postman колекція запитів для взаємодії.
    10.4. Інтеграція з системою менеджменту логування.
    10.5. Інтеграція сервісу імпорту-експорту.
11. Створення структури взаємодії бекенд частини із фронтенд частиною сервісу.

РОЗДІЛ 2

Проєктування сервісної архітектури управління бізнес-процесами університету

2.1 Опис структури класів сервісної архітектури управління бізнес-процесами університету

При проєктуванні нашої системи ми зробили опис класів, які будуть в нашій системі. Загальний вигляд нашої системи зображений на рисунок 2.1

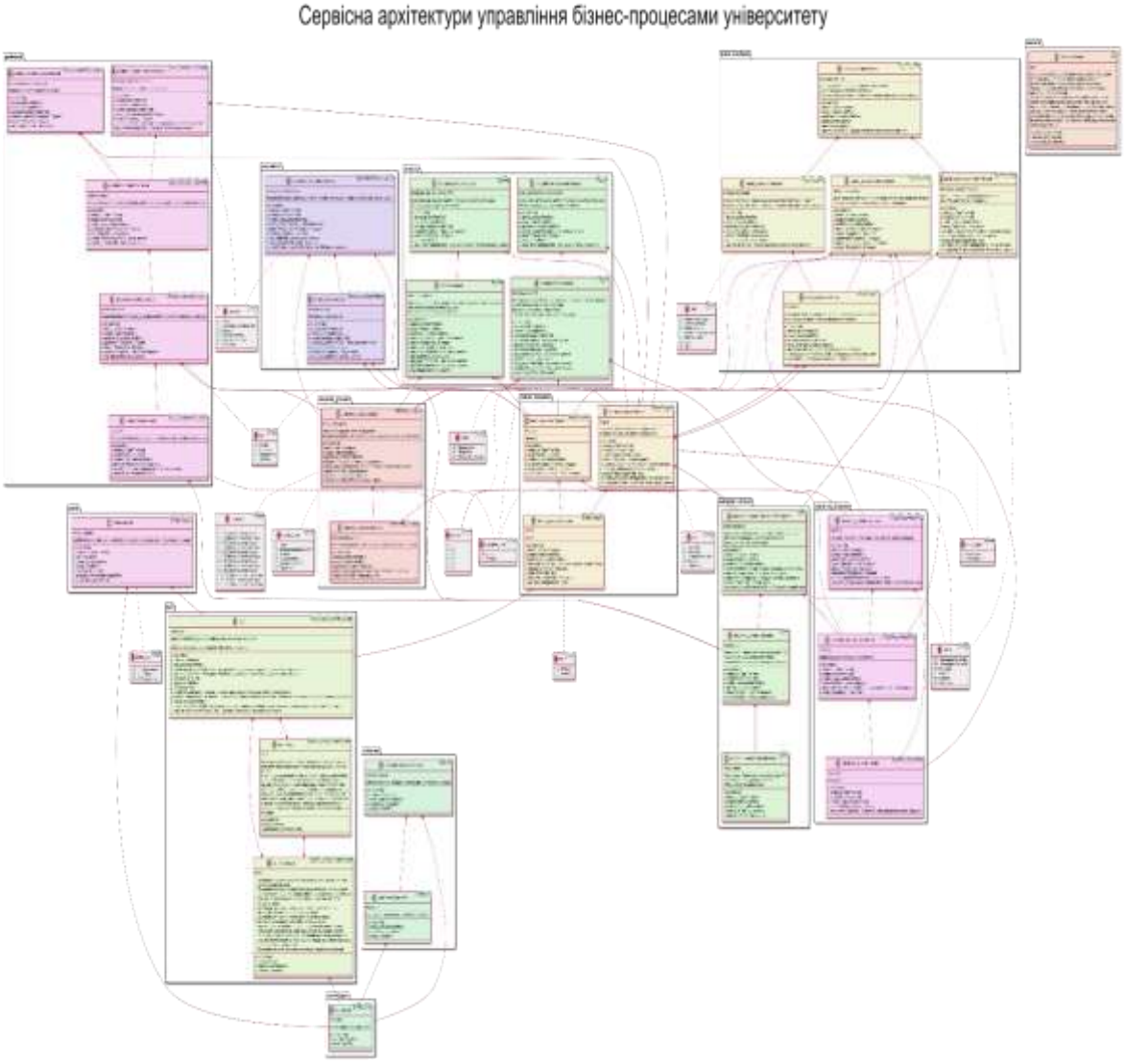

Рисунок 2.1 - Архітектура системи управління бізнес-процесами університа

Далі нами будуть розглянуті основні класи які були спроєктованні

для нашої системи.

Один із перших класів що ми розробили. Це був клас для відображення відділу кадрів університета.

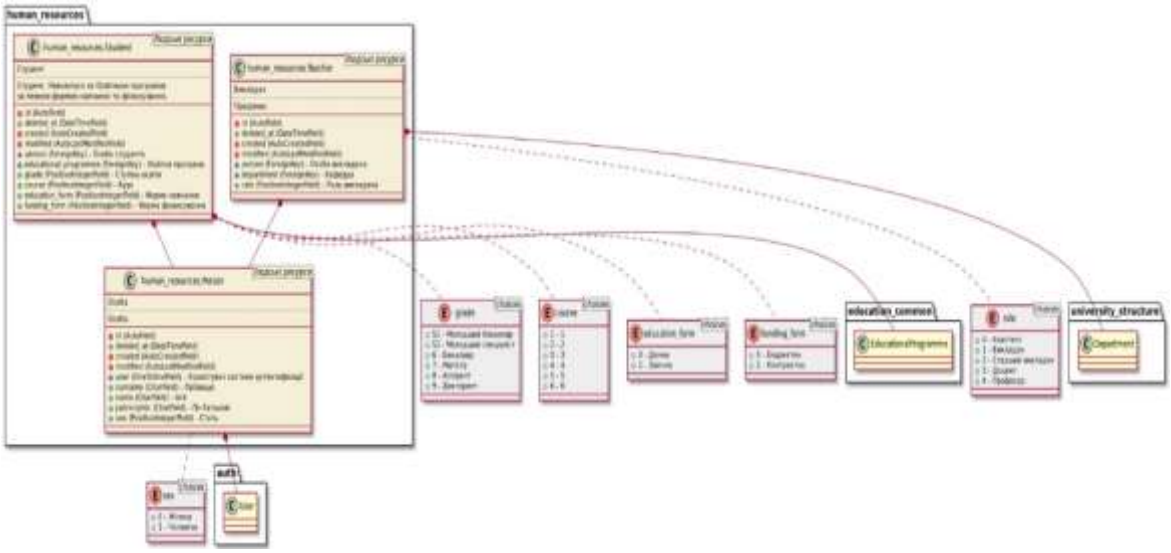

Рисунок 2.2 - Відділ кадрів

В класі людські ресурси для нашої системі, головними стали студенти та викладачі, як зображено на Рисунок 2.2.

Кожен із викладачів відноситься до своєї кафедри та має свою роль. Студент в свою чергу має відношення до своєї освітньої програми, курсу, форми навчання та форми фінансування.

Далі ми спроєктували клас для навчального процесу в університету, цей клас зображений на рисунок 2.3

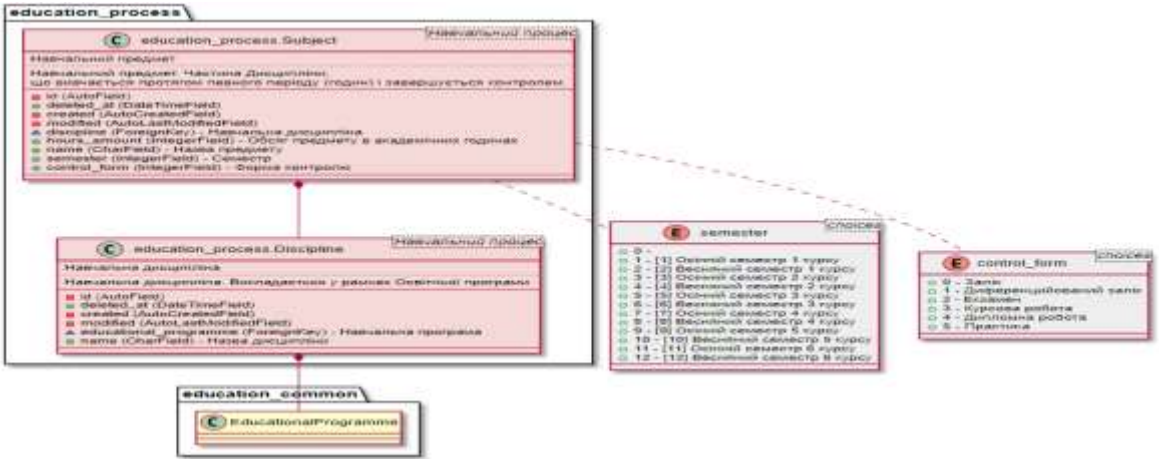

Рисунок 2.3 - Навчальний процес

Після цього ми почали роботу над розробкою класу освіта в якому ми описуємо:

1. Галузь знань.
2. Спеціальність
3. Освітню програму.

Галузь знань та спеціальність, являє собою перелік визначених постановою КМУ, галузей знань та спеціальностей, за якими буде здійснюватися підготовка здобувачів вищої освіти.

Освітня програма має відношення до конкретної спеціальності та пов'язана з кафедрою, що здійснює підготовку.

Даний клас зображений на рисунок 2.4

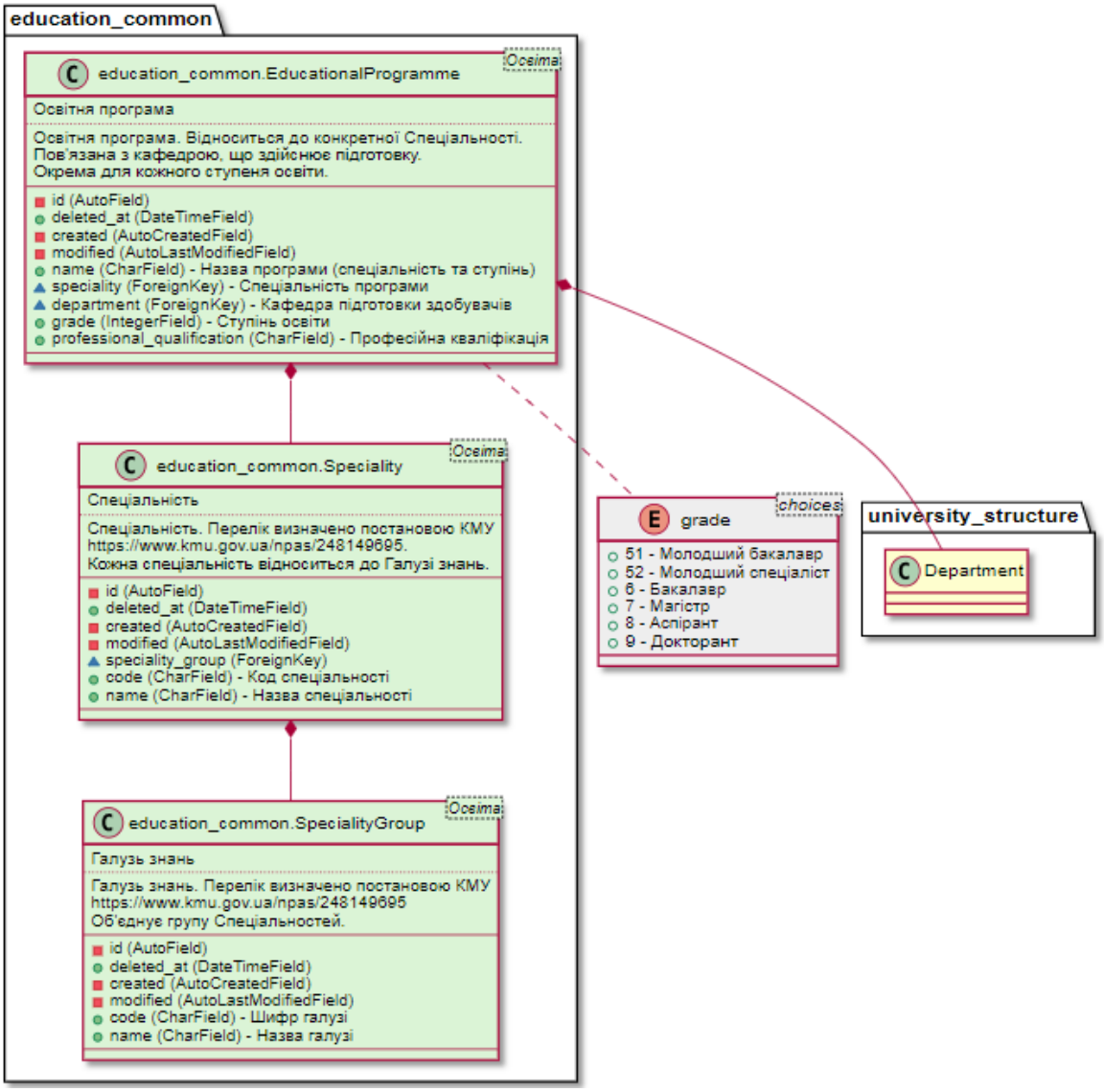

Рисунок 2.4 - Діаграма класу освіта

Після цього ми розробили клас для відділу контролю якості освіти. Головним інструментом для забезпечення контролю якості освіти є

опитування, з яких потім можно було отримати статистичні дані, як зображено на рисунок. 2.5

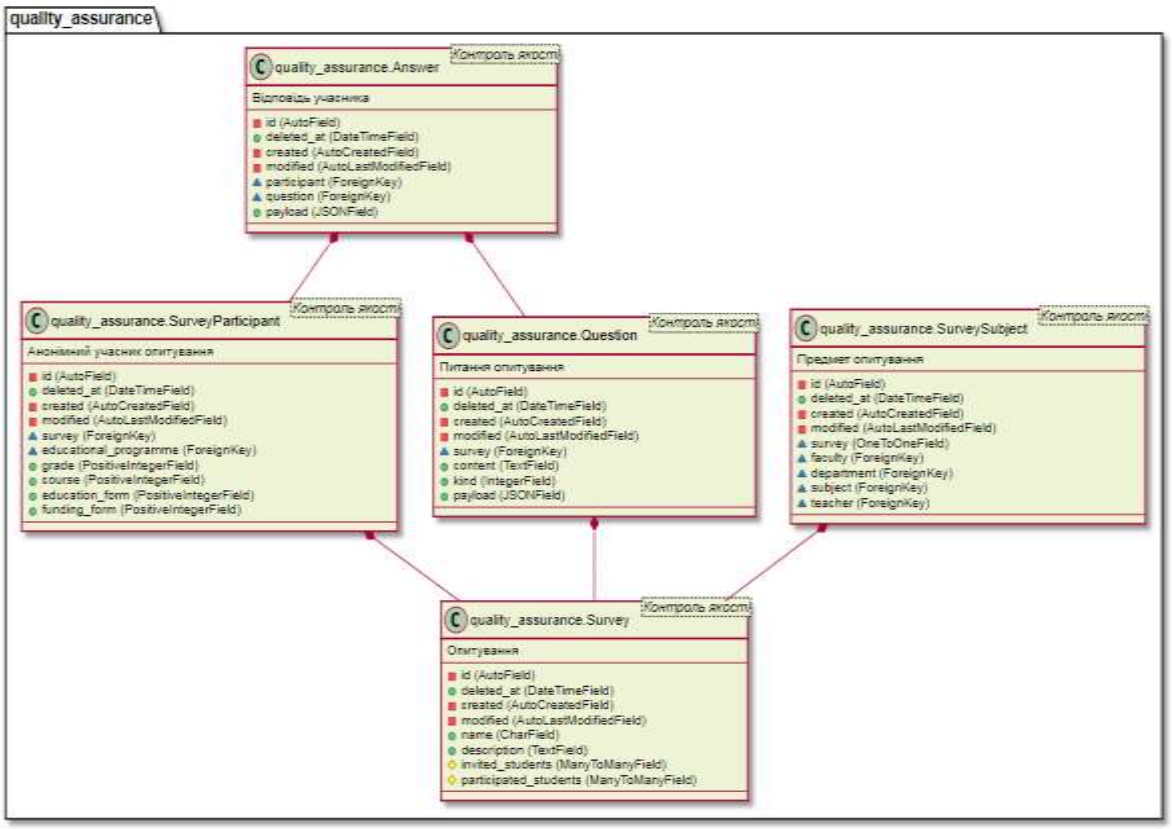

Рисунок 2.5 - Контроль якості освіти

2.2 Проектування моделі даних сервісу "Деканат".

Деканат - це підрозділ управління факультетом, який керується деканом, якого призначає ректор університету, струтура деканату зображена на рисунок 2.6

Основні завдання які виконує деканата:

1. Організації та здійснення навчального процесу;
2. Заповнення і зберігання облікової документації;
3. Контроль за успішністю студентів (розробка діаграм успішності та рейтингу студентів);
4. Участь у розробці та затвердженні навчальних та робочих навчальних планів;
5. Видача документів студентам;
6. Заповнення і зберігання особистої інформації студентів в

особові справи та особові картки;

7. Заповнення і зберігання особистої інформації працівників в особові справи;

8. Ведення архіву документації деканату, на який опираються інші структури університету (бухгалтерія, навчальний відділ, та інші.);

9. Підготовка звітів;

10. Створення розкладу навчальних занять;

11. Участь у розробці та затвердженні навчальних та робочих навчальних планів;

Рисунок 2.6 - Структура деканату

Для коректної роботи сервісу “Деканат” нам потрібно розробити наступні мікросервіси які будуть мати власний функціонал. Для кожного із мікросервісів нами була розроблені діаграми класів (class diagram):

1. Мікросервіс “Розклад навчальних занять”.

Розклад занять – регламентує трудовий ритм, впливає на творчу віддачу викладачів, тому його можна вважати фактором оптимізації

використання обмежених ресурсів – викладацького складу і аудиторного фонду, як зображено на рисунок 2.7

Проблему складання розкладу слід розглядати не тільки як трудомісткий процес, об’єкт автоматизації з використанням комп’ютера, але і як проблему оптимального керування.

Оскільки всі фактори, що впливають на розклад, практично неможливо врахувати, а інтереси учасників навчального процесу різноманітні, задача складання розкладу є багатокритеріальною з нечіткою множиною факторів.

Незалежно від алгоритму побудови розкладу, виникає прикладна проблема з інструментів різних рівнів, що використовуються в процесі.

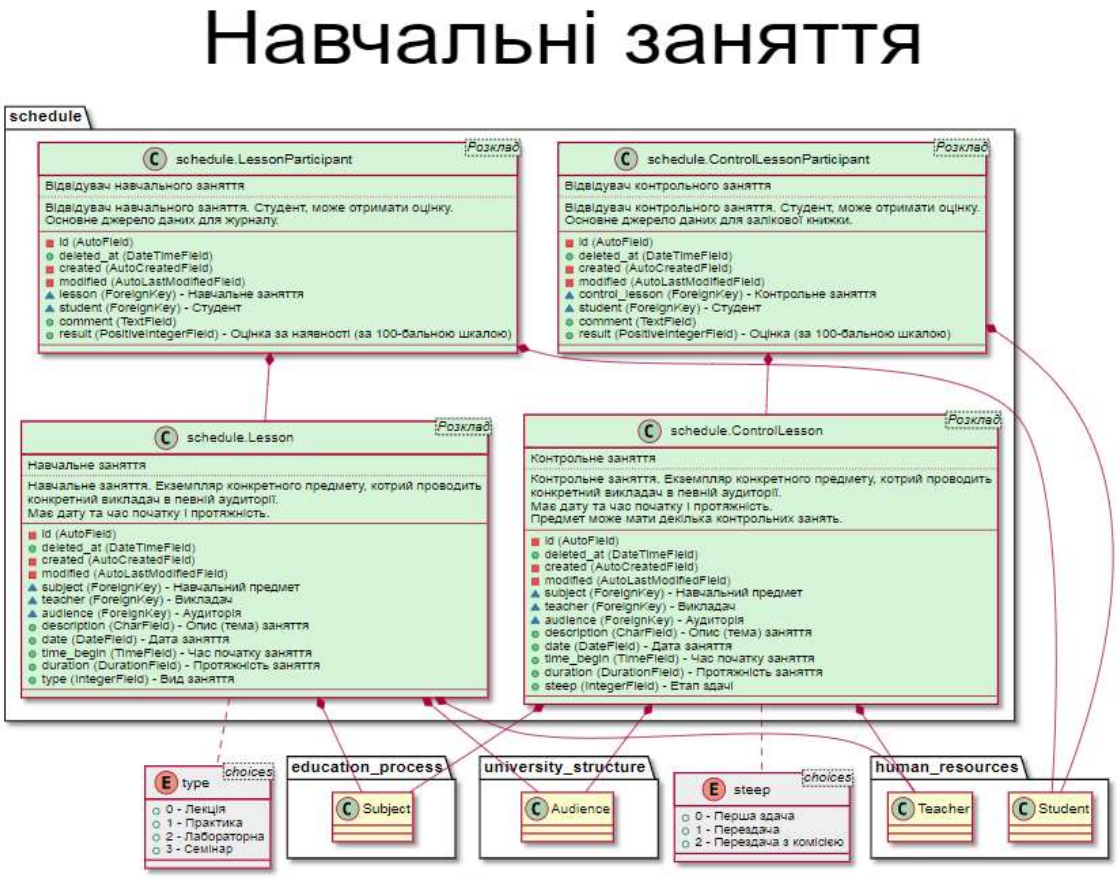

Рисунок 2.7 - Діаграма класів мікросервісу “Розклад навчальних занять”.

2. Мікросервіс “Електронний журнал”.

Електронний журнал - це електронний документ про реєстрацію

відвідуваності та успішності студентів. Він призначений для спрощення контролю та обліку навчальних досягнень і відвідування студентів навчальних занять. Кожен викладач заповнює свій листок так як і в звичайному шкільному журналі, як зображено на рисунок 2.8

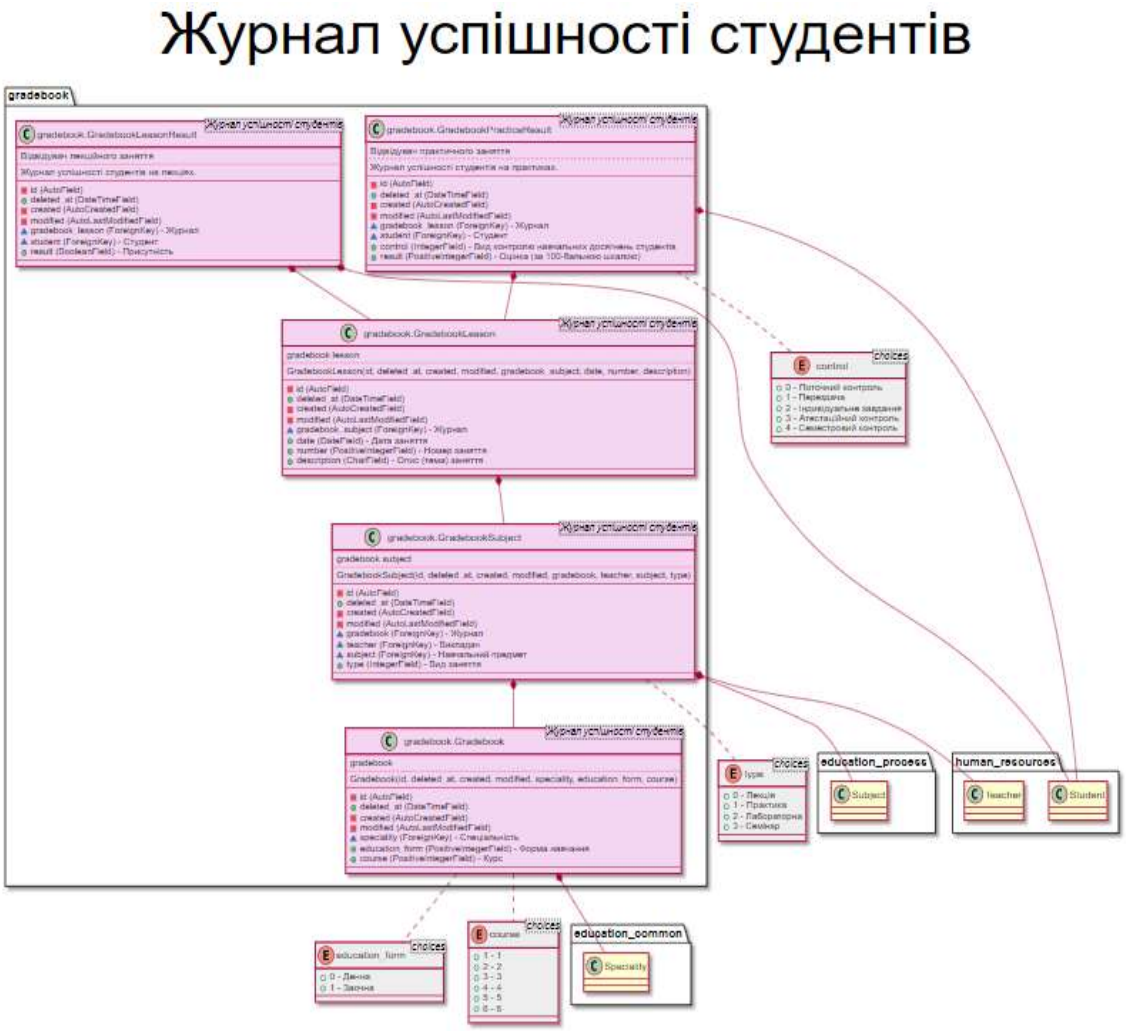

Рисунок 2.8 - Діаграма класів мікросервісу “Електроний журнал”.

3. Мікросервіс “Електронна залікова книжка студента”.

Електронна залікова книжка - це електронний документ, у якому містяться записи про здачу студентом заліків, іспитів, захисту курсових і дипломних робіт, виробничої та педагогічної практики (Рисунок 2.9).

4. Мікросервіс “Електронна відомість”.

Документ до якого вносяться результати заліків, іспитів, курсових і дипломних робіт.

Рисунок 2.9 - Діаграма класів мікросервісу “Електронна залікова книжка студента”.

5. Мікросервіс “Індивідуальний навчальний план успішності студента”.

Індивідуальний навчальний план успішності студента є робочим документом студента, що містить інформацію про перелік і послідовність вивчення навчальних дисциплін, обсяги навчального навантаження студентів із усіх видів навчальної діяльності та відповідні форми контролю.

ІНПС формується на основі навчального плану підготовки фахівців за відповідним освітньо кваліфікаційним рівнем за роками (семестрами). В індивідуальному навчальному плані студента зазначаються перелік нормативних навчальних дисциплін, навчальних дисциплін за вибором, усі види практик у межах нормативно встановлених термінів підготовки фахівців певного освітньо кваліфікаційного рівня та навчальні дисципліни, що вивчаються додатково.

2.3 Проектування API

Для взаємодії між нашими мікросервісами, нам потрібно розробити API. Ми будемо розробляти API для кожного із наших

мікросервісів, а саме:

1. Профіль користувача
2. Мікросервіс “Розклад”
3. Мікросервіс “Електроний журнал”
4. Мікросервіс “Залікова книжка”
5. Мікросервіс “Розклад”
6. Мікросервіс “Забезпечення якості освіти”

Для створення REST ми будемо використовувати Swagger.

Swagger це специфікація фалов з інтерфейсами, для створення, візуалізації, опису nf використання REST веб сервісів.

При створені API, ми будемо отримувати в адмін панелі JSON код який потім будемо віддавати до front-end та виводити на сторінки нашого сервісу.

Для корректного вигляду коду модели, ми будемо використовувати Serializers, як це зображено на рисунок 2.10

```
from rest_framework import serializers

from apps.human_resources.models import Person
from apps.human_resources.serializers import StudentSerializer, TeacherSerializer

__all__ = [
    'UserProfileSerializer',
]


class UserProfileSerializer(serializers.ModelSerializer):
    sex = serializers.CharField(source='get_sex_display')
    email = serializers.CharField(source='user.username')
    is_superuser = serializers.BooleanField(read_only=True, source='user.is_superuser')
    is_staff = serializers.BooleanField(read_only=True, source='user.is_staff')
    is_active = serializers.BooleanField(read_only=True, source='user.is_active')
    date_joined = serializers.DateTimeField(read_only=True, source='user.date_joined')
    as_student = StudentSerializer(source='student_set', many=True)
    as_teacher = TeacherSerializer(source='teacher_set', many=True)

    class Meta:
        model = Person
        fields = [
            'id',
            'surname',
            'name',
            'patronymic',
            'sex',
            'email',
            'date_joined',
            'is_superuser',
            'is_staff',
            'is_active',
            'date_joined',
            'as_student',
            'as_teacher',
        ]
```

Рисунок 2.10 - Приклад опису фрагменту Serializers для виведення інформації о користувачі

Після розробки API для особистого профіля, ми почали розробляти API для мікросервісу "Розклад", ми також використали serealizers для того щоб привести JSON код до корректного вигляду, та

відправляти потом JSON на головну сторінку профіля, це зображено на рисунок 2.11.

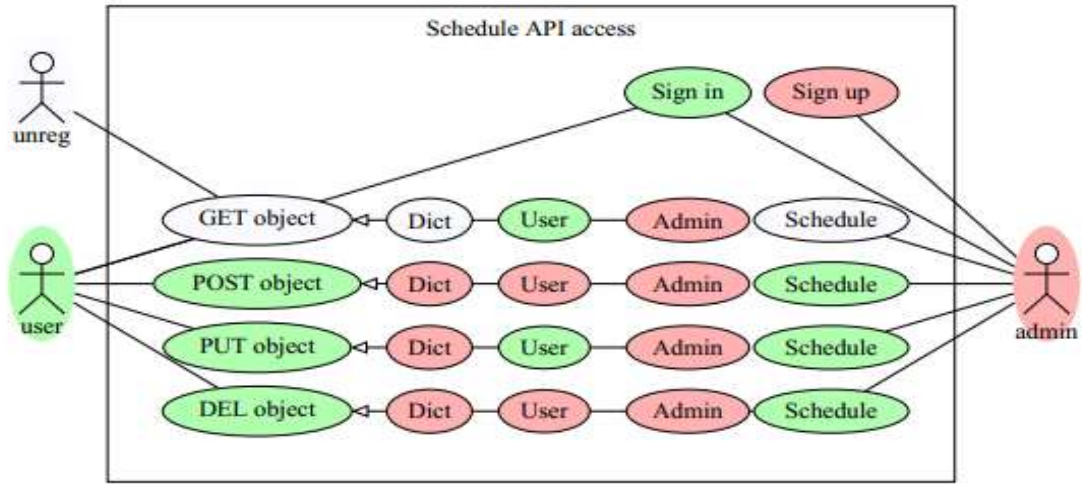

Рисунок 2.11 - Доступ на виконання запитів до системи

2.4 Проектування UI\UX

UX / UI дизайн - це проектування будь-яких призначених для користувача інтерфейсів в яких зручність використання так само важливо як і зовнішній вигляд [20, с. 34].

UX (user experience, досвід взаємодії) - це поведінка людини, його ставлення, і емоції викликані і пов'язані з використанням сайту [18, с. 25].

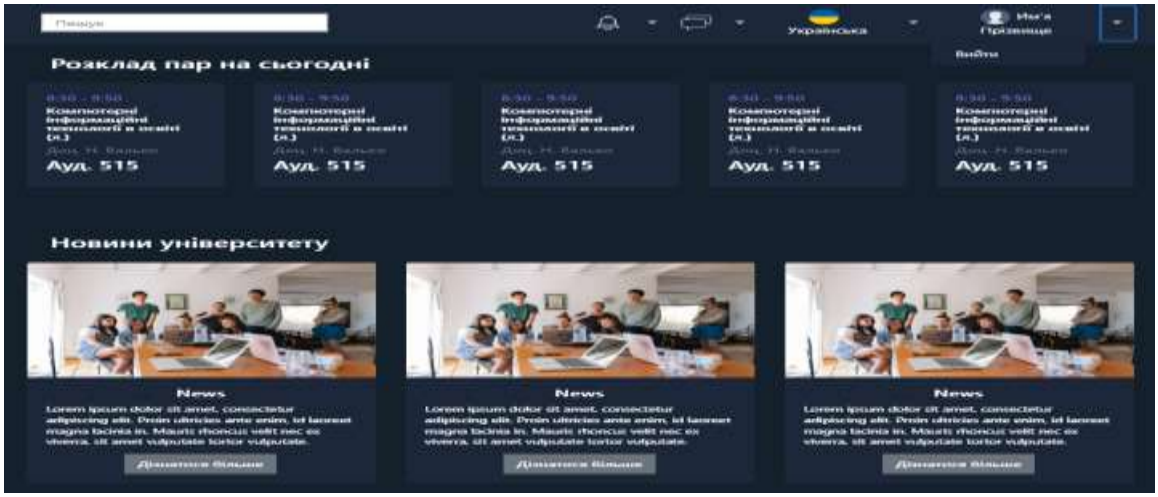

Рисунок 2.12 - Головна сторінка сервісу.

UI (user interface, призначений для користувача інтерфейс) - це сукупність засобів і методів, за допомогою яких користувач взаємодіє з

сайтом, зовнішній вигляд сайту [19, с. 18].

Елементи сайту. Можна сформулювати набір обов'язкових елементів:

1. Профіль користувача
2. Вхід і реєстрація
3. Розклад навчальних занять
4. Розклад контрольних занять
5. Електроний журнал
6. Електрона залікова книжка студента
7. Електронна відомість
8. Анкетування
9. Новинна стрічка

Головна сторінка сервісу матиме вигляд, як на рисунок 2.12.

Сторінка інформації про власника матиме вигляд, як зображено на рисунок 2.13.

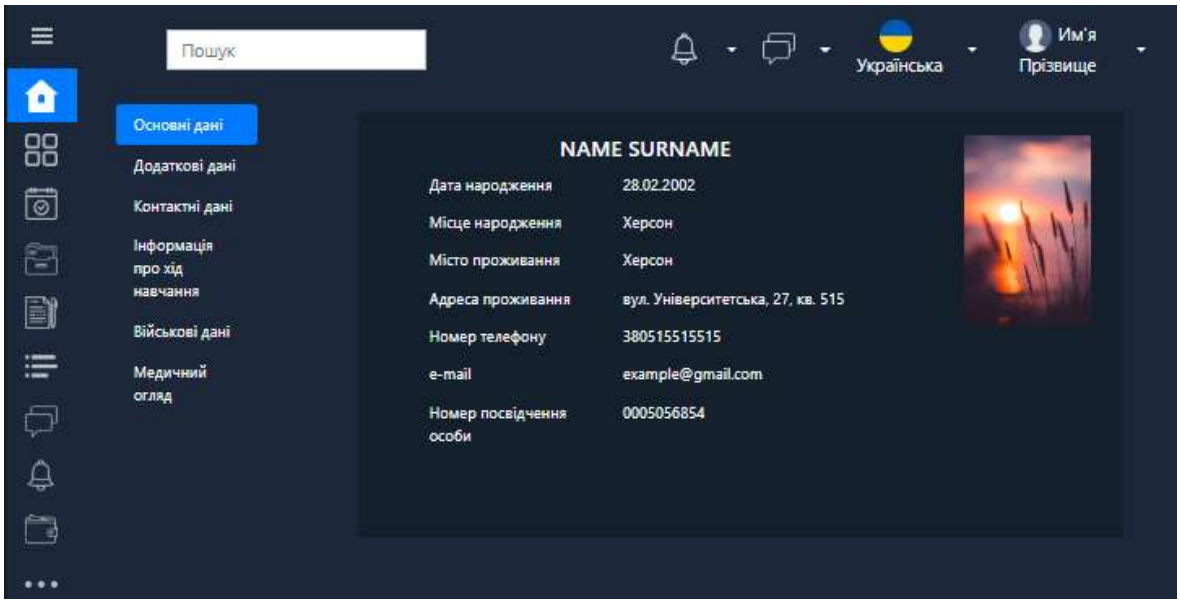

Рисунок 2.13 - Сторінка інформації про студента

Для розробки дизайну сайту нами була обрана найновіший стандарт для мови CSS, а саме CSS3.

CSS3 має розширені можливості ніж CSS2, новими можливостями які є в CSS3, це підтримка веб-шрифтів, нові кольори та нові ефекти для

зображень.

Також при розробці дизайну нашого сервісу особливу увагу ми приділилу вибору кольорів. Кольори мають велике значення при створенні дизайну сайту, деякі кольорм сожуть негативно впливати на емоційний стан людини. А така реакція може негативно вплинути на імідж університету.

Тому основним кольором для нашого сервісу був обраний синій. Синій кольор має заспокійливий ефект, тому він добре підійде як головний кольор.

Для розробки UI ми використали бібліотеку React. Головне завдання реакту надати розробнику високу швидкість при розробці, легкість у застосуванні та легкість у масштабуванні.

Сторінка вибору користувача матиме наступний вигляд, як зображено на рисунок 2.14.

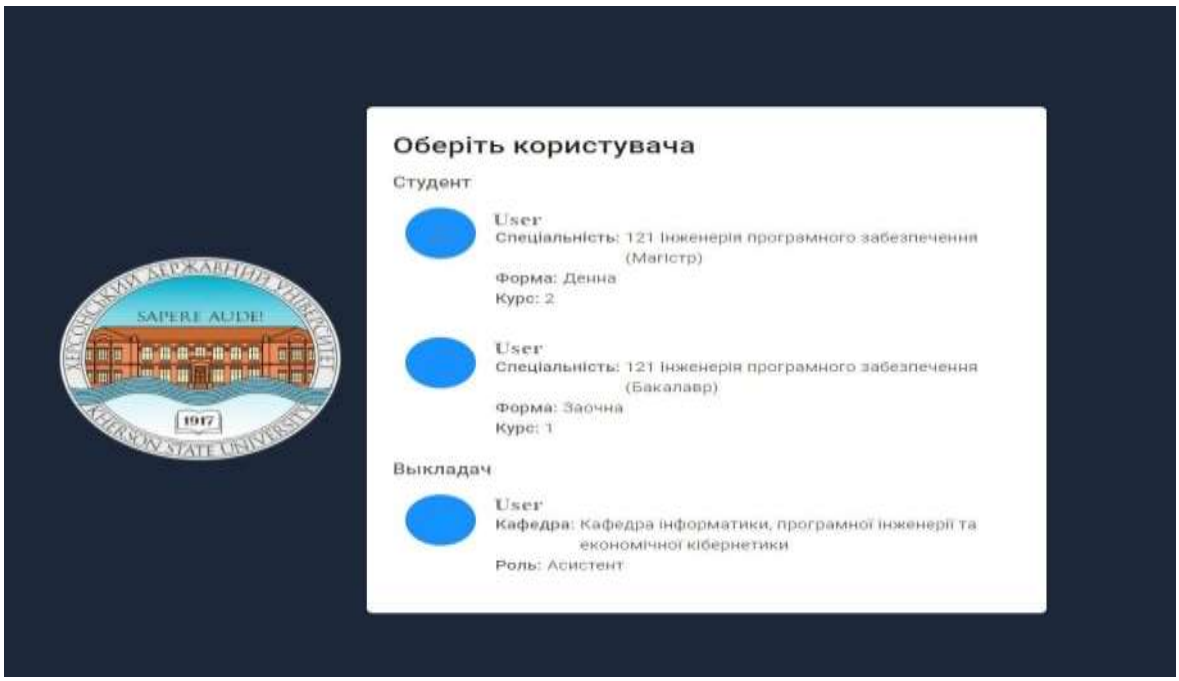

Рисунок 2.14 - Сторінка вибору ролі користувача

РОЗДІЛ 3

Розробка сервісу "Деканат"

3.1 Структура сервісу "Деканат".

Після того як у розділі 2, ми спроєктували архітектуру та описали структуру класів для сервісу “Деканат”. Наступним кроком буде почати розробляти структуру сервісу “Деканат”.

Він буде виглядати наступним чином:

1. Мікросервіс “Розклад навчальних занять”.

1.1.1. Формування розкладу занять.

1.1.2. Формування розкладу контрольних занять (іспитів, диференційованих заліків, заліків).

1.1.3. Можливість додавання занять у ZOOM.

1.1.4. Редагування розкладу навчальних занять (аудиторій, викладачів).

Головним завданням для мікросервісу “Розклад навчальних занять”, це легкість у формуванні розкладу, можливість швидко внести зміни до нього за необхідностью. Коректний вивід на головну сторінку сервісу.

2. Мікросервіс “Електронний журнал”.

2.1. Створення нового журналу для навчальної групи.

2.2. Додавання навчальних дисциплін, які передбачені у навчальній програмі.

2.3. Додавання оцінок, які отримують студенти за контрольне заняття.

2.4. Можливість додавання присутності або відсутності студента на лекційних заняттях.

Мікросервіс “Електронний журнал” повинен відображувати успішність студентів під час навчального процесу.

3. Мікросервіс “Електронна залікова книжка”.

3.1. Створення залыковоъ книжки для студента.

3.2. Додавання навчальних дисциплін.

3.3. Додавання викладачем оцінок за вивчений курс.

Мікросервіс “Електронна залікова книжка”, буде зберігати записи про те що студент склав іспити, залік, курсові роботи.

4. Мікросервіс “Електронна відомість”.

4.1.1. Форма контролю.

Мікросервіс “Електронна відомість”, буде виконувати схожі функції як і мікросервіс “Електронна залікова книжка”, але лише для однієї навчальної дисципліни.

5. Мікросервіс “Індивідуальний навчальний план успішності студента”.

5.1.1. Статистичні результати успішності студента.

3.2 Розробка компонентів сервісу.

При розробці компонентів до сервісу управління бізнес процесами університету, ми використовували інтегроване середовище розробки PyCharm для написання вихідного коду.

Один із перших компонентів який нами було розроблено це мікросервіс “Розклад навчальних занять”, як показано на рисунок 3.1.

По-перше ми реалізували модель данних розкладу в нашому проекті. Ми довели її до вигляду діаграми класів мікросервісу “Розклад навчальних занять”. (Рисунок 2.3.)

Рисунок 3.1 - Приклад опису фрагменту моделі даних навчального придмету

Далі ми розробили “Serializers” для того щоб перетворювати складні дані, такі як набори запитів і екземпляри моделей, у власні типи даних Python, щоб потім можно було їх легко перетворити на JSON, як показано на рисунок 3.2.

Серіалізатор також забезпечують десеріалізацію, дозволяючи перетворювати проаналізовані дані назад в складні типи після першої перевірки вхідних даних.

```python
from rest_framework import serializers

__all__ = [
    'LessonSerializer',
    'ControlLessonSerializer',
    'LessonParticipantSerializer',
    'ControlLessonParticipantSerializer',
]

from .models import Lesson, ControlLesson, LessonParticipant, ControlLessonParticipant


class LessonSerializer(serializers.ModelSerializer):
    class Meta:
        model = Lesson
        fields = [
            'id',
            'subject',
            'teacher',
            'audience',
            'description',
            'date',
            'time_begin',
            'duration',
        ]


class ControlLessonSerializer(serializers.ModelSerializer):
    class Meta:
        model = ControlLesson
        fields = [
            'id',
            'subject',
            'teacher',
            'audience',
            'description',
            'date',
            'time_begin',
            'duration',
            'steep',
        ]
```

Рисунок 3.2 - Приклад опису фрагменту serializers для розкладу начальних занять.

Після цього ми розробили API, як показано на рисунок 3.3.

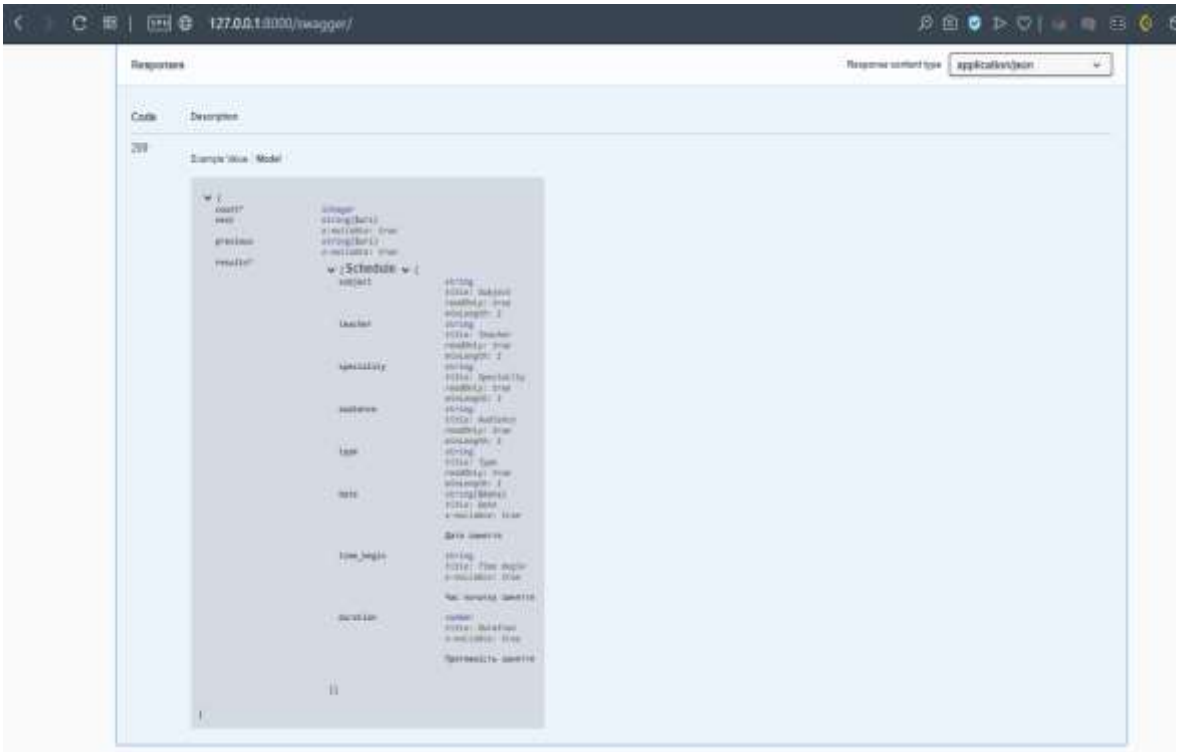

Рисунок 3.3 - API для розкладу.

Після завершиння налаштування методів, ми зможемо побачити реалізований нами розклад навчальних занять на головній сторінці нашого сервісу, як показано на рисунок 3.4.

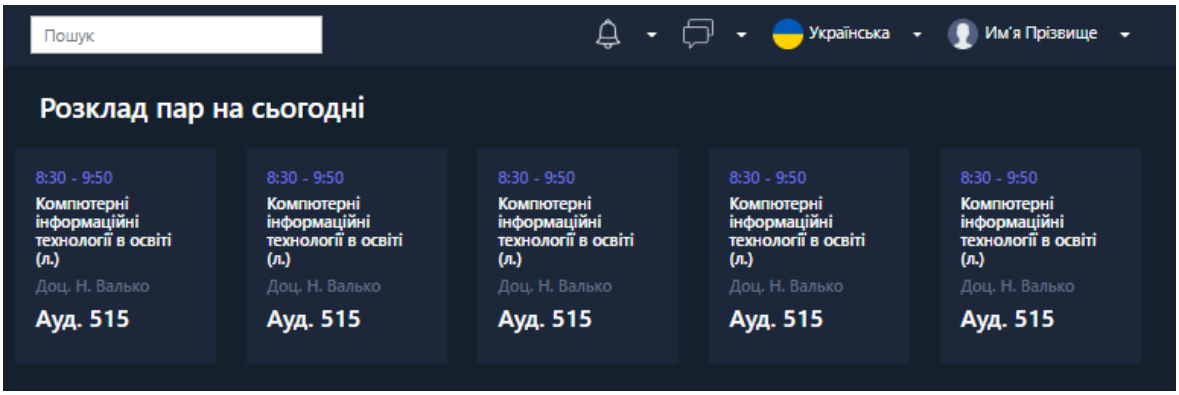

Рисунок 3.4 - Відображення розкладу навчальних занять.

Далі ми почали розробку мікросервісу “Електроний журнал”.

Знову ж таки, ми почали з розробки моделі данних для данного мікросервісу, як показано на Рисунок 3.3. Ми довели данну модель до вигляду діаграми класів мікросервісу “Електроний журнал”. (Рисунок 2.4.)

```python
class Gradebook(DataObject):
    speciality = models.ForeignKey(Speciality, on_delete=models.PROTECT, help_text='Спеціальність ')
    education_form = models.PositiveIntegerField(choices=EDUCATION_FORM_CHOICES, help_text="Форма навчання")
    course = models.PositiveIntegerField(choices=COURSE_CHOICES, help_text="Курс")


class GradebookSubject(DataObject):
    gradebook = models.ForeignKey(Gradebook, on_delete=models.PROTECT, help_text='Журнал')
    teacher = models.ForeignKey(Teacher, on_delete=models.PROTECT, help_text='Викладач')
    subject = models.ForeignKey(Subject, on_delete=models.PROTECT, help_text='Навчальний предмет')
    type = models.IntegerField(choices=LESSON_TYPE_CHOICES, default=0, help_text="Вид заняття")


class GradebookLesson(DataObject):
    gradebook_subject = models.ForeignKey(GradebookSubject, on_delete=models.PROTECT, help_text='Журнал')
    date = models.DateField(blank=True, help_text="Дата заняття", null=True)
    number = models.PositiveIntegerField(default=1, help_text="Номер заняття")
    description = models.CharField(max_length=255, blank=True, help_text='Опис (тема) заняття', null=True)


class GradebookResult(DataObject):
    gradebook_lesson = models.ForeignKey(GradebookLesson, on_delete=models.PROTECT, help_text='Журнал')
    student = models.ForeignKey('human_resources.Student', on_delete=models.PROTECT, help_text='Студент')

    class Meta:
        abstract = True
```

Рисунок 3.5 - Приклад опису фрагменту моделі даних електроного журналу

3.3 Розробка адміністративної частини сервісу

Після створення основної частини web-додатку, нами була розроблена адміністративна частина. За допомогою адміністративної частини у нас з'явиться можливість більш легко керувати сайтом: додавати нові матеріали, керувати користувачами, оновлення та видалення записів. Перед початком роботи необхідно пройти авторизацію, для забезпечення захисту системі, як показано на рисунок 3.6, містить два поля введення і кнопку входу до системи.

Рисунок 3.6 - Авторизація

Після входу ми опиняємося на головній сторінці адмін-панелі. На головній сторінці відображаються всі моделі що містяться у нашому сервісі, вони згруповані за встановленим додатком. На цій сторінці ми можемо вносити зміни в будь яку існуючу модель. У головному меню користувач вибирає адміністративний розділ для роботи та здійснює дії з обраним розділом (Рисунок 3.7).

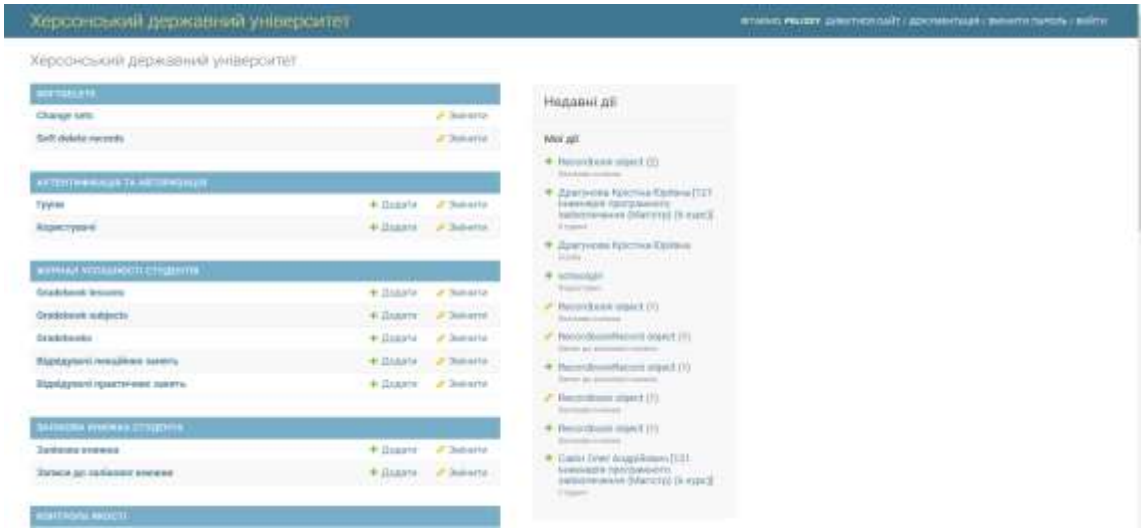

Рисунок 3.7 - Загальний вигляд адміністративної системи

Скориставшись кнопкою “Додати”, ми можемо додати новий запис до розкладу навчальних занять або іншої моделі. (Рисунок 3.8).

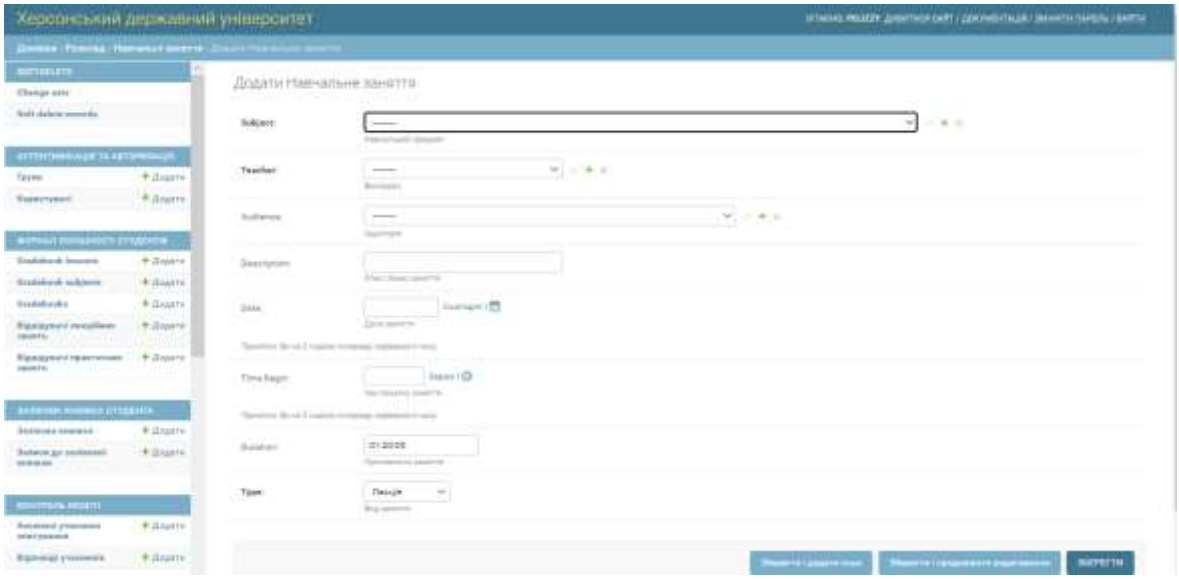

Рисунок 3.8 - Додавання навчальних занять

Наразі в моделі “Розклад навчальних занять” ми можемо додавати навчальні заняття та додавати контрольні заняття.

Також ми можемо вносити зміни до моделі “Електронний журнал”, “Залікова книжка” та “Опитування”

В моделі "Електронний журнал", ми можемо створювати сам журнал та виставляти оцінки на лабараторних заняттях або присутність на лекційних.

В моделі “Залікова книжка”, ми можемо створювати власну залікову книжку студента або додавати запис вже до існуючої.

Та у моделі “Опитування”, ми можемо створювати опитування або анонімне опитування та відслідковувати результати опитувань.

Тому адміністративна частина ділиться на різні розділи які відповідають потрібному призначенню.

Також може відрізнятися оформлення робочого простору. Це залежить від того над якою моделью ви працюєте. Проте існують загальні стандарти вигляду адміністративного інтерфейсу.

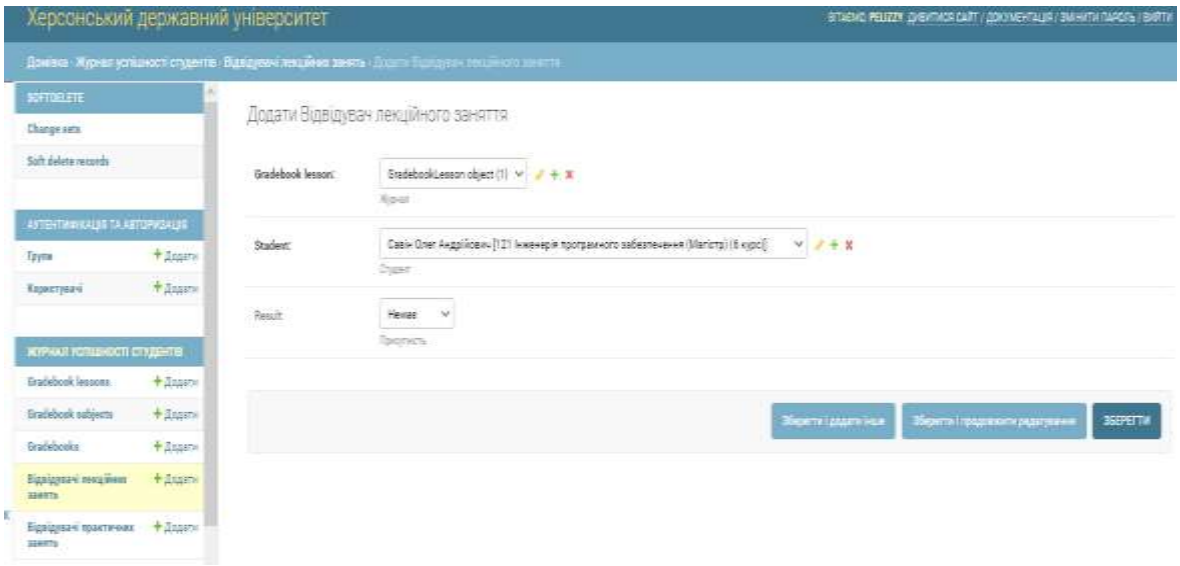

Рисунок 3.9 - Додавання результатів відвідувача лекційного заняття

3.4 Тестування програмного продукту

Під час тестування, які проводились згідно з програмою та методикою тестування сервісної архітектури “Управління бізнес-процесами університету. Сервіс "Деканат”.

Цілью проведених випробувань було індифікування неточностей і помилок, допущених при створенні коду і розробці ПЗ, а також надання виготовленому програмному забезпеченню закінченого та готового вигляду, придатного до застосування.

Основними вимогами до програмного продукту під час випробування стали:

1) Швидкий у освоєнні та використанні інтерфейс;
2) Простий у розумінні функціонал;
3) Швидкість роботи програмного продукту.

Випробування проводились при наявності компьютера з підключенням інтернету згідно розробленої завчасно методики випробувань.

Отримані такі результати:

1. Функція переходу між сторінками та відкриваня вспливаючього вікна користувацького інтерфейсу програми. Випробовувалась можливість переходу до голоної сторінки, знаходячись на будь якій іншій сторінці сервісу. Програма не зазнала збоїв у роботі і користувачеві було зручно реалізовувати дії.

2. Функція реєстрації нового користувача. Випробовувалась функція реєстрації нового користувача та функція перемикання між ролями. Функція не зазнала збоїв у роботі. На Рисунок 3.10 зображен процес перемикання між існуючими ролями користувача.

3. Функціонал для додавання навчальних занять. Випробовувалося можливість додавання декількох навчальних занять із ризними даними. Функціонал не зазнавав збоїв у роботі.

4. Функціонал перегляду електроного журналу успішності. Функціонал додавання оцінок до електроного журналу. Функціонал не зазнавав збоїв у роботі. Процес переглядуелектроного журналу.

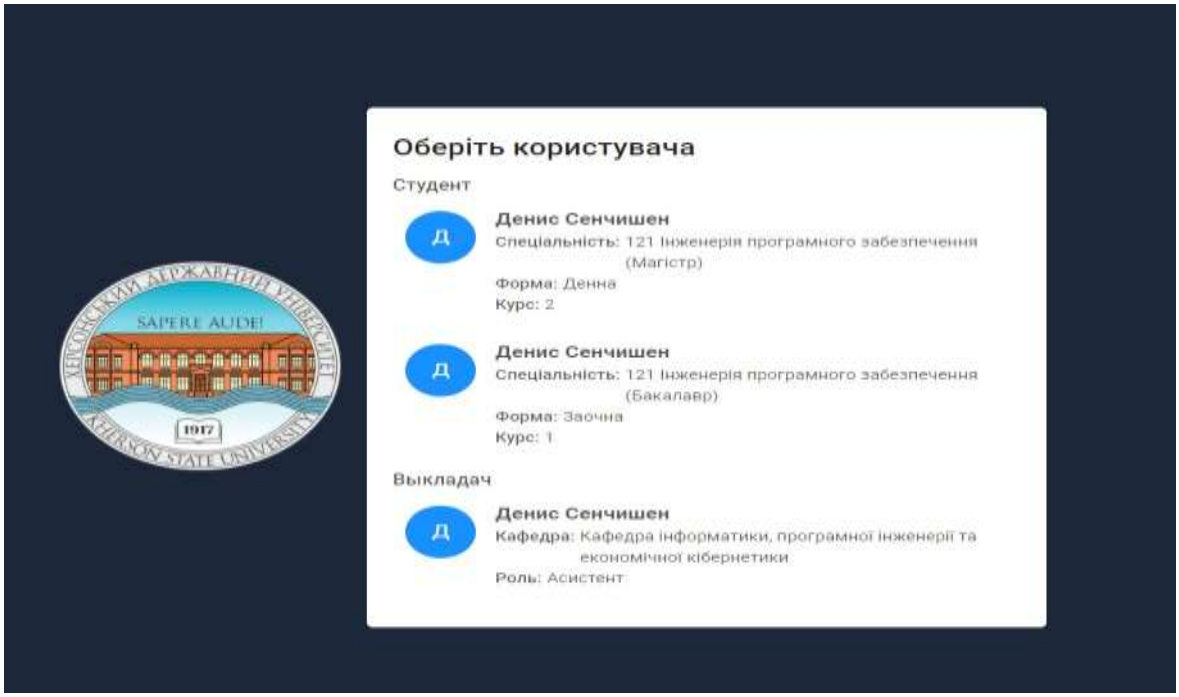

Рисунок 3.10 - Вибір користувача

5. Функціонал додавання оцінок до залікової книжки студента. Функція не зазнала збоїв у роботі.

ВИСНОВКИ

Основою успішного університету є доступ студентів до інформації в електронному варіанті. Крім того, широке застосування інформаційних технологій у роботі університета.

Для виконання поставлених завдань було проведено аналіз характеристик існуючих систем, зокрема обсяг їх можливостей.

На основі проведеного аналізу розроблено базові вимоги щодо можливостей сервісу “Деканат”. Детально досліджено роботу клієнт--серверних додатків та супутніх технологій. Розглянуто та обгрунтовано використання сервісного підходу при проектуванні системи.

Для досягнення мети даної роботи був проведений глибокий аналіз предметної області, виявлені проблеми усунуті, для чого були складені діаграми процесів, варіантів використання, а також схема бази даних.

Метою даної роботи була розробка сервісу “Деканат” для системи управління бізнес-процесами університету.

Розроблено публічний прикладний програмний інтерфейс (API), та сформовано документацію до нього.

При виконанні завдання мною були реалізовані наступні завдання:

1. Розроблений мікросервіс “Розклад навчальних занять”.
2. Розроблений мікросервіс “Електронний журнал”.
3. Розроблений мікросервіс “Електронна залікова книжка”.
4. Розроблений мікросервіс “Електронна відомість”.
5. Розроблений мікросервіс “Індивідуальний навчальний план успішності студента”.
6. Розроблено API для мікросервісів.

При розробці проекту використовується система контролю версій git з публічним репозиторієм на сервісі GitLab, що дозволяє використовувати сучасні методи сумісної роботи та дозволяє

використовувати результати проведеного дослідження всім охочим під ліцензією MIT.

Розроблена система може використовуватись по прямому призначенню.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Расс Унгер. Кэролайн Чендлер. UX-дизайн. Практичний посібник з проектування досвіду взаємодії. 2011. с. 26-115.
2. Бирман Ілья. Інтерфейс користувача. 2016. с. 15-46.
3. Итана Маркотта. Чуйний веб-дизайн. 2012. с. 31-39.
4. Ден Сідерхолм. CSS3 для веб-дизайнерів. 2017. с. 50-89.
5. Джеймі Леві. UX-стратегія. Чого хочуть користувачі і як їм це дати. 2017. с. 25-96.
6. Дронов В.А. Django 2.1. Практика створення веб-сайтів на Python. 2019. с. 89-115.
7. Малярчук С. М. Основи інформатики у визначеннях, таблицях і схемах: Довідково-навчальний посібник. 2017. с. 112.
8. Едріен Моует. Використання Docker. Розробка і впровадження програмного забезпечення за допомогою технології контейнерів. 2017. с. 18-356.
9. Дронов В. А. Django: практика створення Web-сайтів на Python. 2016. с. 29-36.
10. Парміндер С.К. Мікросервіси і контейнери Docker. 2016. с. 34-215.
11. Джефф Форсье. Django. Розробка веб-додатків на Python. 2009. с. 64.
12. Прохоренок Н.А. Python 3 і PyQt. Розробка додатків. 2018. с. 26.
13. Персіваль Г. Python. Розробка на основі тестування. 2018. с. 87.
14. Алчин М. Pro Django, 2-е видання. 2013. с. 168.
15. Тарек Зиаде. Розробка Python Microservices. 2017. с. 201.
16. Єдинадержавнаелектроннабазазпитаньосвіти.—2020.— URL:https://mon.gov.ua/ua/ministerstvo/yedebo
17. Кравцов Д.Web-портал «Херсонський Віртуальний Універси-тет». — 2010. — URL:http://dls.kherson.ua/.
18. Львов М.Співаковський О.Щедролосьєв Д.Інформаційна систе-ма

управління вищим навчальним закладом як платформа реалі-зації управління академічним процесом // Комп’ютер у школі тасім’ї. — 2007. — No 2. — С. 3—6.

19. Параничев А. Опыт проектирования учебного программного про-дукта с помощью инструментария PlantUML // Системы прое-ктирования, технологической подготовки производства и управ-ления этапами жизненного цикла промышленного продукта(CAD/CAM/PDM–2017). — 2017. — С. 224—227.
20. Ситник Н.В.Краснюк М.Т.Проектування баз і сховищ да-них // навчально-методичний посібник для самост. вивч. дисц. —2005. — С. 384.
21. Співаковський О.Вінник М.Тарасіч Ю.Побудова ІКТ інфра-структури ВНЗ: проблеми та шляхи вирішення // Інформаційнітехнологіїі засоби навчання.— 2014. — 39, вип. 1. — С. 99—116.
22. Співаковський О. В.Web-портал Херсонського державного уні-верситету. — 2013. — URL:http://kspu.edu/.
23. Hellmann D.The Python standard library by example. — Addison-Wesley Professional, 2011. — С. 1302.
24. Ed-Douibi H.Izquierdo J. L. C.Cabot J.OpenAPItoUML: a toolto generate UML models from OpenAPI definitions // InternationalConference on Web Engineering. — Springer. 2018. — С. 487—491.
25. Codd E. F.A relational model of data for large shared data banks //Communications of the ACM. — 1970. — Т. 13, No 6. — С. 377—387.
26. Carter J. L.Wegman M. N.Universal classes of hash functions //Journal of computer and system sciences. — 1979. — Т. 18, No 2. —С. 143—154.
27. Basu A.How to write using rich text format and markdown in latexand

overleaf // Universidad de Canterbury. — 2016.

28. Gatherer D.Less is more: the battle of Moore's law against Bremermann's limit on the field of systems biology // BMC systemsbiology. — 2007. — T. 1, S1. — P53.

ДОДАТКИ

Додаток А

КОДЕКС АКАДЕМІЧНОЇ ДОБРОЧЕСНОСТІ ЗДОБУВАЧА ВИЩОЇ ОСВІТИ ХЕРСОНСЬКОГО ДЕРЖАВНОГО УНІВЕРСИТЕТУ

Я, _____,
учасник(ця) освітнього процесу Херсонського державного університету, **УСВІДОМЛЮЮ**, що академічна доброчесність – це фундаментальна етична цінність усієї академічної спільноти світу.

ЗАЯВЛЯЮ, що у своїй освітній і науковій діяльності **ЗОБОВ'ЯЗУЮСЯ**:

– дотримуватися:

- вимог законодавства України та внутрішніх нормативних документів університету, зокрема Статуту Університету;
- принципів та правил академічної доброчесності;
- нульової толерантності до академічного плагіату;
- моральних норм та правил етичної поведінки;
- толерантного ставлення до інших;
- дотримуватися високого рівня культури спілкування;

– надавати згоду на:

- безпосередню перевірку курсових, кваліфікаційних робіт тощо на ознаки наявності академічного плагіату за допомогою спеціалізованих програмних продуктів;
- оброблення, збереження й розміщення кваліфікаційних робіт у відкритому доступі в інституційному репозитарії;
- використання робіт для перевірки на ознаки наявності академічного плагіату в інших роботах виключно з метою виявлення можливих ознак академічного плагіату;

– самостійно виконувати навчальні завдання, завдання поточного й підсумкового контролю результатів навчання;

– надавати достовірну інформацію щодо результатів власної навчальної (наукової, творчої) діяльності, використаних методик досліджень та джерел інформації;

– не використовувати результати досліджень інших авторів без використання покликань на їхню роботу;

– своєю діяльністю сприяти збереженню та примноженню традицій університету, формуванню його позитивного іміджу;

– не чинити правопорушень і не сприяти їхньому скоєнню іншими особами;

– підтримувати атмосферу довіри, взаємної відповідальності та співпраці в освітньому середовищі;

– поважати честь, гідність та особисту недоторканність особи, незважаючи на її стать, вік, матеріальний стан, соціальне становище, расову належність, релігійні й політичні переконання;

– не дискримінувати людей на підставі академічного статусу, а також за національною, расовою, статевою чи іншою належністю;

– відповідально ставитися до своїх обов'язків, вчасно та сумлінно виконувати необхідні навчальні та науково-дослідницькі завдання;

– запобігати виникненню у своїй діяльності конфлікту інтересів, зокрема не використовувати службових і родинних зв'язків з метою отримання нечесної переваги в навчальній, науковій і трудовій діяльності;

– не брати участі в будь-якій діяльності, пов'язаній із обманом, нечесністю, списуванням, фабрикацією;

– не підроблювати документи;

– не поширювати неправдиву та компрометуючу інформацію про інших здобувачів вищої освіти, викладачів і співробітників;

– не отримувати і не пропонувати винагород за несправедливе отримання будь-яких переваг або здійснення впливу на зміну отриманої академічної оцінки;

– не залякувати й не проявляти агресії та насильства проти інших, сексуальні домагання;

– не завдавати шкоди матеріальним цінностям, матеріально-технічній базі університету та особистій власності інших студентів та/або працівників;

– не використовувати без дозволу ректорату (деканату) символіки університету в заходах, не пов'язаних з діяльністю університету;

– не здійснювати і не заохочувати будь-яких спроб, спрямованих на те, щоб за допомогою нечесних і негідних методів досягати власних корисних цілей;

– не завдавати загрози власному здоров'ю або безпеці іншим студентам та/або працівникам.

УСВІДОМЛЮЮ, що відповідно до чинного законодавства у разі недотримання Кодексу академічної доброчесності буду нести академічну та/або інші види відповідальності й до мене можуть бути застосовані заходи дисциплінарного характеру за порушення принципів академічної доброчесності.

_____	_____	_____
(дата)	(підпис)	(ім'я, прізвище)