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. Електронна залікова книжка та електронна відомість

Кваліфікаційна робота(проєкт)
на здобуття ступеня вищої освіти магістр

Виконав: здобувач 2 курсу 261М групи
Спеціальності 126 Інформаційні системи та технології
Освітньо-професійної програми
Інформаційні системи та технології другого (магістерського) рівня вищої освіти
Нагачевський Андрій Олександрович
Керівники: Доктор педагогічних наук, професор Круглик Владислав Сергійович,
Кандидат фізико-математичних наук,
доцент Єрмолаєв Вадим Анатолійович
Рецензент: кандидат фізико-математичних наук, доцент
Котова Ольга Володимирівна

Херсон – 2021

ЗМІСТ

ВСТУП

Актуальність роботи. В останні роки потреба у власній системі управління бізнес -процесами університету задовольняє потреби як в Україні, так і у всьому світі.

Все це пов’язано з розвитком інформаційно -комунікаційних технологій, університетська система починає відігравати надзвичайно важливу роль у житті університету. Велика кількість людей залучена до таких типів мереж.

Об’єктом дослідження є управління бізнес -процесом, електронний буклет та електронний звіт.

Предмет дослідження є архітектура служби управління бізнес -процесами університету. Мікропослуги «Виписка з рахунку», «Буклет»

Метою дослідження є розроблення архітектури послуг для управління університетськими бізнес -процесами та розробки мікросервісів Електронне декларування та тестовий зошит

Усередині роботи:

• аналіз додатків та комерційних систем, що використовуються в інших університетах,

• аналіз структури залікової книги та відомості

• Сервісна архітектура, призначена для управління університетськими бізнес -процесами

• Завдання щодо розробки залікової книжки та модулів інформаційних мікросервісів.

• Розроблено мікросервіс «Залікова книга» та «Відомість».

для системи управління бізнес -процесами Університету.

Розроблений мікросервіс може бути використана за призначенням.

РОЗДІЛ 1

Загальні відомості про організацію сервісів для управління бізнес-процесами університету

1.1 Огляд наявних сервісів для управління бізнес-процесами університету.

Університет Мілана - це сервіс для управління бізнес-процесами університету який був розроблений найбільшим державним університетом у Мілані і єдиний італійський університет, що входить в Лігу європейських дослідницьких університетів.

Університет Болонії - це сервіс для управління бізнес-процесами університету який був розроблений найбільшим державним університетом у Болонії.

Університет Пізи - це сервіс для управління бізнес-процесами університету який був розроблений найбільшим державним університетом у Пізі.

Університет Флоренції - це сервіс для управління бізнес-процесами університету який був розроблений найбільшим державним університетом у Флоренції.

Університет	Управління навчальними матеріалами	Управління оцінкою	Управління бібліотекою	Управління кампусом	Управління навчальними програмами	Управління фінансовою допомогою	Соціальна освіта	Онлайн-платежі	Управління доступом	Фінансовий менеджмент	Портал для батьків та учнів	Управління випускниками	Факультет\управління персоналом	Управління збіру коштів	Планування	Інформація для студентів\записи	Студентський портал	Інструменти звязку	Календар подій	Інтерактивне навчання	Опитування\голосування	Управління безпекою	Управління студентською групою
Болонии	*	-	*	*	*	*	*	-	*	*	*	*	-	-	*	-	-	*	*	*	*	*	-
Милан	*	-	*	-	*	*	*	*	-	*	*	*	*	*	-	-	*	-	*	*	*	*	-
Пиза	*	-	*	-	-	*	*	*	-	*	-	-	*	-	*	*	*	*	*	*	-	*	-
Флоренция	*	*	*	-	*	*	*	-	*	*	*	-	*	-	*	-	*	-	*	*	-	*	*

Рисунок 1.1 - Матриця порівняння існуючих систем

Рисунок 1.2 - Аналіз використання залікової книжки

Переваги та недоліки використання електронної книжки:

Переваги:

1. Електронна книга вважається оптимальним інструментом для моніторингу показників ефективності;

2. Можливість аналізу оцінок за попередні роки для визначення динаміки навчання та статистики:

2.1. Визначення сильних та слабких сторін під час навчання для адаптації вже набутих знань;

2.2. Дослідження та аналіз оцінок за навчальні дисципліни;

2.3. Використання отриманої інформації для підготовки звітів та інших документів;

3. Впровадження інформаційної культури в життя учнів та вчителів;

4. Наявність інформації:

4.1. Чітка структура поданої інформації та виклад матеріалу;

4.2. Можливість використання ресурсу в будь-який час і в будь-якому місці за наявності Інтернету;

5. Зручний метод використання: не може бути втрачений або зіпсований зовнішніми факторами, такими як аналог паперового носія;

Недоліки:

1. Складність використання вчителями третього покоління:

1.1. Невелика сумка досвіду використання інформаційних технологій;

1.2. Недостатні знання у сфері інформаційних технологій та послуг;

1.3. Відсутність або обмежена кількість профорієнтаційних заходів, спрямованих на набуття нових навиків у сфері технологій.

2. Фінансові бар’єри для впровадження нових інформаційних систем та програм;

3. Неможливість або перешкоди для використання в деяких селах та інших населених пунктах через відсутність доступу до Інтернету;

4. Фінансові труднощі для невузівських студентів щодо підтримки доступу до Інтернету.

5. Шкода фізичному здоров'ю через збільшення використання пристроїв.

1.2 Вибір програмних засобів для розробки системи

Перед тим як обрати засіб для розробки системи ми виявили найпопулярніші та зробили їх аналіз для вибору оптимального.

C++ - це мова програмування високого рівня з підтримкою кількох парадигм програмування: об'єктно-орієнтованої, узагальненої та процедурної. Розроблений Б'ярне Страуструпом у лабораторіях AT&T Bell Laboratories (Мюррей -Хілл, штат Нью -Джерсі) у 1979 році, спочатку він називався «C з класами». Згодом Straustrup змінив назву мови на C ++ у 1983 році. Він базується на мові програмування C. Він був вперше описаний стандартом ISO / IEC 14882: 1998, найбільш актуальним є стандарт ISO / IEC 14882: 2014 рік.

У 1990 -х роках C ++ став однією з найбільш широко використовуваних загальних мов програмування. Мова використовується для системного програмування, розробки програмного забезпечення, написання драйверів, потужних клієнтських і серверних програм, а також для розробки розважальних програм, таких як відеоігри. C ++ значно вплинув на інші популярні сьогодні мови програмування: C # та Java[32].

Java - Мова запозичила велику кількість синтаксису у C та C ++. Зокрема, за основу береться об'єктна модель C ++, але модифікована.

Ми усунули можливість виникнення деяких конфліктних ситуацій, які могли виникнути через помилки програміста, а процес розробки об’єктно-орієнтованих програм спростився. Ряд дій, які програмісти повинні виконати на C / C ++, покладаються на віртуальну машину. Java в першу чергу була розроблена як незалежна від платформи мова, тому вона має менші апаратні можливості низького рівня, що уповільнює роботу програм порівняно з, наприклад, C ++. При необхідності Java дозволяє викликати підпрограми, написані іншими мовами програмування[33].

Node.js - це платформа з відкритим кодом для високопродуктивних мережевих програм, написаних на JavaScript. Засновник платформи - Райан Дал. Якщо раніше Javascript використовувався для обробки даних у браузері користувача, node.js надав можливість запускати сценарії JavaScript на сервері та надсилати результат їх виконання користувачеві. Платформа Node.js перетворила JavaScript на спільну мову з великою спільнотою розробників.

Node.js має такі властивості:

• однопотокова модель виконання асинхронного запиту;

• Неблокуючий ввід / вивід;

• система модулів CommonJS;

• движок JavaScript Google V8;

Менеджер пакетів npm npm використовується для управління модулями[34].

Python - це високорівнева інтерпретована об'єктно-орієнтована мова програмування із суворим динамічним написанням. Структури даних високого рівня разом із динамічною семантикою та динамічним зв’язуванням роблять її привабливою для швидкої розробки програм, а також засобом поєднання наявних компонентів. Python підтримує

модулі та пакети модулів, які сприяють модульності та повторному використанню коду. Інтерпретатор і стандартні бібліотеки Python доступні у вихідних та компільованих форматах на всіх основних платформах. Мова програмування Python підтримує кілька парадигм програмування, включаючи об'єктно-орієнтовану, процедурну, функціональну та аспектно-орієнтовану.

• чистий синтаксис (для вибору блоків необхідно використовувати відступи);

• портативність програми (характерна для більшості інтерпретованих мов);

• стандартний дистрибутив має велику кількість корисних модулів (включаючи модуль для розробки графічного інтерфейсу);

• можливість використання Python у режимі діалогу (дуже корисно для експериментів та вирішення простих задач);

• стандартний дистрибутив має просте середовище розробки, але в той же час досить потужне, яке називається IDLE і написане на Python;

• зручний для вирішення математичних задач (він має засоби для роботи зі складними числами, може оперувати цілими числами будь - якого розміру, у діалоговому режимі може використовуватися як потужний калькулятор);

• відкритий вихідний код (можливість змінювати його іншими користувачами)[35].

Ruby — це інтерпретована, повністю об'єктно-орієнтована мова програмування з чіткою динамічною типізацією. Мова вирізняється високою ефективністю розробки програм і увібрала в себе найкращі риси Perl, Java, Python, Smalltalk, Eiffel, Ada і Lisp. Ruby поєднує в собі Perl-подібний синтаксис із об'єктно-орієнтованим підходом мови

програмування Smalltalk. Також деякі риси запозичено із мов програмування Python, Lisp, Dylan та CLU.

Багатоплатформова реалізація інтерпретатора мови Ruby поширюється як Вільне програмне забезпечення. Початковий код проекту розповсюджується під ліцензіями BSD («2-clause BSDL») і «Ruby», яка посилається на останній варіант ліцензії GPL і повністю сумісна з GPLv3[36].

	Императивная	Объектно-ориентированная	Функциональная	Рефлексивная	Логическая	Статическая типизация	Динамическая типизация	Явная типизация	Неявная типизация	Вывод сигнатуры для локальных функций	Параметрический полиморфизм	Информация о типах в runtime	Информация о типах-параметрах в runtime	Open-source компилятор (интерпретатор)	Возможность компиляции	Bootstrapping	Многопоточная компиляция	Интерпретатор командной строки	Условная компиляция	Создание объектов на стеке	Неуправляемые указатели	Ручное управление памятью	Сборка мусора	Поддержка try/catch	Блок else (исключения)	Кортежи	Алгебраические типы данных	Многомерные массивы	Динамические массивы	Ассоциативные массивы	Списковые включения	Целые числа произвольной длины	Целые числа с контролем границ	Интерфейсы	Мультиметоды	Переименование членов при наследовании	Множественное наследование	Анонимные функции	Лексические замыкания	Частичное применение	Макросы	Шаблоны/Generics	Поддержка Unicode в идентификаторах	Перегрузка функций	Динамические переменные	Значения параметров по умолчанию	Локальные функции	Контрактное программирование
C++	+	+	+/-	-/+	–	+	–	+	-/+	+/-	–	-/+	-/+	+	+	+	–	+	+	+	–	+	+	-/+	+	–	+/-	–	+	+	+	–	–	–	+	–	-/+	+	+	+	+/-	+	+	–	–	+	+	–
Java	+	+	+/-	+	–	–	+	–	+	–	–	-/+	-/+	+	+	+	–	–	-/+	–	–	–	+	+	–	–	–	+/-	+/-	+	–	–	–	-/+	–	-/+	+	+	+	–	–	+	-/+	–	+	+	–
Node JS	+	+	+	+	–	–	+	+/-	+	–	–	+	+	+	+	+	–	+	–	–	–	–	+	+	–	+	–	+/-	+/-	+	+	+	–	+	–	–	+	+	+	+	–	–	+	–	+	+	+/-
Python	+	+	+	+	–	–	+	–	+	–	–	+	+	+	+	+	–	+	–	–	–	–	+	+	+	+	–	+/-	+/-	+	+	+	–	-/+	-/+	-/+	–	+	+	+	+	–	–	+	–	+	+	+/-
Rubi	+	+	-	-	–	+	–	+	+	+	–	-	+	+	+	+	–	+	+	+	+	+	-	+	–	+	–	+	+	+	+	+	+	+	-	-	+	+	+	-	+	+	+	+	+	+	–

Рисунок. 1.3 - Матриця порівняння програмних засобів для розробки системи

Вибір бібліотеки React JS

Бібліотека React була вперше випущена компанією Facebook в 2013 році. Для того, щоб зрозуміти, наскільки популярною ця технологія стала за минулий час, давайте звернемося до опитування розробників, проведеним сайтом StackOverflow в цьому році. Більш 50 000 розробників поділилися інформацією про свою роботу і професійних перевагах. Крім більш-менш передбачуваних результатів, які описують стан IT-індустрії на сьогоднішній день, є також і дещо цікаве щодо безпосередньо React. Бібліотека стала однією з

найпопулярнішою і потребуючою технологією, а також сама трендова технологія на StackOverflow:

Virtual DOM може підвищити продуктивність високонавантажених додатків, що може знизити ймовірність виникнення можливих незручностей і покращує користувальницький досвід;

Використання ізоморфного підходу допомагає виробляти рендеринг сторінок швидше, тим самим дозволяючи користувачам відчувати себе більш комфортно під час роботи з вашим додатком. Пошукові системи індексують такі сторінки краще. Оскільки один і той же код може бути використаний як в клієнтської, так і в серверній частині програми, немає необхідності в дублюванні одного і того ж функціоналу. В результаті час розробки і витрати знижуються;

Завдяки перевикористанню коду стало набагато простіше створювати мобільні додатки. Код, який був написаний під час створення сайту, може бути знову використаний для створення мобільного застосування. Якщо ви плануєте використовувати не тільки сайт, але і мобільний додаток, немає необхідності наймати дві великі команди розробників.[38]

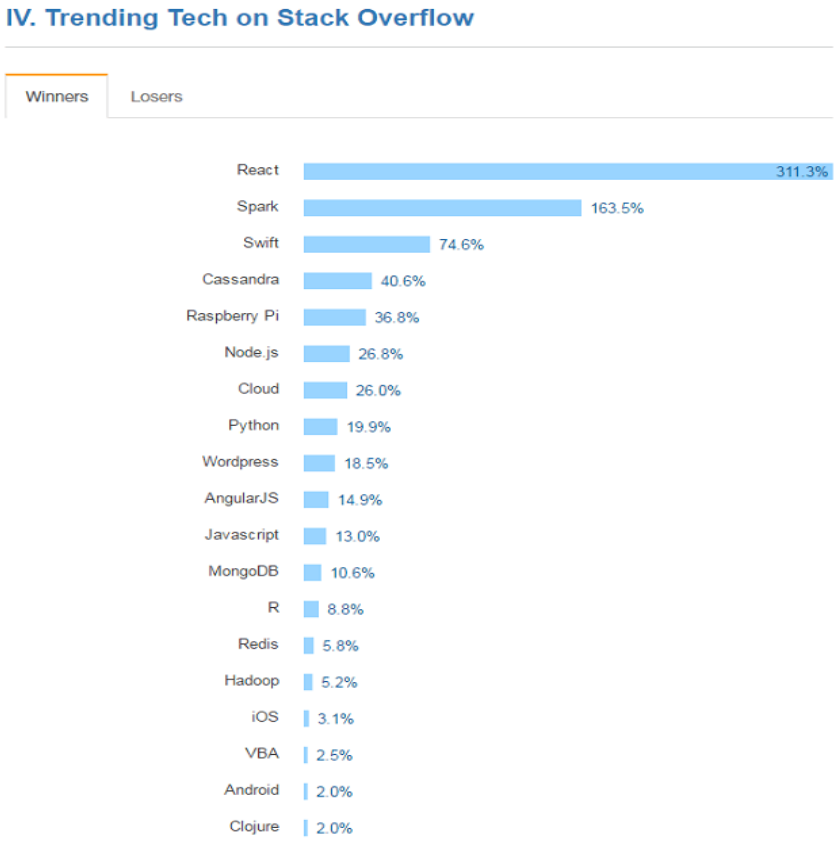

Рисунок 1.4 - Статистика бібліотек

Порядок оформлення Індивідуального навчального плану студента.

Індивідуальна навчальна програма студента розроблена кураторами ECTS за допомогою програмного забезпечення "Університет" IASU із загальної бази навантажень університету у спеціальному електронному меню "Індивідуальний план навчання". Наприкінці поточного року для кожного навчального року формується індивідуальний план навчання. Для першокурсників - до школи. Вибрані Університетом нормативні дисципліни та дисципліни, які повинні вивчатися студентом відповідно до навчального плану роботи, друкуються в індивідуальній навчальній програмі студента автоматично після введення редактора ECTS у відповідне меню їх закладу (факультету) та обрали ім'я студента. Дисципліни вільного

вибору студентів, які згідно з наказом "Про включення дисциплін вільного вибору студентів до індивідуальних навчальних планів студентів закладу" повинні бути вивчені студентом у наступному навчальному році, вони надруковані в необхідному розділі. Наприкінці навчання редактор ECTS роздруковує індивідуальну навчальну програму для студентів у 2 -х примірниках, один для студента, інший - в Дирекції Інституту (деканаті), що додається до навчального квитка студента. Індивідуальна друкована навчальна програма студента підписується редактором ECTS, студентом та затверджується директором закладу (деканом факультету). [31]

1.3 Обґрунтування функціоналу сервісу.

Після аналізу популярних схожих додатків, ми підвели підсумки в їх роботі та розробили власну структуру системи управління бізнес-процесами університету сервісу яка б нас задовольнила.

Ми розробили потрібний функціонал для нашого сервісу:

1. Секретар учнів.
    1.1.1. Увійти / Вийти.
    1.1.2. Інформація про користувача.
2. Мікросервіс «Програма».
    2.1.1.1. Програмне навчання.
    2.1.1.2. Формування розкладу тестів, тестів, диференційованих іспитів.
    2.1.1.3. Відредагуйте інформацію про навчальний сеанс.
3. Мікросервіс «Електронний журнал».
4. Мікросервіс «Електронний тестовий зошит».
5. Мікросервіс «Виписка електронного рахунку».
6. Мікросервіс «Успіх індивідуальної навчальної програми учня».
7. Служба «Відділ кадрів».

7.1. Облік інформації про кількісний та якісний персонал університету.

7.2. Облік бухгалтерської звітності та документації (включаючи запити з особистих питань).

7.3. Облік, складання та ведення трудових книжок.

7.4. Облік, складання та зберігання особистих справ та посвідчень працівників.

7.5. Надати державну статистичну звітність з кадрових питань.

7.6. Надати діючим та попереднім свідоцтвам про роботу працівників.

7.7. Розрахунок трудового стажу працівників.

7.8. Контроль за визначенням збільшення (компенсацій) за вислугу років (якщо це передбачено положеннями про оплату праці).

7.9. Облік відпусток працівників.

7.10. Облік особистих справ учнів усіх форм навчання.

7.11. Вести облік працівників відділу кадрів, адміністрації.

7.12. Підготовка до Міністерства освіти і науки України кадрового модуля КДУ.

7.13. Вона працює в інформаційно -аналітичній системі університету "Особистий" та "Контингент".

7.14. Оформлення лікарняного.

7.15. Запис особистих справ студентів, які передаються до КДУ з інших вищих навчальних закладів.

7.16. Робота з особовими справами студентів, отриманих від приймальної комісії.

7.17. Впровадження організаційних заходів щодо своєчасного щорічного представлення звітів про майно, доходи, витрати та пасиви фінансового характеру компетентними посадовими особами КСУ.

8. Послуга "Навчальний відділ".

8.1. Участь у підготовці та затвердженні навчальних та робочих планів.

8.2. Контроль за написанням розкладів занять у класі, організацією самостійної роботи студентів, контрольних робіт, семестрових та державних іспитів.

8.3. Аналіз організації та реалізації навчального процесу, результати комплексної перевірки знань з предмету та студентських практик, самостійної роботи, виконання роботи та дипломної роботи.

8.4. Складання календарів навчального процесу.

8.5. Контроль за плануванням та виконанням навчального навантаження професорсько-викладацького складу відповідно до навчальних планів денної, заочної та зовнішньої форм навчання.

8.6. Підготувати, замовити, отримати, видати студентські квитки та дублювати студентські квитки.

8.7. Підготовка, замовлення, прийом, видача навчальних документів (дипломів) та дублікатів дипломів.

8.8. Підготовка, замовлення, отримання, видача Додатків до дипломів та дублікатів Додатків до дипломів.

8.9. Видача вчених ступенів.

8.10. Робота з єдиною державною базою електронної освіти.

8.11. Розробка справ щодо акредитації та ліцензування разом із випускними кафедрами.

8.12. Облік навчальної документації відповідно до вимог розпорядження Міністерства освіти і науки, молоді та спорту України від 29 березня 2012 р. - 384.

8.13. Моніторинг виконання індивідуальних планів учителів.

8.14. Складання даних зі статистичних звітів за стандартними формами.

8.15. Контроль за дотриманням вимог чинного законодавства щодо утримань, поновлення, надання академічної ліцензії, індивідуального плану навчання, надходження навчальних документів у КДУ.

8.16. Ведення документації про переміщення контингенту студентів.

8.17. Спільно з відділом кадрів готуємо документацію та приймаємо та видаємо документи про освіту.

8.18. Аналіз паспортів персоналу відділів.

9. Послуга "ВЗЯВО"

9.1. Послуга анкетування.

9.1.1. Підготовка анкет.

9.1.2. Створення анонімних анкет.

9.1.3. Складання анкет.

9.1.4. Дивіться поточні результати.

9.2. Результати.

9.2.1. Перелік результатів опитування.

9.2.2. Статистичні результати.

9.2.3. Якість опитування.

10. Відділ служби обслуговування.

11. Система реєстрації.

12. Послуга імпорту-експорту.

13. Розробка фронтендної частини послуги.

14. API.

15.Створення структури взаємодії бекенд-частини з фронтендною частиною служби.

РОЗДІЛ 2

Проєктування сервісної архітектури управління бізнес-процесами університету

2.1 Опис структури класів сервісної архітектури управління бізнес-процесами університету та індивідуального плану.

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

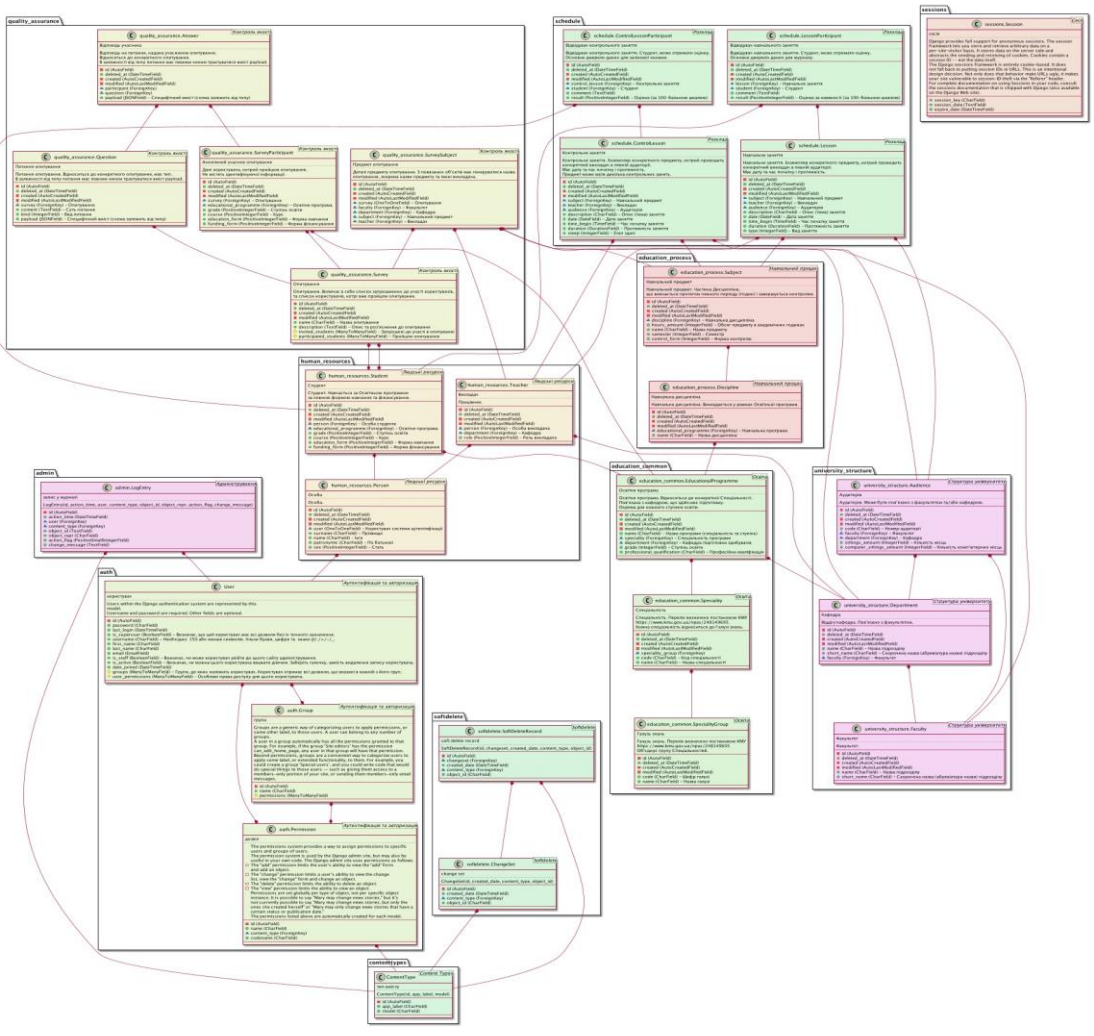

Рисунок 2.1 - Архітектура системи управління бізнес-процесами університету

2.2 Проектування моделі даних мікросервісу "Залікова книжка" та "Електрона відомість"

Мікросервіс “Електронна залікова книжка”.

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

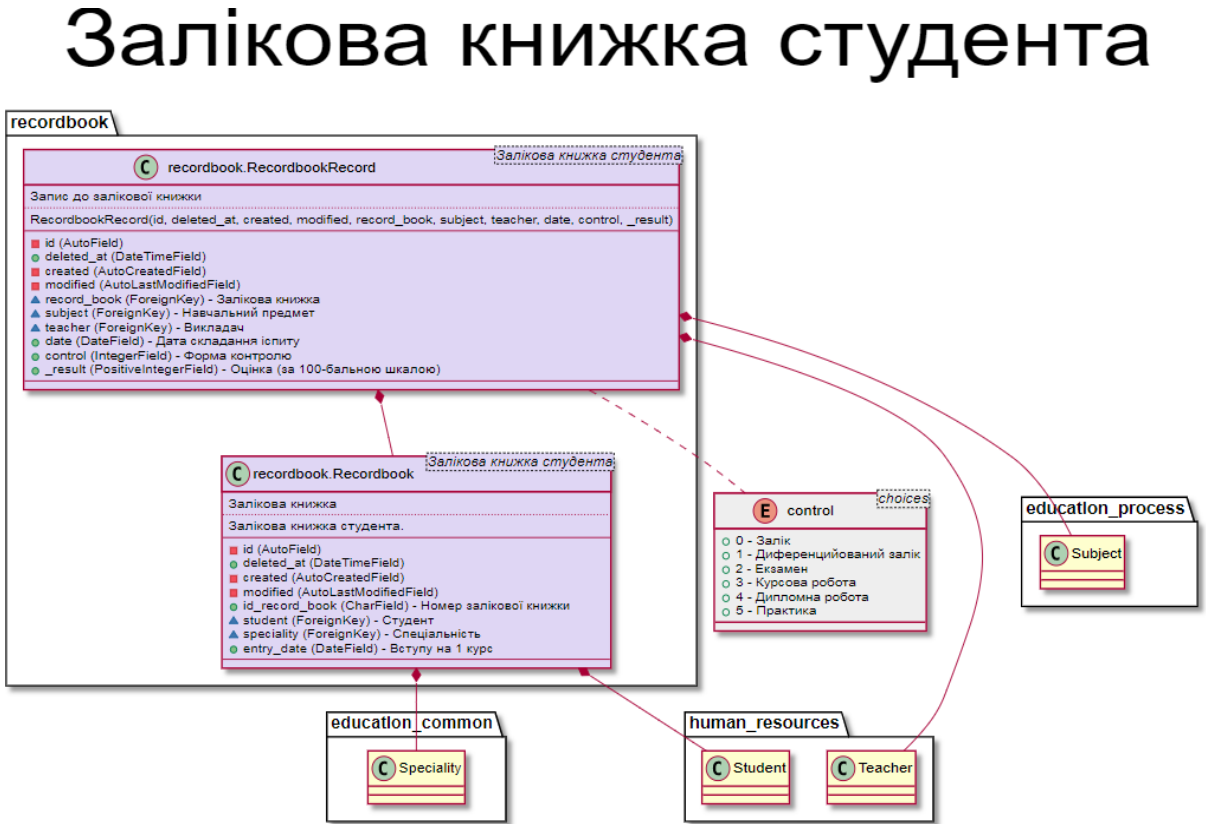

Рисунок 2.2 - Діаграма класів мікросервісу “Електронна залікова книжка студента”.

Мікросервіс “Електронна відомість”.

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

Структура індивідуального плану

Оцінка за 100-баловою шкалою університету	**Оцінка за національною шкалою**		**Оцінка за шкалою ECTS**
	Екзамен	***Залік***	
90-100 (творчий рівень)	**5** (відмінно)	**Зараховано**	***A***
80-89 (високий рівень)	**4** (добре)		**B**
70-79 (достатній рівень)			**C**
65-69 (задовільний рівень)	**3** (задовільно)		**D**
60-64 (задовільний рівень)			**E**
35-59 (низький рівень)	**2** (незадовільно з можливістю повторного складання)	**Незараховано** (з можливістю повторного складання заліку	**FX**
0-34 (незадовільний рівень)	**2** (незадовільно з обов'язковим повторним вивченням дисципліни)	**Незараховано** (з обов'язковим повторним вивченням дисципліни)	**F**

Рисунок 2.4 - кредитно-модульна система організації

2.3 Проектування API

Клієнтський сервер. Основною архітектурою, від якої ви успадковуєте обмеження, є клієнтський сервер. Він вимагає розподілу обов'язків між частинами системи, які займаються зберіганням та обробкою даних (сервер), та тими компонентами, які мають справу з візуалізацією даних на стороні користувача та діями з цим інтерфейсом (клієнт). Таке розділення дозволяє компонентам розвиватися незалежно один від одного та спрощує зміни системи.

Відсутність умов. Інше обмеження полягає в тому, що акти взаємодії між сервером і клієнтом не мають стану, тому кожен окремий запит містить всю інформацію, необхідну для обробки, і це не використовує той факт, що сервер знає ту чи іншу інформацію. з попередньої заявки. Однак відсутність держави не означає, що вона не існувала. Відсутність стану означає, що сервер не знає стан клієнта. Наприклад, коли клієнт запитує домашню сторінку сайту, сервер

відповідає на запитання і забуває про клієнта. Клієнт може залишити сторінку відкритою протягом кількох років, перш ніж натиснути посилання, тоді сервер відповість на наступний запит. Тим часом сервер може відповідати на запити інших клієнтів або нічого не робити, для клієнта це не має значення [15].

Так, наприклад, дані про стан сеансу (користувача, який увійшов у систему) зберігаються на клієнті і передаються окремо з кожним запитом. Це покращує масштабованість, оскільки сервер може звільнити всі ресурси, які використовуються для цієї операції після обробки запиту, для використання в обробці інших запитів, без ризику втрати цінної інформації. Це також полегшує перевірку та пошук помилок, тому що для аналізу того, що відбувається у певному запиті, потрібно лише переглянути цей запит. Надійність зростає, оскільки помилка в одному запиті не впливає на інші.

API використовуються для обміну даними між мікросервісами.

Програмний інтерфейс програми - Application Programming Interface (API), нашого сервісу зв'язуються з нами через API. Ми вибираємо API, нейтральні по відношенню до тих чи інших технологій, щоб гарантувати відсутність обмежень у виборі технологій [15, с. 63].

У нашому випадку база даних являє собою дуже великий спільний API. Також ми будемо використовувати API з більш дрібної структурою, такою, яку можуть надати мікросервіси.

Наші сервіси будуть спілкуються один з одним за допомогою XML або JSON через HTTP і доступний нам варіант полягає в взаємодії призначеного для користувача інтерфейсу з тими API, які показані на рис.2.4. В інтерфейсі на основі веб-технологій для вилучення даних можна скористатися написаними на JavaScript GET-запитами, а для зміни цих даних можна скористатися POST запитами [15, с. 77].

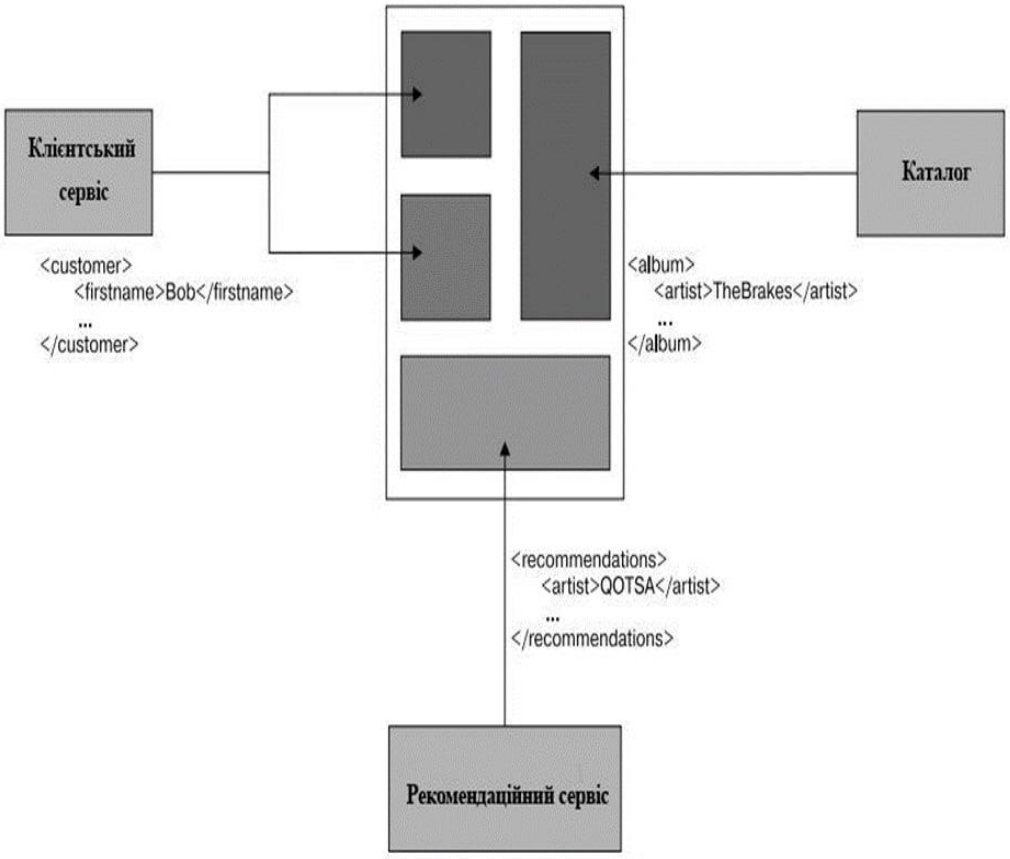

Рисунок. 2.5 - Використання декількох API для представлення інтерфейсу

Але у цього підходу є ряд недоліків. В першу чергу це обмежене коло можливостей для адаптації відповідей під пристрої різних типів. Замість того щоб користувальницький інтерфейс займався API-викликами і відображав всі назад на свої елементи управління, ми можемо зробити так, щоб сервіси надавали частини призначеного для користувача інтерфейсу безпосередньо, а потім, як показано на рис. 2.5, просто вставити ці фрагменти з метою створення призначеного для користувача інтерфейсу.

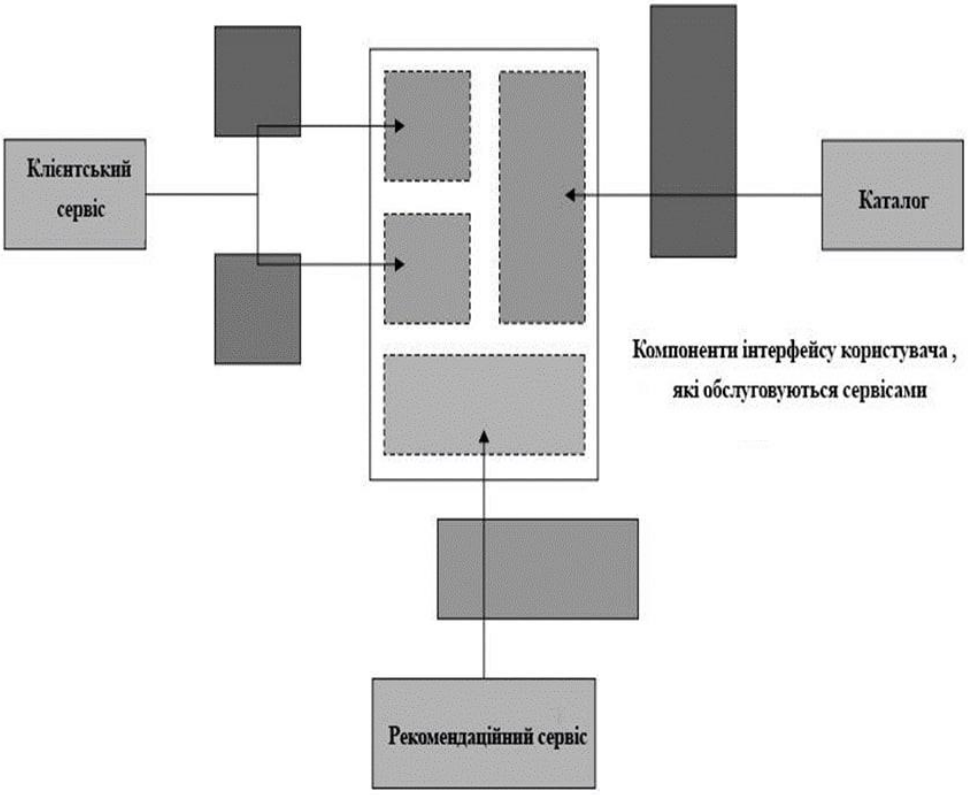

Рисунок 2.6 - Сервіси, які безпосередньо обслуговують компоненти інтерфейсу користувача, призначені для створення збірки.

API-ключі використовуються всіма відкритими API-інтерфейсами таких сервісів, як Twitter, Google, Flickr і AWS. API-ключі дозволяють сервісу ідентифікувати того, хто здійснює виклик, і накласти обмеження на те, що він може зробити. Популярність API-ключів обумовлена тим фактом, що їх застосування для програм зовсім не складно. У порівнянні з обробкою SAML-квітірованія аутентифікація на основі API-ключа набагато простіше і зрозуміліше. Деякі API-системи дозволяють створювати міст від API-ключів до існуючих сервісів каталогів [15, с. 222].

Документація API. Однією із поставлених задач було розроблення документації до публічного API. Саме це є необхідною умовою для забезпечення принципів перевикористання коду і можливості використання проекту або окремих його мікросервісів

сторонніми учасниками. Документацію опубліковано в репозиторії проекту.

РОЗДІЛ 3

Розробка мікросервісу ""Залікова книжка" та "Електрона відомість"

3.1 Структура мікросервісу "Залікова книжка" та "Електрона відомість" та база даних.

Структура мікросервісів які ми розробляємо буде виглядати наступним чином.

1.1. Створення залікової книжки для студента.

1.2. Додавання навчальних дисциплін.

1.3. Додавання викладачем оцінок за вивчений курс.

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

```
class Recordbook(DataObject):
    """
    Залікова книжка студента.
    """
    id_record_book = models.CharField(max_length=10, help_text='Номер залікової книжки')
    student = models.ForeignKey('human_resources.Student', on_delete=models.PROTECT,
                                help_text='Студент')
    speciality = models.ForeignKey(Speciality, on_delete=models.PROTECT, help_text='Спеціальність ')
    entry_date = models.DateField(help_text="Вступу на 1 курс")

    class Meta:
        verbose_name = 'Залікова книжка'
        verbose_name_plural = 'Залікова книжка'
```

Рисунок 3.1 - Клас залікова книжка перша сторінка

На рис 3.1 зображено код класу залікова книжка, де ми робимо записи лицьової частини залікової книжки, а саме номер залікової книжки, спеціальність, дата вступу на дану спеціальність, декан, форма навчання.

```python
class RecordbookRecord(DataObject):
    record_book = models.ForeignKey(Recordbook, on_delete=models.PROTECT, help_text='Залікова книжка')
    subject = models.ForeignKey(Subject, on_delete=models.PROTECT, help_text='Навчальний предмет')
    teacher = models.ForeignKey('human_resources.Teacher', on_delete=models.PROTECT, help_text='Викладач')
    date = models.DateField(blank=True, help_text="Дата складання іспиту", null=True)
    control = models.IntegerField(choices=CONTROL_FORM_CHOICES, default=0, help_text="Форма контролю")
    _result = models.PositiveIntegerField(null=True, blank=True,
                                          validators=[MinValueValidator(0), MaxValueValidator(100)],
                                          help_text="Оцінка (за 100-бальною шкалою)",
                                          db_column="result", )

```

Рисунок 3.2 - Клас залікової книжки друга сторінка

На рис 3.2 зображено код запису залікової книжки головної частини. Тут знаходиться інформація щодо предмету, форму оцінювання, дата проведення чи то заліку чи то екзамену, оцінка та підпис викладача.

```python
def result_to_ects(result: int) -> str:
    assert result > 0
    assert result <= 100
    if result >= 90:
        return ECTS_GRADING_CHOICES.A
    elif result >= 82:
        return ECTS_GRADING_CHOICES.B
    elif result >= 74:
        return ECTS_GRADING_CHOICES.C
    elif result >= 64:
        return ECTS_GRADING_CHOICES.D
    elif result >= 60:
        return ECTS_GRADING_CHOICES.E
    elif result >= 35:
        return ECTS_GRADING_CHOICES.Fx
    else:
        return ECTS_GRADING_CHOICES.F


def result_to_national(result: int) -> str:
    assert result > 0
    assert result <= 100
    if result >= 90:
        return NATIONAL_DIFFERENTIATED_GRADING_CHOICES.Excellent
    elif result >= 74:
        return NATIONAL_DIFFERENTIATED_GRADING_CHOICES.Good
    elif result >= 60:
        return NATIONAL_DIFFERENTIATED_GRADING_CHOICES.Satisfactor
    else:
        return ECTS_GRADING_CHOICES.Fail
```

Рисунок 3.3 Функція переведення оцінки в шкалу ECTS

На цьому рис зображена функція, яка може перевести оцінку із стобальної шкали в ECTS, ми бачимо грані оцінювання тої чи іншої букви в системі ECTS, а також функція яка переводить зі стобальної шкали в результат(задовільно, погано, добре і т.д.).

2. Мікросервіс “Електронна відомість”.

Форма контролю.

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

```
class Bill(DataObject):
    record_book = models.ForeignKey('recordbook.Recordbook', on_delete=models.PROTECT, help_text="Залікова книжка")
    subject = models.ForeignKey('education_process.Subject', on_delete=models.PROTECT, help_text='Навчальний предмет')
    teacher = models.ForeignKey('human_resources.Teacher', on_delete=models.PROTECT, help_text='Викладач')
    control = models.IntegerField(choices=CONTROL_FORM_CHOICES, default=0, help_text="Форма контролю")
    _result = models.PositiveIntegerField(null=True, blank=True,
                                          validators=[MinValueValidator(0), MaxValueValidator(100)],
                                          help_text="Оцінка (за 100-бальною шкалою)",
                                          db_column="result", )

    class Meta:
        abstract = True

    class Meta:
        verbose_name = 'Ведомость'
        verbose_name_plural = 'Ведомость'
```

Рисунок 3.4 - Клас Електрона відомість

Створення відомості та запису до неї, вона буде напряму зв’язана з заліковою книжкою. Дані про навчальний предмет, викладача, кількість годин, акредитація, форма контролю, оцінка за 100-бальною шкалою там результат. Так передбачено можливість зміни оцінки. Через можливість корупції була створено таку функцію як закриття комірки з оцінкою після того як вона була поставлена, щоб її не можна було виправити в відомості, і буде просто функція яка буде давати окрему комірку для перездачі того предмету який ви не змогли здати відразу.

```
class BillRes(Bill):
    date = models.DateField(help_text="Дата складання іспиту")

    class Meta:
        verbose_name = 'Складання іспиту'
        verbose_name_plural = 'Складання іспиту'


class BillResLiquidation(Bill):
    date = models.DateField(help_text='Дата ліквідація академічної заборгованості')

    class Meta:
        verbose_name = 'Ліквідація академічної заборгованості'
        verbose_name_plural = 'Ліквідація академічної заборгованості'

```

Рисунок 3.5 - Клас для перездачі оцінювання

Функція для перездачі іспиту, яка надає окрему комірку на сторінці для внесення даних щодо того як студент буде перездавати іспит, де і коли та хто буде його оцінювати.

3. Мікросервіс “Індивідуальний навчальний план успішності студента”.

Статистичні результати успішності студента.

Система адміністрування баз даних. База даних - це набір даних, організованих за певною прийнятою концепцією, яка описує характеристики цих даних та взаємозв’язок між їх елементами. Дані в базі даних організовані відповідно до моделі організації даних. У загальному випадку базою даних можна вважати будь -який упорядкований набір даних, наприклад, бібліотечний паперовий файл. Але термін "база даних" дедалі частіше використовується в контексті використання баз даних в інформаційних системах, так само, як самі бази даних переносяться до електронних систем у процесі комп’ютеризації. В даний час програми для роботи з базами даних є одними з найпоширеніших. Через тісний зв'язок баз даних із системами управління базами даних (СУБД) термін "база даних" часто вільно позначає систему управління базами даних. Але необхідно розрізняти базу даних - сховище даних та СУБД - засоби роботи з базою даних.

Також у роботі під терміном «база даних», залежно від контексту, її можна розуміти як сукупність даних або деякі її параметри та СУБД, якщо це не очевидно. Розробка перших баз даних розпочалася в 1960 -х рр. Більшість дослідницької роботи проводиться над проектами в IBM та великих університетах. Пізніше, на початку 1970 -х років, Едгар Ф. Код заклав основи реляційної моделі. Ця модель була вперше використана в базах даних Ingres та System R, які були лише прототипами. Однак перші комерційні версії реляційних баз даних від Oracle та DB2 з’явилися у 1980 -х рр. Реляційні бази даних починають успішно замінювати ієрархічні та мережеві бази даних. Починаються дослідження розподілених (децентралізованих) баз даних. Реляційна модель даних Реляційна модель даних: логічна модель даних, вперше описана Едгаром Ф. Коддом. В даний час ця модель є фактичним стандартом, який є ядром більшості сучасних баз даних. Реляційна модель досягає більш високого рівня абстрагування даних, ніж ієрархічна або мережева модель. Стверджується, що "реляційна модель забезпечує засоби опису даних, виходячи виключно з їх природної структури, тобто без необхідності введення будь -якої додаткової структури для цілей машинного представлення". Це означає, що подання даних не залежить від форми їх фізичної організації, що забезпечується використанням математичної концепції відношення[37].

Модель реляційних даних зазвичай включає теорію нормалізації. Дані ідентифікували такі частини реляційної моделі даних:

а) структурний

б) маніпулятор

в) цілісні

Структурна частина моделі визначає, що єдиною структурою даних є нормалізований рівномірний зв'язок.

Нормалізація бази даних Нормалізація схеми бази даних - це процес поділу відносин (таблиць у концепціях СУБД) відповідно до алгоритму нормалізації на різні відносини на основі функціональних залежностей. Звичайна форма визначається як набір вимог, яким відносини повинні відповідати з точки зору надмірності, що потенційно може призвести до логічно неправильних результатів вибірки. Тому схема реляційної бази даних крок за кроком у процесі виконання відповідного алгоритму переходить до першої, другої, третьої тощо. Якщо відношення задовольняє критеріям у нормальній формі та всім попереднім нормальним формам, то це відношення вважається на n -му рівні нормальної форми. СУБД PostgreSQLPostgreSQL - популярна система управління базами даних з відкритим вихідним кодом. Прототип був розроблений в Каліфорнійському університеті в Берклі в 1987 році, пізніше проект Берклі був припинений, а реалізація була опублікована в Інтернеті під назвою

Postgres95 після вдосконалення вихідного коду. Наразі він підтримує та розвиває групу фахівців, які добровільно долучилися до проекту. Сервер PostgreSQL написаний на C. Він поширюється як вихідний код для компіляції. Детальна документація поширюється разом з кодом.

Модель проектування ORMORM-це модель програмування, яка пов'язує бази даних з поняттями об'єктно-орієнтованих мов програмування, створюючи "базу даних віртуальних об'єктів". В об’єктно-орієнтованому програмуванні об’єкти в програмі представляють об’єкти реального світу. Серце проблеми полягає в перетворенні таких об'єктів у форму, в якій вони можуть зберігатися у

файлах або базах даних і їх можна легко отримати пізніше, зберігаючи властивості об'єктів та відносини між ними. Ці об’єкти називаються «постійними». Існує кілька підходів до вирішення цієї проблеми. Деякі пакети вирішують цю проблему, надаючи бібліотеки класів, які можуть автоматично виконувати такі перетворення. Маючи список таблиць у базі даних та об’єктів у програмі, вони автоматично перетворюватимуть запити з одного подання в інше. Розробники фреймворку Django запропонували використовувати Django-ORM, який забезпечує механізм набору запитів високого рівня. Вони дозволяють виконувати оптимізовані запити до бази даних і є оболонкою для прямої генерації низькорівневих запитів до бази даних. моделі даних.

3.1. Розробка компонентів сервісу.

Для розробки компонентів сервісу ми обрали таку середу розробки як PyCharm.

По-перше, ми реалізували модель даних електронної залікової книжки(Рис. 2.2.) Яку ми можемо побачити далі(Рис. 3.1)

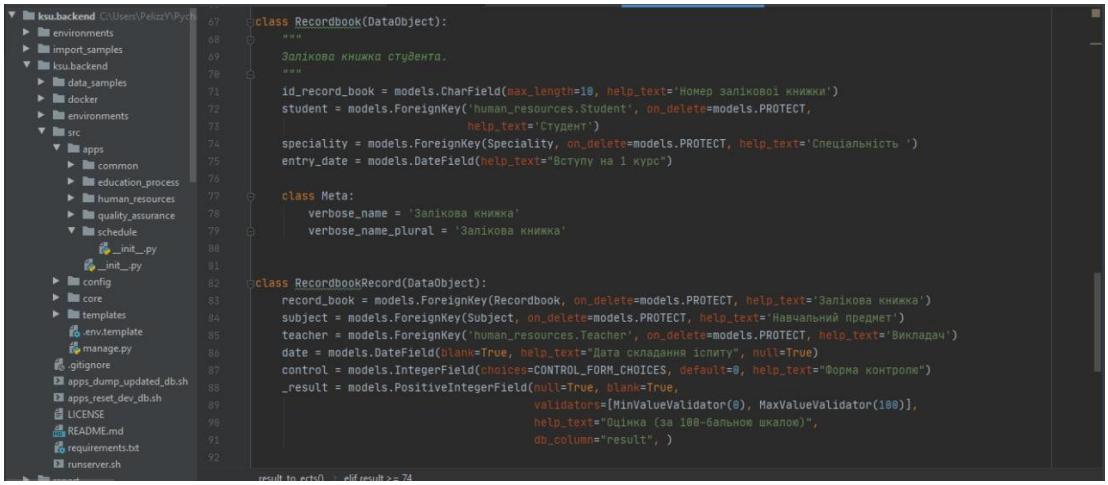

Рисунок 3.6 - Модель даних Електронної залікової книжки.

По-друге, ми реалізували модель даних електронної відомості. Яку ми можемо побачити далі (Рис. 3.2).

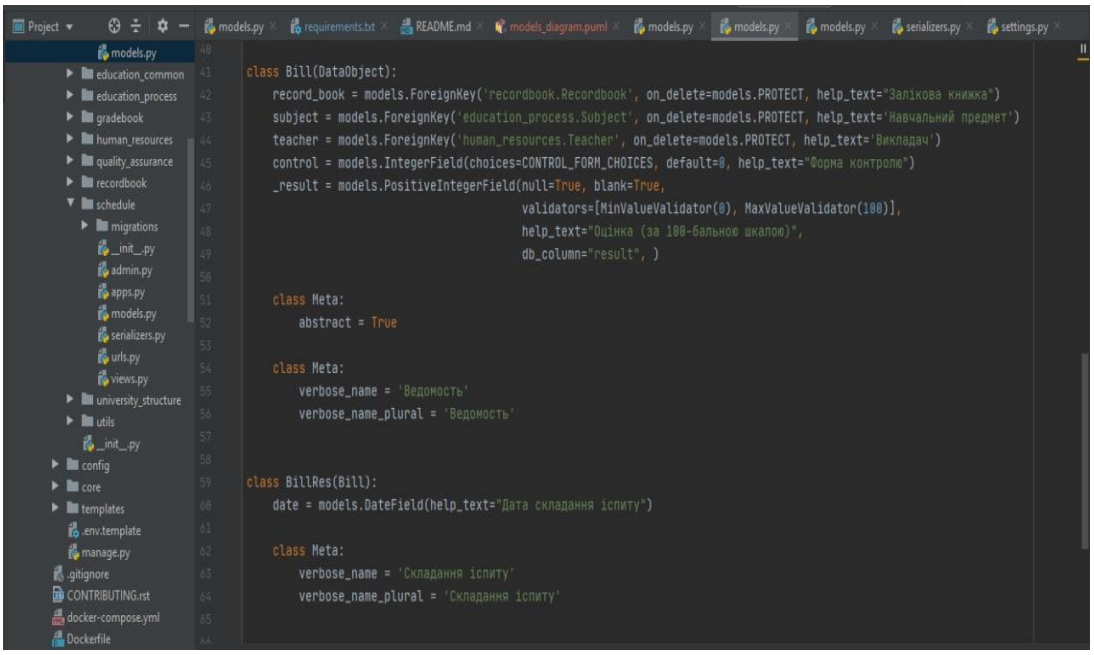

Рисунок. 3.7 - Модель даних Електронної відомості

3.2.Розробка адміністративної частини сервісу.

```python
from django.contrib import admin

from .models import *

__all__ = [
    'RecordbookAdmin',
    'RecordbookRecordAdmin',
]


class RecordbookAdmin(admin.ModelAdmin):
    model = Recordbook


class RecordbookRecordAdmin(admin.ModelAdmin):
    model = RecordbookRecord


admin.site.register(Recordbook, RecordbookAdmin)
admin.site.register(RecordbookRecord, RecordbookRecordAdmin)
```

Рис 3.8 - Додавання проекту до адмін панелі.

Після створення основної частини web-сайту, була розроблена адміністративна частина. За допомогою цієї частини сайт наповнитися необхідним контентом. З метою забезпечення захисту і розмежування доступу в системі перед початком роботи необхідно пройти авторизацію, яка, як показано на Рис. 3.1., містить два поля введення і кнопку.

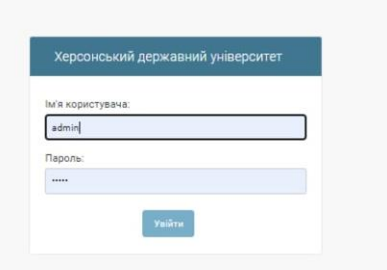

Рисунок. 3.1 - Сторінка авторизації до сервісу

Вікно системи ділиться на дві основні частини: перелік основних розділів і робоча область. У головному меню користувач вибирає адміністративний розділ для другої частини і виконує дії з обраним розділом.

Вікно системи розділене на дві основні частини: список основних розділів і робочу область. У головному меню користувач вибирає адміністративний розділ для роботи, у другій частині - здійснює дії з обраним розділом (Рис. 3.2.).

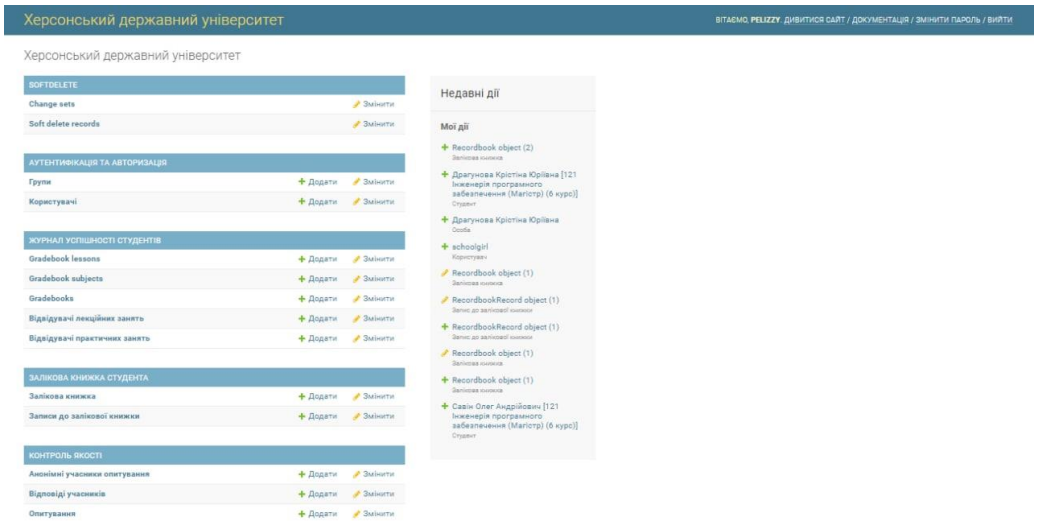

Рисунок. 3.2 - Головна сторінка сервісу

Нами була реалізована електрона залікова книжка. Її можна створити ввівши id залікової книжки студента, ініціали студента, його спеціальність та форму навчання (Рис. 3.3.).

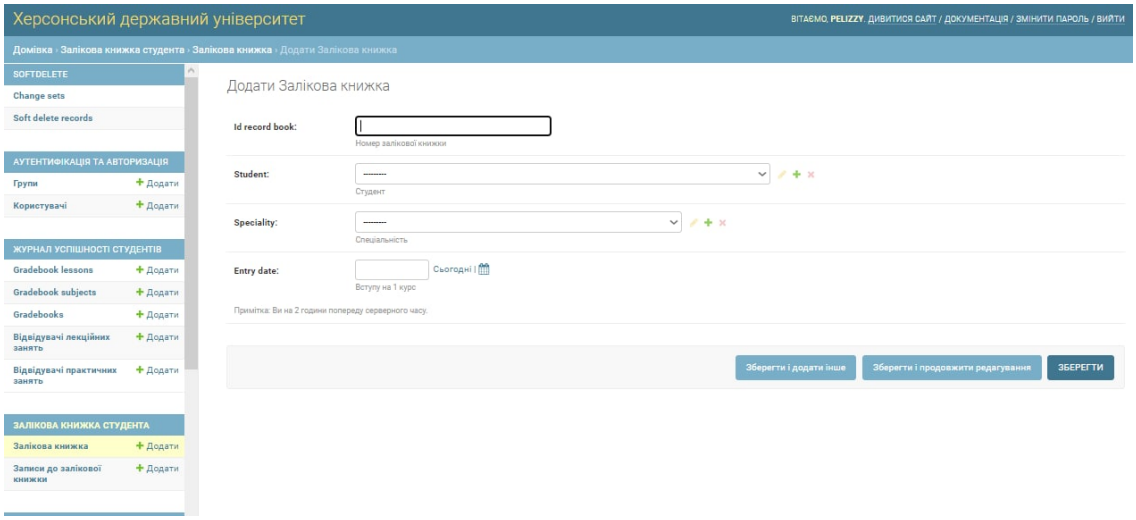

Рисунок. 3.3 - Сторінка Залікової книжки

Після створення електронної залікової книжки ми можемо додавати записи(Рис. 3.4):

- Предмет
- Викладач
- Дата проведення
- Форма контролю
- Результат

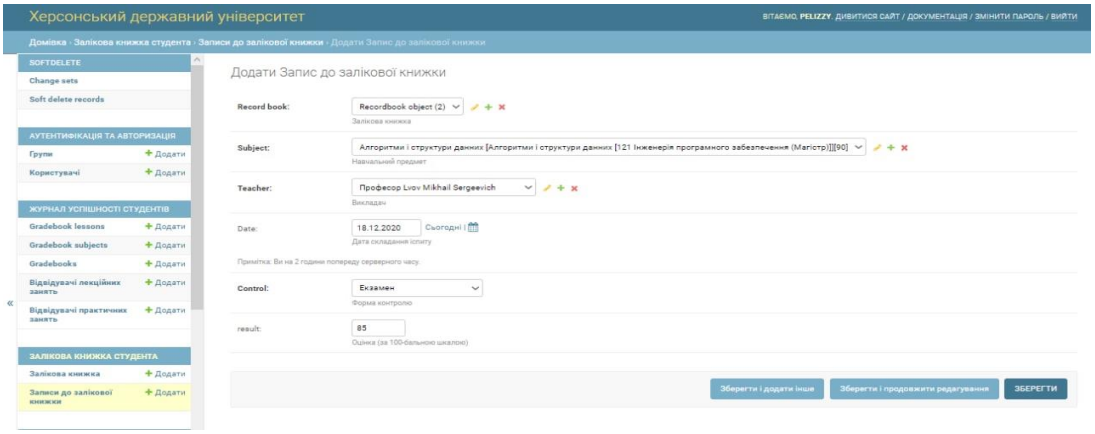

Рисунок. 3.4 - Сторінка запису до залікової книжки

3.4 Тестування програмного продукту.

Після розробки наших мікросервісів нам треба було провести тестування самого програмного продукту на можливі збої та помилки.

1. Функціонал додавання залікової книжки. Функціонал не зазнав збоїв, все працює швидко та без помилок.
2. Функціонал додавання записів до залікової книжки. Функціонал не зазнав збоїв, все працює швидко та без помилок.
3. Функціонал додавання відомості. Функціонал не зазнав збоїв, все працює швидко та без помилок.
4. Функціонал додавання запису до залікової. Функціонал не зазнав збоїв, все працює швидко та без помилок.

ВИСНОВКИ

Для виконання поставлених завдань було проведено аналіз існуючих систем бізнес процесів, виявлено переваги та недоліки кожного з них і обрано оптимальну структуру нашої системи бізнес процесу.

На основі цього аналізу було розроблену такі мікросервіси: Електрона залікова книжка та Електрона відомість. Спроектовано та розроблено серверну частину.

Проаналізувавши вимоги розроблено серверну частину спроектованої системи. Після проведення аналізу популярних технологій розробки вебсервісів для реалізації основної частини системи обрано Node.js

Були виконані наступні завдання:

1) Аналіз наявних сервісів для управління бізнес-процесами університету

2) Вибір програмних засобів для розробки системи

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

4)Розроблено мікросервіс Електрона залікова книжка.

5) Розроблено мікросервіс Електрона відомість.

При розробці проекту використовується система контролю версій git з публічним репозиторієм на сервісі GitHub, 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. Офіційний сайт PandaJs - Режим доступу http://www.pandajs.net/
17. Коржинський С. Н. Настольна книга Web-мастера: ефективне примі неня HTML, CSS и JavaScript.
18. Офіційний сайт Python Режим доступу - https://www.python.org
19. Офіційний сайт вікіпедія Режим доступу - https://ru.wikipedia.org

20. Створення сайтів з Python Режим доступу - http://python-3.ru/page/php-vs-python
21. Владимир Дронов. Практика создания веб-сайтов на Python,. С. 220-235.
22. Алексей Васильев. Python на примерах. Практический курс по программированию, 2019. С. 156-160.
23. Веб-разработка. Исчерпывающее руководство . Режим доступу - https://habr.com/ru/company/piter/blog/314562/
24. 28. Сьюзан Уэйншенк. “100 главных принципов дизайна”, 2010. С. 102-
25. 29. Дмитрий Кирсанов. “Веб-дизайн” 2012. С. 98-105.
26. Итан Маркотт. “Отзывчивый веб-дизайн” 2014. С. 107-120.
27. Співаковський О. В.Web-портал Херсонського державного університету. — 2013. — URL:http://kspu.edu/.
28. 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.
29. 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.
30. Тарек Зиаде. Розробка Python Microservices. 2017. с. 201.
31. Сайт національний педагогічний університет імені М.П. Драгоманова [Електронний ресурс] - URL:http://surl.li/hpxy
32. Мова програмування C++ [Електронний ресурс] Режим доступу: URL:https://uk.wikipedia.org/wiki/C%2B%2B
33. Мова програмування Java [Електронний ресурс] Режим доступу: URL: https://uk.wikipedia.org/wiki/Java

34. Мова програмування Node.js [Електронний ресурс] Режим доступу: URL: https://uk.wikipedia.org/wiki/Node.js
35. Мова програмування Python [Електронний ресурс] Режим доступу: URL:https://uk.wikipedia.org/wiki/Python
36. Мова програмування Python [Електронний ресурс] Режим доступу: URL:https://uk.wikipedia.org/wiki/Ruby
37. Система керування базою даних [Електронний ресурс] URL: http://surl.li/hpxv
38. Чому треба використовувати React Js для створення додатків [Електронний ресурс] URL:https://xbsoftware.ru/blog/pochemu-stoit-ispolzovat-react-js-razrabotke-prilozhenij/

ДОДАТКИ

ПОЛОЖЕННЯ про індивідуальний навчальний план студента НПУ імені М.П. Драгоманова

Індивідуальний навчальний план студента (ІНПС) 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. Форма ІНПС наведена в додатку 1.

1.3. 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 освітньокваліфікаційного рівня та навчальні дисципліни, що вивчаються додатково.

1.4. Виконання ІНПС здійснюється протягом часу, який не перевищує граничного терміну навчання. Нормативний термін навчання визначається відповідно до галузевих стандартів вищої освіти. Студент персонально відповідає за виконання ІНПС.

1.5. Сумарна трудомісткість навчальної роботи з дисциплін, що включено до ІНПС, у кожному семестрі повинна складати не менше 30 3 кредитів (60 кредитів на рік). Враховується час на аудиторні заняття (лекції, практичні, семінарські, лабораторні, індивідуальну роботу), самостійну роботу студента та час проходження навчальних і виробничих практик.

II. Порядок формування ІНПС

2.1. ІНПС формується за відповідною освітньо-професійною програмою і складається студентом під керівництвом куратора ECTS

на кожний рік навчання (на наступний навчальний рік складається в кінці поточного).

2.2. Нормативні дисципліни, що входять до робочого навчального плану відповідного семестру, та дисципліни за вибором Університету є обов'язковими для включення до ІНПС.

2.3. У кожному із циклів дисциплін, які входять до навчального плану (гуманітарної та соціально-економічної, природничо-наукової, професійної та практичної підготовки), студент має можливість обрати вибіркові дисципліни (у межах передбаченої варіативної складової), які в сукупності з нормативними (обов'язковими) формують його індивідуальний навчальний план.

2.4. Студенти першого року навчання отримують ІНПС, сформований випусковою кафедрою до початку навчального року.

2.5. Вивчення навчальних дисциплін вільного вибору студентів освітньо-кваліфікаційного рівня «бакалавр» планується, як правило, починаючи з другого курсу навчання (ІІІ семестру).

2.6. Організація формування вибіркової складової ІНПС «за вибором студентів» для ОКР «бакалавр» починається за півроку до початку вивчення даних дисциплін і здійснюється зазвичай на наступний навчальний рік в такому порядку:

2.6.1. На початку ІІ семестру дирекції інститутів доводять до відома студентів перелік дисциплін вільного вибору студентів на подальший період навчання, порядок їх вивчення, анотації до змісту вибіркових навчальних курсів, організовують зустрічі з викладачами відповідних кафедр та забезпечують консультування кураторів ECTS з усіх питань щодо вибору тієї чи іншої дисципліни;

4 2.6.2. До кінця другого тижня ІІ семестру кожного навчального року студенти подають до дирекції інституту заяви за формою, зазначеною у додатку 2, в яких із запропонованого переліку дисциплін

вільного вибору студентів обирають курси для включення їх до ІНПС наступного навчального року.

2.6.3. Дирекції інститутів на підставі аналізу заяв студентів включають до робочих навчальних планів обрані студентами дисципліни і формують навчальні групи для вивчення вибіркових дисциплін та готують відповідні проекти наказів. Дана процедура має бути завершена до кінця лютого місяця.

2.6.4. Згідно з нормативними вимогами чисельність студентів у групі з вивчення вибіркової дисципліни циклу гуманітарної та соціальноекономічної підготовки та циклу природничо-наукової підготовки, як правило, повинна складати не менше 25 осіб; з вивчення вибіркової дисципліни циклу професійної та практичної підготовки – не менше 12 осіб.

2.7. Після затвердження в установленому порядку робочого навчального плану з переліком вибіркових дисциплін, які студенти будуть вивчати в наступному навчальному році, дирекція інституту (деканат факультету) за участю кураторів ECTS організовує включення відповідних дисциплін до індивідуальних навчальних планів кожного студента.

2.8. Зміни до вибіркової частини свого індивідуального навчального плану студент може внести не пізніше, ніж за два місяці до початку навчального року за обґрунтованою заявою на ім’я директора інституту (на підставі чого готується відповідний наказ). Вказані зміни, внесені до індивідуального навчального плану, затверджуються директором інституту.

2.9. При затвердженні індивідуального навчального плану студента на наступний навчальний рік враховується фактичне виконання студентом плану за попередні роки.

2.10. За умови переведення (поновлення) студента формування індивідуального навчального плану проводиться з урахуванням

дисциплін 1 В іншому випадку, за рішенням дирекції інституту, студентам за сприяння кураторів ЄКТС пропонується скоригувати свій вибір. 5 чинного навчального плану, вивчених у попередньому навчальному закладі (відповідно до академічної довідки).

2.11. По закінченню навчального року, за умови виконання індивідуального навчального плану, наказом по університету студента переводять на наступний курс. III. Порядок оформлення Індивідуального навчального плану студента

3.1. Оформлення ІНПС здійснюється кураторами ECTS за допомогою комп'ютерної програми ІАСУ «Університет» із загальної бази даних навчального навантаження університету в спеціальному електронному меню «Індивідуальний навчальний план студента».

3.2. ІНПС формується на кожний навчальний рік в кінці поточного року. Для студентів І курсу − перед початком занять.

3.3. Нормативні дисципліни та дисципліни за вибором Університету, які мають вивчатися студентом згідно з робочим навчальним планом, вдруковуються в ІНПС автоматично після входження куратора ECTS у відповідне меню свого інституту (факультету) та вибору прізвища студента. Дисципліни вільного вибору студентів, які згідно з наказом «Про включення дисциплін вільного вибору студентів до індивідуальних навчальних планів студентів інституту» має вивчати студент в наступному навчальному році, вдруковуються в необхідний розділ.

3.4. Після завершення формування куратор ECTS роздруковує ІНПС в 2-х примірниках, – один студенту, другий – в дирекцію інституту (деканат) як додаток до навчальної картки студента.

3.5. Роздрукований ІНПС підписується куратором ECTS, студентом та затверджується директором інституту (деканом факультету). IV. Контроль за виконанням ІНПС

4.1. Контроль за виконанням студентами індивідуальних навчальних планів здійснює академічний куратор ECTS.

4.2. Академічні куратори ECTS призначаються наказом ректора за поданням дирекції інституту (деканату факультету). 6

4.3. На академічного куратора ECTS покладається виконання таких основних завдань: - ознайомлення студентів із нормативно-методичними матеріалами, які регламентують організацію навчального процесу за кредитно-модульною системою; - надання кваліфікованих консультацій щодо формування ІНПС, його виконання студентом протягом усього періоду навчання; - оформлення нових ІНПС в частині реквізитів та подання індивідуальних навчальних планів студентів на затвердження в установленому порядку; - оформлення ІНПС в частині переліку навчальних дисциплін та їх обсягів; - перевірка та уточнення особових даних про студента, внесених до електронної бази даних; - відслідковування та внесення на підставі наказів та розпоряджень по університету змін до контингенту студентів відповідної академічної групи, в т.ч. переведення на наступний навчальний рік; - разом з відповідальними особами від кафедр внесення до електронної бази даних результатів поточної та підсумкової успішності студентів; - подання пропозицій стосовно перезарахування залікових кредитів, які студент отримав під час навчання в інших ВНЗ України або за кордоном; - погодження ІНПС та подання його на затвердження директорові інституту (декану факультету); - контроль за виконанням індивідуального навчального плану студентом на підставі відомостей підсумкової атестації з подальшим поданням пропозицій стосовно продовження навчання або його відрахування.

4.4. Академічний куратор ECTS має право: - отримувати для роботи освітньо-професійні програми та освітньокваліфікаційні характеристики підготовки відповідних фахівців, ухвали Вчених та науково-методичних рад, наказів та розпоряджень по 7 університету

стосовно організації та змісту підготовки студентів; - подавати пропозиції директорові інституту (декану факультету) щодо переведення на інший курс, відрахування та заохочення студента; - подавати пропозиції щодо вдосконалення організації навчального процесу та поліпшення роботи кураторів.

V. Оцінювання результатів виконання ІНПС

5.1. Результати підсумкової атестації щодо виконання ІНПС та рівня навчальних досягнень студентів відображаються в заліковій книжці студента (форма № Н-2.03.2. у редакції наказу МОН від 05.06.2013 р. № 683).

5.2. Залікова книжка видається студентам, зарахованим на перший курс, протягом першого семестру, але не пізніше ніж за місяць до початку зимової екзаменаційної сесії.

5.3. Реєстрація залікових книжок проводиться в спеціальній книзі, до якої вносяться наступні дані: - порядковий реєстраційний номер залікової книжки; - прізвище, ім'я та по батькові особи, якій видано залікову книжку; - порядковий реєстраційний номер залікової книжки, який складається з двох чисел, записаних через дріб: порядковий номер і останні дві цифри року, коли видана залікова книжка; - дата видачі залікової книжки; - підпис студента, що отримав залікову книжку.

5.4. Записи у заліковій книжці здійснюються з використанням чорнила або пасти чорного, синього або фіолетового кольору. Виправлення, які не завірено в установленому порядку, не допускаються.

5.5. Протягом усього періоду навчання під час підсумкової атестації до залікової книжки викладачами Університету вносяться дані щодо підсумкової успішності студента з нормативних і вибіркових навчальних дисциплін та результати щодо проходження практики. Під час роботи Державної екзаменаційної комісії до залікової книжки

секретарем ДЕК вносяться результати державної атестації випускника та рішення комісії. 8 Оцінки з навчальних дисциплін виставляються викладачем на сторінці залікової книжки, що відповідає семестру, у якому вивчається ця дисципліна.

5.6. Підсумкове оцінювання знань студента проводиться з урахуванням Положення про кредитно-модульну систему організації навчального процесу в НПУ імені М.П. Драгоманова відповідно до таблиці.

<table>
<tr><th rowspan="2">Оцінка за 100-баловою шкалою університету</th><th colspan="2">Оцінка за національною шкалою</th><th rowspan="2">Оцінка за шкалою ECTS</th></tr>
<tr><th>Екзамен</th><th>Залік</th></tr>
<tr><td>90-100
(творчий рівень)</td><td>5
(відмінно)</td><td rowspan="5">Зараховано</td><td>A</td></tr>
<tr><td>80-89
(високий рівень)</td><td rowspan="2">4
(добре)</td><td>B</td></tr>
<tr><td>70-79
(достатній рівень)</td><td>C</td></tr>
<tr><td>65-69
(задовільний рівень)</td><td rowspan="2">3
(задовільно)</td><td>D</td></tr>
<tr><td>60-64
(задовільний рівень)</td><td>E</td></tr>
<tr><td>35-59
(низький рівень)</td><td>2 (незадовільно з можливістю повторного складання)</td><td>Незараховано
(з можливістю повторного складання заліку</td><td>FX</td></tr>
<tr><td>0-34
(незадовільний рівень)</td><td>2 (незадовільно з обов’язковим повторним вивченням дисципліни)</td><td>Незараховано
(з обов’язковим повторним вивченням дисципліни)</td><td>F</td></tr>
</table>

Рис. 2.4 кредитно-модульна система організації

5.7. До залікової книжки оцінки «незадовільно» і «незараховано» та відповідні бали не записуються, а проставляються лише у відомості підсумкової атестації.

5.8. Під час виставлення екзаменаційної оцінки та відмітки про залік викладач зобов’язаний розбірливо записувати оцінки, своє прізвище, ініціали та проставити особистий підпис.

5.9. Допускається виправлення помилково виставленої оцінки в заліковій книжці студента. У цьому випадку за обґрунтованим зверненням викладача директор інституту (декан факультету) приймає рішення стосовно запису правильного варіанту (цифрою та прописом)

та внизу сторінки додає фразу: «виправленому вірити» й скріплює власний підпис печаткою Оцінка за 100- баловою шкалою університету Оцінка за національною шкалою Оцінка за шкалою Екзамен Залік ECTS 90-100 (творчий рівень) 5 (відмінно) Зараховано A 80-89 (високий рівень) 4 (добре) B 70-79 (достатній рівень) C 65-69 (задовільний рівень) 3 (задовільно) D 60-64 (задовільний рівень) E 35-59 (низький рівень) 2 (незадовільно з можливістю повторного складання) Незараховано (з можливістю повторного складання заліку FX 0-34 (незадовільний рівень) 2 (незадовільно з обов'язковим повторним вивченням дисципліни) Незараховано (з обов'язковим повторним вивченням дисципліни) F 9 інституту.

5.10. По закінченню навчального семестру (навчального року) при виконанні студентом індивідуального навчального плану та успішному складанні підсумкової атестації директор інституту (декан факультету) затверджує результати виконання плану, внесені до залікової книжки, власним підписом та печаткою інституту.

5.11. У випадку відрахування студента з Університету до закінчення курсу навчання за певним освітньо-кваліфікаційним рівнем залікова книжка передається до дирекції інституту (деканату факультету), де студенту видається академічна довідка.

5.12. У разі втрати залікової книжки необхідно: - підготувати клопотання студента на ім'я ректора (проректора) щодо видачі йому дублікату, завізувати його в директора інституту (декана факультету) і подати на розгляд ректору; - підписане ректором (проректором) клопотання передається до студентського відділу Навчально-методичного центру для підготовки наказу про видачу дублікату залікової книжки; - на підставі наказу ректора дирекцією інституту (деканату факультету) видається дублікат залікової книжки, про що робиться відповідний запис в книзі реєстрації залікових книжок. До третьої сторінки дубліката залікової книжки вноситься запис

«дублікат». Усі дані щодо успішності студента за весь період навчання до моменту видачі дубліката вносяться в дублікат залікової книжки на підставі відомостей обліку успішності за попередні семестри.

5.13. За умови успішного засвоєння студентом освітньої програми підготовки з обраного напряму (спеціальності) та успішного складання Державної атестації студенту видається диплом про вищу освіту. [31]

Додаток 1

КОДЕКС АКАДЕМІЧНОЇ ДОБРОЧЕСНОСТІ ЗДОБУВАЧА ВИЩОЇ ОСВІТИ ХЕРСОНСЬКОГО ДЕРЖАВНОГО УНІВЕРСИТЕТУ

Я, _____Нагачевський Андрій Олександрович_____, 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.

18.10.2021 (дата) _____ (підпис) Нагачевський Андрій (ім'я, прізвище)