

ВИПУСКНИЙ КВАЛІФІКАЦІЙНИЙ ПРОЕКТ

на тему:

**«Розробка освітньої інформаційної системи
“Evolution”»**

Студента 2м курсу, бз групи, спеціальності 121 «Інженерія програмного забезпечення», спеціалізації «Інженерія програмного забезпечення»	<i>підпис студента</i>	Усика Сергія Анатолійовича
Науковий керівник Кандидат технічних наук, ст. викладач	<i>підпис керівника</i>	Жирова Тетяна Олександрівна
Гарант освітньої програми доктор технічних наук, професор кафедри інженерії програмного забезпечення та кібербезпеки	<i>підпис керівника</i>	Криворучко Олена Володимирівна

Факультет інформаційних технологій

Кафедра інженерії програмного забезпечення та кібербезпеки

Освітній ступінь магістр

Спеціальність 121 «Інженерія програмного забезпечення»

Затверджую

Зав. кафедри інженерії програмного
забезпечення та кібербезпеки

Криворучко О. В.

8 листопада 2019 р.

Завдання

на випускний кваліфікаційний проект студентів

Усику Сергію Анатолійовичу

(прізвище, ім'я, по батькові)

1. Тема випускного кваліфікаційного проекту «Розробка освітньої
інформаційної системи “Evolution”»

Затверджена наказом ректора від "24" грудня 2019 р. № 4440

2. Строк здачі студентом закінченої проекту 01 грудня 2020

3. Цільова установка та вихідні дані до проекту

Метою даної роботи є розробка інформаційної системи «Online навчання»
для забезпечення надійного і якісного сервісу. А саме завантаження різних
курсів, та матеріалу.

Предмет дослідження розробка веб-сервісу для онлайн навчання

Об'єктом дослідження є програмний сервіс для обслуговування кабінету
викладача та студента системи Online навчання.

4. Консультанти проекту із зазначенням розділів, які консультують:

Розділ	Консультант (прізвище, ініціали)	Підпис, дата	
		Завдання видав	Завдання прийняв

5. Зміст випускного кваліфікаційного проекту (перелік питань за кожним розділом)

ВСТУП

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 ПОСТАНОВКА ЗАДАЧІ

1.2 ОГЛЯД ІСНУЮЧИХ РІШЕНЬ

РОЗДІЛ 2. МОДЕЛЮВАННЯ ПРЕДМЕТНОЇ ОБЛАСТІ

2.1 СТРУКТУРНО-ФУНКЦІОНАЛЬНИЙ АНАЛІЗ І ПРОЕКТУВАННЯ

2.2 ОБ'ЄКТНО-ОРІЄНТОВАНЕ ПРОЕКТУВАННЯ

РОЗДІЛ 3. ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМИ

3.1 ПОБУДОВА ЛОГІЧНОЇ МОДЕЛІ ДАНИХ

3.2 ВИБІР ІНСТРУМЕНТАРІЮ ДЛЯ СТВОРЕННЯ БАЗИ ДАНИХ ТА РОБОТИ З НЕЮ

3.3 ФІЗИЧНА МОДЕЛЬ БАЗИ ДАНИХ

РОЗДІЛ 4. РЕКОМЕНДАЦІЇ ЩОДО ВПРОВАДЖЕННЯ ТА ЕКСПЛУАТАЦІЇ СИСТЕМИ

4.1 ВПРОВАДЖЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

4.2 ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ

4.3 ФУНКЦІОНАЛЬНЕ ТЕСТУВАННЯ

ВИСНОВКИ

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

ДОДАТОКИ

6. Календарний план виконання проекту

№ пор.	Назва етапів випускного кваліфікаційного проекту	Строк виконання етапів проекту	
		за планом	фактично
1	2	3	4
1.	<i>Вибір теми випускного кваліфікаційного проекту</i>	20.09.2019	20.09.2019
2.	<i>Розробка та затвердження завдання на проект магістрам</i>	08.11.2019	08.11.2019
3.	<i>Вступ та перелік літературних джерел</i>	24.01.2020	24.01.2020
4.	<i>Наукова стаття</i>	01.09.2020	01.09.2020
5.	<i>Технічне завдання</i>	28.02.2020	28.02.2020
6.	<i>Розділ 1. Аналіз предметної області</i>	25.06.2020	25.06.2020
7.	<i>Розділ 2. Моделювання предметної області</i>	07.09.2020	07.09.2020
8.	<i>Розділ 3. Інформаційне забезпечення системи</i>	19.10.2020	19.10.2020
9.	<i>Розділ 4. Рекомендації щодо впровадження та експлуатації системи</i>	19.10.2020	19.10.2020
10.	<i>Керівництво користувача</i>	23.10.2020	23.10.2020
11.	<i>Висновки та пропозиції</i>	30.10.2020	30.10.2020
12.	<i>Здача випускного кваліфікаційного проекту на кафедрі (перша перевірка)</i>	05.11.2020	05.11.2020
13.	<i>Підготовка автореферату та презентації доповіді</i>	05.11.2020	05.11.2020
14.	<i>Попередній захист випускного кваліфікаційного проекту</i>	25.11.2020- 27.11.2020	25.11.2020 -27.11.2020
15.	<i>Зовнішнє рецензування випускного кваліфікаційного проекту</i>	01.12.2020	01.12.2020
16.	<i>Підготовка до публічного захисту випускного кваліфікаційного проекту</i>	10.12.2020- 11.12.2020	

7. Дата видачі завдання «8» листопада 2019 р.

8. Науковий керівник випускного кваліфікаційного проекту Жирова Т. О.

(прізвище, ініціали, підпис)

9. Гарант освітньої програми Криворучко О.В.

(прізвище, ініціали, підпис)

10. Завдання прийняв до виконання студент Усик С. А.

(прізвище, ініціали, підпис)

This image shows a single sheet of white paper with horizontal ruling lines. The lines are evenly spaced and run across the width of the page. There is no text or other markings on the paper.

(*nidnuc*, *dama*)

12. Висновок про випускний кваліфікаційний проект

« » 20 p.

АНОТАЦІЯ

Мета дослідження – розробити працюючу систему типу «Програмний сервіс для обслуговування кабінету викладача системи Online навчання», що буде складатися з програмної частини, як веб-сервіс – для викладача який хоче розмістити свої курси, та студента який навчається за допомогою них.

Для досягнення визначеної мети дослідження було проаналізовано існуючі рішення на ринку, їх переваги та недоліки, для того, щоб створити рішення, яке буде відрізнятися від існуючих на сьогодні.

Ключові слова: Веб-Сервіс, он-лайн навчання, викладач, студент.

ABSTRACT

The purpose of the study is to develop a working system such as "Software Service for Online Teacher Training", which will consist of a program part as a web service for a teacher who wants to post their courses and a student who is studying with them.

To achieve the goal of the study, existing market solutions, their advantages and disadvantages were analyzed in order to create a solution that would be different from existing ones.

Keywords: Web Service, online learning, teacher, student.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

АС – Автоматизована Система

БД – База Даних

ДГ - Діаграма

ПЗ – Програмне Забезпечення

ПС – Програмна Система

					КНТЕУ 121 063-15.МР						
Зм.	№	№ докум	Підпис	Дата	«Розробка освітньої інформаційної системи “Evolution”»			Стадія	Аркуш	Аркушів	
Зав. каф.	Криворучко			19.10.20				ПП	2	41	
Керівник	Жирова Т.О.			19.10.20				Факультет інформаційних технологій 2м курс, 63 група			
Гарант	Криворучко			19.10.20							
Розробив	Усик СА.			19.10.20							
					Перелік умовних позначень						

Зміст

ВСТУП.....	4
РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ.....	6
1.1 Постановка задачі.....	6
1.2 Огляд існуючих рішень	8
1.3 Висновки до розділу 1.....	11
РОЗДІЛ 2. МОДЕЛЮВАННЯ ПРЕДМЕТНОЇ ОБЛАСТІ.....	12
2.1 Структурно-функціональний аналіз і проектування.....	13
2.2 Об'єктно-орієнтоване проектування	18
2.3 Висновки до розділу 2.....	23
РОЗДІЛ 3. ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМИ.....	24
3.1 Побудова логічної моделі даних	24
3.2 Вибір інструментарію для створення бази даних та роботи з нею	26
3.3. Фізична модель бази даних.....	27
3.4 Висновки до розділу 3.....	27
РОЗДІЛ 4. РЕКОМЕНДАЦІЇ ЩОДО ВПРОВАДЖЕННЯ ТА ЕКСПЛУАТАЦІЇ СИСТЕМИ.....	28
4.1 Впровадження програмного забезпечення	28
4.2 Програмне забезпечення	30
4.3 Функціональне тестування.....	33
4.4 Висновки до розділу 4.....	37
ВИСНОВКИ	38
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	39
ДОДАТКИ.....	42

					<i>КНТЕУ 121 063-15.МР</i>			
Зм.	№	№ докум	Підпис	Дата	«Розробка освітньої інформаційної системи “Evolution”» Зміст	Стадія	Аркуш	Аркушів
Зав. каф.		Криворучко		19.10.20		3	3	41
Керівник		Жирова Т.О.		19.10.20		Факультет інформаційних технологій 2м курс, 63 група		
Гарант		Криворучко		19.10.20				
Розробив		Усик СА.		19.10.20				

ВСТУП

Дистанційне навчання — це форма навчання з використанням комп'ютерних і телекомунікаційних технологій, які забезпечують інтерактивну взаємодію викладачів та студентів на різних етапах навчання і самостійну роботу з матеріалами інформаційної мережі

У наш час Інтернет-технологій багато аспектів нашого життя переноситься в мережу, прискорюючи тим самим темпи розвитку інформаційного суспільства і долаючи географічні бар'єри. Не стає виключенням і освіта. Зараз вже не обов'язково знаходитись поруч з викладачем. Достатньо великий час існує заочна форма навчання студентів. Але її можливості дуже обмежені. Інтернет дає змогу розширити їх, зробити заочне навчання справді повноцінним та всеохоплюючим.

Дистанційне навчання – сукупність наступних заходів:

- засоби надання учбового матеріалу студенту;
- засоби контролю успішності студента;
- засоби консультації студента програмою-викладачем;
- засоби інтерактивної співпраці викладача і студента;
- можливість швидкого доповнення курсу новою інформацією, коригування помилок.

					<i>КНТЕУ 121 063-15.МР</i>			
Зм.	№	№ докум	Підпис	Дата	«Розробка освітньої інформаційної системи "Evolution"» <i>Вступ</i>	Стадія	Аркуш	Аркушів
Зав. каф.		Криворучко О.В.		24.01.20		В	4	41
Керівник		Жирова Т.О.		24.01.20		Факультет інформаційних технологій 2м курс, 63 група		
Гарант		Криворучко		24.01.20				
Розробив		Усик С.А.		24.01.20				

У наш час в умовах економічних відносин і жорсткої конкуренції на ринку праці особливе значення мають знання, навички та досвід. Фахівець XXI століття - це людина, яка вільно володіє сучасними інформаційними технологіями, постійно підвищує і вдосконалює свій професійний рівень. Придбання нових знань і навичок, практично корисних і застосовуваних у роботі в епоху інформаційного суспільства значно розширює можливості самореалізації і сприяє кар'єрному росту. Проте одною з головних перешкод, що виникає на шляху тих, хто бажає продовжити навчання (враховуючи, що більшість з них вже працює), є брак часу. Більшість не має можливості приїжджати кожного дня на заняття до навчального закладу. Іншою значною перешкодою є відстань. Якщо навчальний заклад розташований в іншому місті, часто відвідувати заняття також незручно і дорого.

«Класична» заочна форма навчання часто не виправдовує свого призначення. Знання, що отримує студент, часто є поверховими, а самі заняття непродуктивними. Крім того, навчальний процес продовжується досить довго.

Об'єктом дослідження є програмний сервіс для обслуговування кабінету викладача системи Online навчання.

Метою даної роботи є розробка інформаційної системи «Online навчання» для забезпечення надійного і якісного сервісу. А саме завантаження різних курсів, та матеріалу.

Гіпотезою дослідження є припущення, що створення та використання системи «Online навчання» сприятиме підвищенню якості дистанційного навчання.

					КНТЕУ 121 063-15.МР	5
Зм.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 1.

АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1. Постановка задачі

Призначенням програмного сервісу для обслуговування кабінету викладача системи online навчання, є полегшення викладачам пошуку аудиторії для своїх уроків, курсів А саме:

- додавання курсів, лекцій, завдань;
- надання студентам сертифікату, про проходження даного курсу;
- можливість перевіряти відправлені роботи;

Інформаційна система повинна обробляти наступні види інформаційних даних відповідно до виділених об'єктів, а саме:

Вхідна інформація:

- *Викладач:*
 - прізвище, ім'я, по-батькові;
 - електронна адреса;
 - логін;
 - пароль.
- *Список курсів:*
 - назва;
 - опис;
 - створення нового курсу;
- *Список завдань:*
 - Тести;
 - лекції;
 - презентація.

					<i>КНТЕУ 121 063-15.MP</i>			
Зм.	№	№ докум	Підпис	Дата				
Зав. каф.		Криворучко О.В.		25.06.20	«Розробка освітньої інформаційної системи "Evolution"» Аналіз предметної області	Стадія	Аркуш	Аркушів
Керівник		Жирова Т.О.		25.06.20		P1	6	41
Гарант		Криворучко		25.06.20		Факультет інформаційних технологій 2м курс, 6з група		
Розробив		Усик С.А.		25.06.20				

1.1.1. Обґрунтування проблеми. Визначення проблем, які вирішить розробка даної системи, подано у табл. 1.1.

Таблиця 1.1.

Проблема	Труднощі пошуку платформи розміщення
Охоплює	Викладачів
Наслідком є	Витрачання значного часу на пошук середовища і вибір платформи, що не задовольняють потреби викладачів
Успішне вирішення	Підвищення якості online освіти

1.1.2. Позиціонування. Визначення позиції виробу та мети створення представлено в табл. 1.2:

Таблиця 1.2.

Для	Київський національний торговельно-економічний університет
Необхідно	Оптимізувати процес online курсів
Назва продукту	Програмний сервіс для обслуговування кабінету викладача системи online навчання
Деталі	Використовуване програмне забезпечення: Графічний інтерфейс розроблено за допомогою технологій HTML5 та мови програмування JavaScript.
На відміну від	Існуючих альтернатив: CourseTalk (coursetalk.org), Class-Central (class-central.com), CourseBuffet (coursebuffet.com)
Наш продукт	Надає можливість вибору курсів та викладачів у відкритому режимі

1.1.3. Опис користувачів. У системі існує два типи користувачів: студент та викладач. Викладач здійснює додавання курсів та завдань на сайт.

Користувачі реєструються в системі для отримання профілю та доступу до ресурсів, авторизуються, обирають категорію, здійснюють пошук курсу по назві, рейтингу або популярності, переглядають опис.

У табл. 1.3 наведений короткий опис профілів користувачів інформаційної системи.

Таблиця 1.3.

Представник	Студент
Опис	Має доступ до інформації про курс, детальний опис.
Відповідальність	Вводить особисті дані.
Представник	Викладач
Опис	Має доступ до інформації про студента. Має змогу додавання та редагування, створеного курсу
Відповідальність	Додає нові курси, здійснює об'єктивний та доступний опис.

1.2. Огляд існуючих рішень

1.2.1. Аналіз потреб. Аналіз потреб користувачів при виборі середовища для розміщення своїх курсів дозволив визначити такі основні проблеми, що стосуються процесу підбору:

- важкість пошук online-ресурсів, які б задовольняли певні потреби користувача, серед множини існуючих в мережі Інтернет;
- трудність до сприйняття інтерфейсу;

Для вирішення зазначених проблем створено значну кількість веб-застосунків.

Для визначення переваг і недоліків альтернативних інформаційних системи автором здійснено аналіз кількох рішень.

1.2.2. Огляд альтернативних систем.

Prometheus — всі курси безкоштовні, українською мовою. Чудова платформа для старту вивчення програмування та основ аналізу даних (рис.1.1).

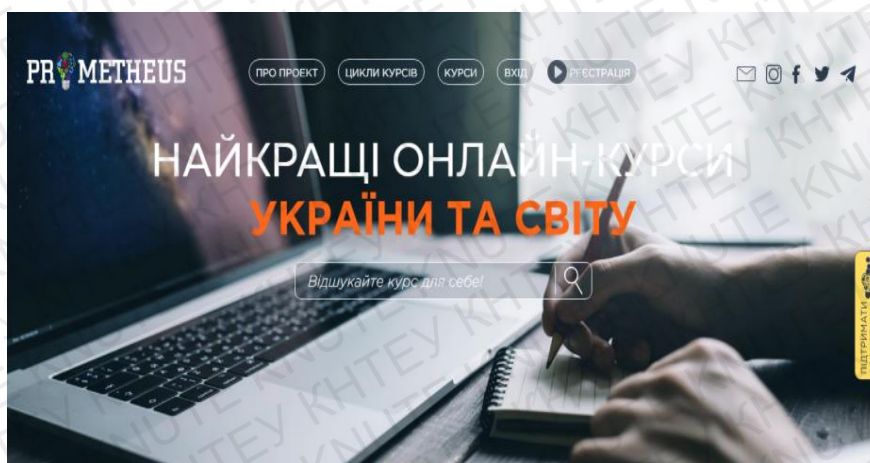


Рис. 1.1. Prometheus

Coursera — хоча навчальні матеріали і лекції безкоштовні, проте контрольні завдання і тести платні. А практичні завдання — це головне для закріплення матеріалу і перевірки знань (рис.1.2)

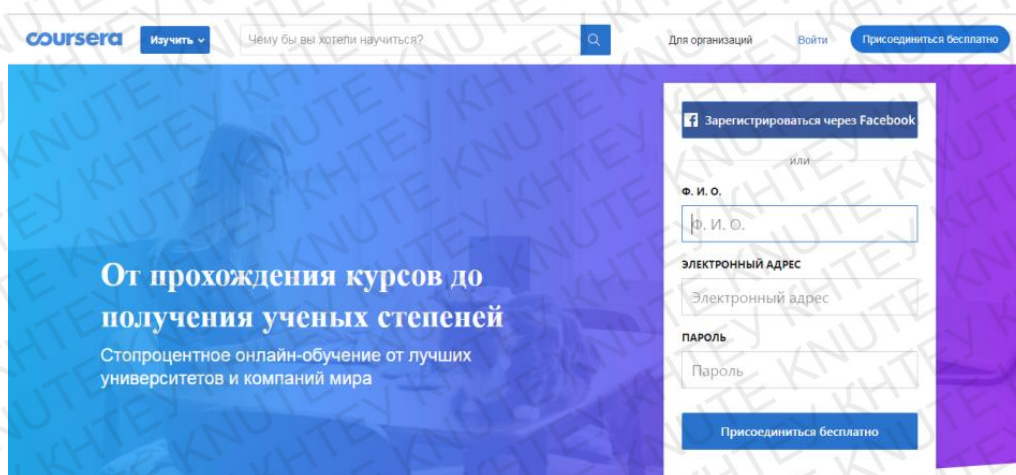
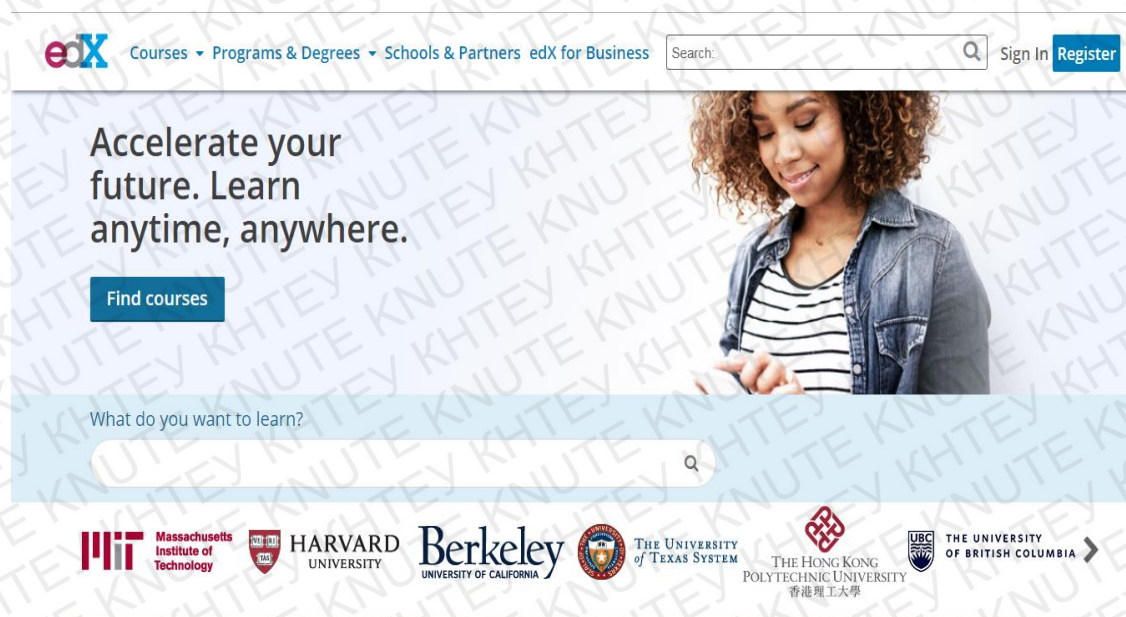


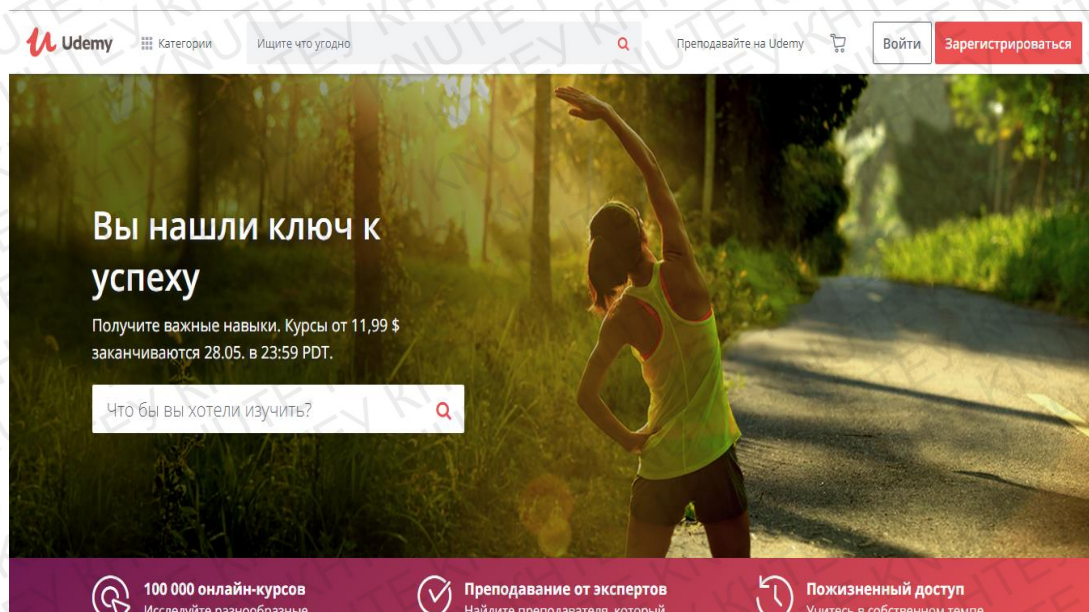
Рис. 1.2. Coursera

edX — навчальні матеріали, лекції, контроль і тести безкоштовні



Puc. 1.3. edX

Udemy — є багато безкоштовних introduction курсів, більшість курсів платні (ціни від 20 до 50 доларів за курс), проте деякі з них справді великі за вмістом, майже як цілі серії (спеціалізації) курсів на Coursera (49 — 490 доларів) (рис. 1.4).



Puc. 1.4. Udemy

Зм.	Арк.	№ докум.	Підпис	Дата

1.4 Висновки до розділу 1

В даному розділі аналізуються запити користувачів, вибираються інформаційні об'єкти та їх характеристики і на основі проведеного аналізу формується структура предметної області, яка не залежить від програмного та технічного середовища, в якому буде реалізовуватися проект.

					<i>КНТЕУ 121 063-15.МР</i>	11
Зм.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 2.

МОДЕЛЮВАННЯ ПРЕДМЕТНОЇ ОБЛАСТІ

Для моделювання предметної області було обрано мову UML (Unified Markup Language).

UML — це мова для визначення, подання, проектування і документування систем різного роду (програмних, організаційно-економічних, технічних тощо). UML містить стандартний набір різноманітних діаграм і нотацій. Це готова до використання мова візуального моделювання для розробки моделей та обміну ними з можливостями розширення і спеціалізації, незалежна від конкретних мов програмування і процесів розробки [4].

UML створено для оптимізації процесу розробки програмних систем, її використання дає можливість багаторазово збільшити ефективність розробки і суттєво поліпшити якість кінцевого продукту. UML застосовують на всіх етапах аналізу бізнес-систем і розробки додатків. За допомогою діаграм подаються як ділові, так і програмні системи. Згодом їх можна перевести у програмний код. Діаграми полегшують супровід проекту і документування.

UML застосовується:

- керівниками проектів під час управління розподілом задач і контролю за їх виконанням;
- проектувальниками при розробці технічних завдань для програмістів;
- бізнес-аналітиками, які досліджують реальну систему і проводять інжиніринг і реінжиніринг бізнес-процесів;
- програмістами, які реалізують модулі інформаційної системи [5].

					КНТЕУ 121 063-15.МР				
Зм.	№	№ докум	Підпис	Дата	«Розробка освітньої інформаційної системи “Evolution”»	Стадія	Аркуш	Аркушів	
Зав. каф.		Криворучко О.В		19.10.20		P2	12	41	
Керівник		Жирова Т.О.		19.10.20		Факультет інформаційних технологій 2м курс, 63 група			
Гарант		Криворучко		19.10.20					
Розробив		Усик С.А.		19.10.20	Моделювання предметної області				



Рис. 2.1. Структура моделі складної ІС нотації UML

2.1. Структурно-функціональний аналіз і проектування

Протягом еволюції мов та принципів програмування програма стала являти собою не просто єдине ціле з важко помітною внутрішньою структурою, а структуру, що складається з чітко виражених модулів, пов'язаних між собою певними відношеннями (інтерфейсами). Тобто програма набула структури ієрархічної багаторівневої модульної системи. Кожен рівень такої системи є закінченим модулем, підтримуваним і контролюваним модулем, що знаходяться над ним.

Методології структурного аналізу і проектування інформаційних систем з'явилися пізніше фактичного використання цих принципів на практиці (структурного програмування).

При аналізі та проектуванні структурним підходом прийнято називати метод дослідження системи, заснований на представленні її у вигляді ієрархії взаємозалежних функцій. Зазвичай опис системи починається з її загального огляду і потім деталізується, набуваючи ієрархічну структуру з усе більшим числом рівнів. Розбиття на рівні абстракції проводиться з обмеженням числа елементів на кожному з них. Опис кожного рівня включає в себе тільки

істотні для цього рівня елементи (принцип абстрагування). Процес розбиття триває аж до конкретних процедур, подальша деталізація яких не має сенсу. При цьому автоматизована система повинна зберігати цілісне уявлення, в якому всі складові її компоненти взаємопов'язані (принцип узгодженості) [6].

2.1.1. Функціональна декомпозиція (FDD – Functional Decomposition Diagram). Функціональна декомпозиція – метод бізнес-аналізу, який розкладає складний бізнес-процес, щоб показати його окремі елементи. Даний метод використовується для полегшення розуміння і управління великими або складними процесами, а також застосовується в області комп'ютерної техніки, а саме при розробці програмного забезпечення [7].

Функціональна декомпозиція інформаційної системи «Підтримка неформального навчання студентів» представлена на плакаті 1.

2.1.2. Діаграма SADT. SADT (Structured Analysis and Design Technique, технологія структурного аналізу і проектування) - графічні позначення і підхід до опису систем. Ідея діаграм SADT лежить в тому, що процес відображається у вигляді прямокутника, в якій входять і виходять стрілки.

Для має значення сторона процесу і пов'язана з нею стрілка:

- зліва входить стрілка - вхід бізнес-процесу - інформація (документ), який буде перетворений в ході виконання процесу;
- справа виходить стрілка - вихід бізнес-процесу - перетворена інформація (документ);
- зверху входить стрілка - управління бізнес-процесу - інформація або документ, який визначає як повинен виконуватися бізнес-процес, як має відбуватися перетворення входу у вихід;
- знизу входить стрілка - механізм бізнес-процесу - то, що перетворює вхід у вихід: співробітники або техніка. Вважається, що за один цикл процесу не відбувається зміни механізму [8].

Functional block diagram of the "Ukrainian Language" website system. The diagram illustrates the flow of data and control between four main functional blocks: 01 (Management of resource users), 02 (Management of learning), A0 (Management of courses), and 04 (Report).

- Block 01: Управління користувачами (User Management)**
 - Inputs: Шаблон додавання ресурсу (Resource addition template), Додання ресурсу (Resource addition), ПІБ (Surname), Email, Пароль (Password), Адреса (Address), Опис ресурсу (Resource description), Логотип ресурсу (Resource logo).
 - Outputs: Акаунт, Дані користувача, логін, Кабінет (Account, User data, login, Cabinet), Статус автора (Author status).
- Block 02: Управління навчанням (Learning Management)**
 - Inputs: Заявка на вивчення (Learning request), Назва категорії (Category name), Назва ресурсу (Resource name), Вподобання (Likes), Коментар (Comment), МВОК (MVOK).
 - Outputs: Ресурс на вивчення (доданий в кабінет) (Resource for learning (added to cabinet)), Сторінка реєстрації на ресурсі (Resource registration page), Вподобання (Likes), Коментар (Comment).
- Block A0: Управління курсами (Course Management)**
 - Inputs: Назва категорії (Category name), Назва ресурсу (Resource name), Адреса ресурсу (Resource address), Текстовий опис ресурсу (Text description of resource), Логотип ресурсу (Resource logo), МВОК (MVOK), Шаблон додавання ресурсу (Resource addition template).
 - Outputs: Зареєстрована категорія (Registered category), Зареєстрований ресурс (Registered resource).
- Block 04: Звіт (Report)**
 - Inputs: Зареєстрована категорія (Registered category), Зареєстрований ресурс (Registered resource), Вподобання (Likes), Коментар (Comment).
 - Outputs: Популярність ресурсів (Resource popularity), Популярність авторів (Author popularity), Вподобання по авторам (Likes by author), Вподобання по ресурсів (Likes by resource), Кількість ресурсів (Number of resources).

KHTEY 121 063-15.MP

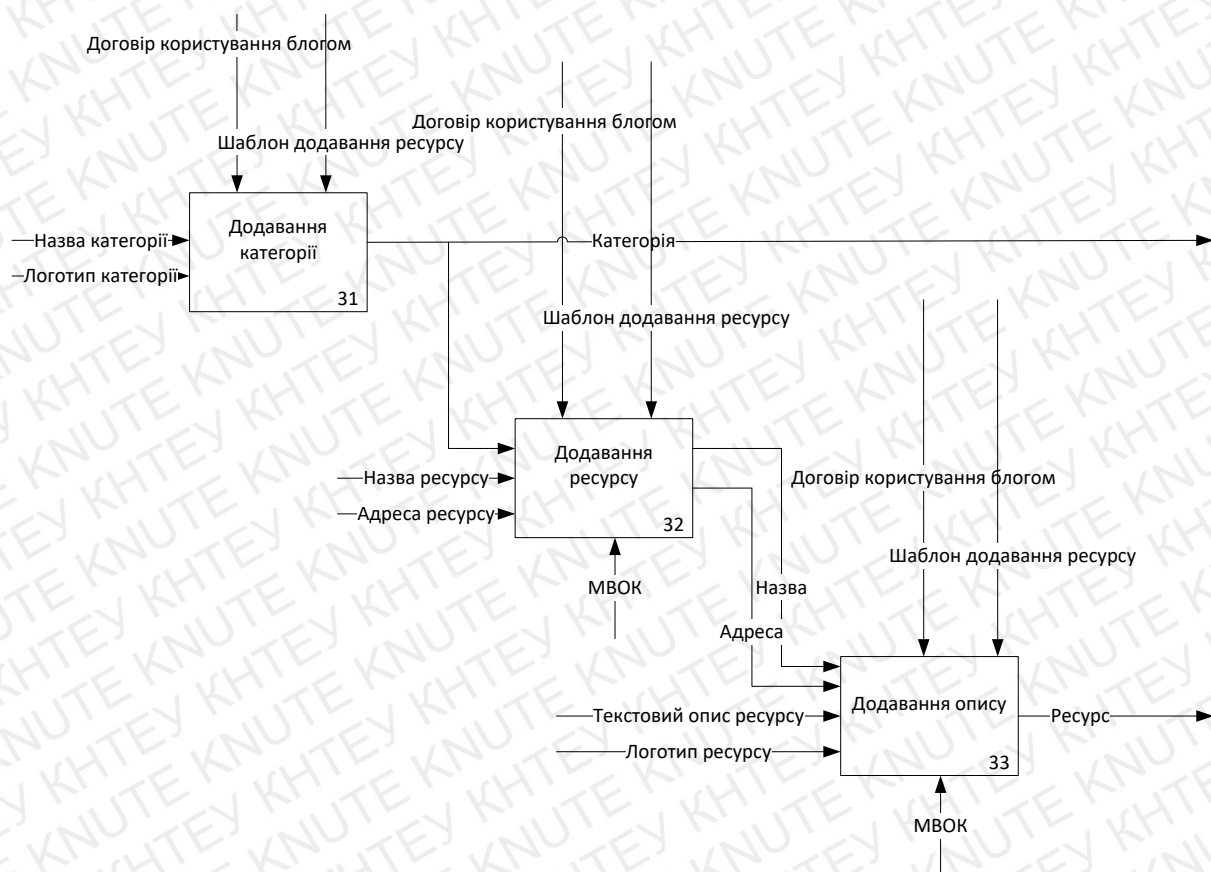


Рис. 2.3. Діаграма SADT 1-го рівня для процесу «Управління курсами»

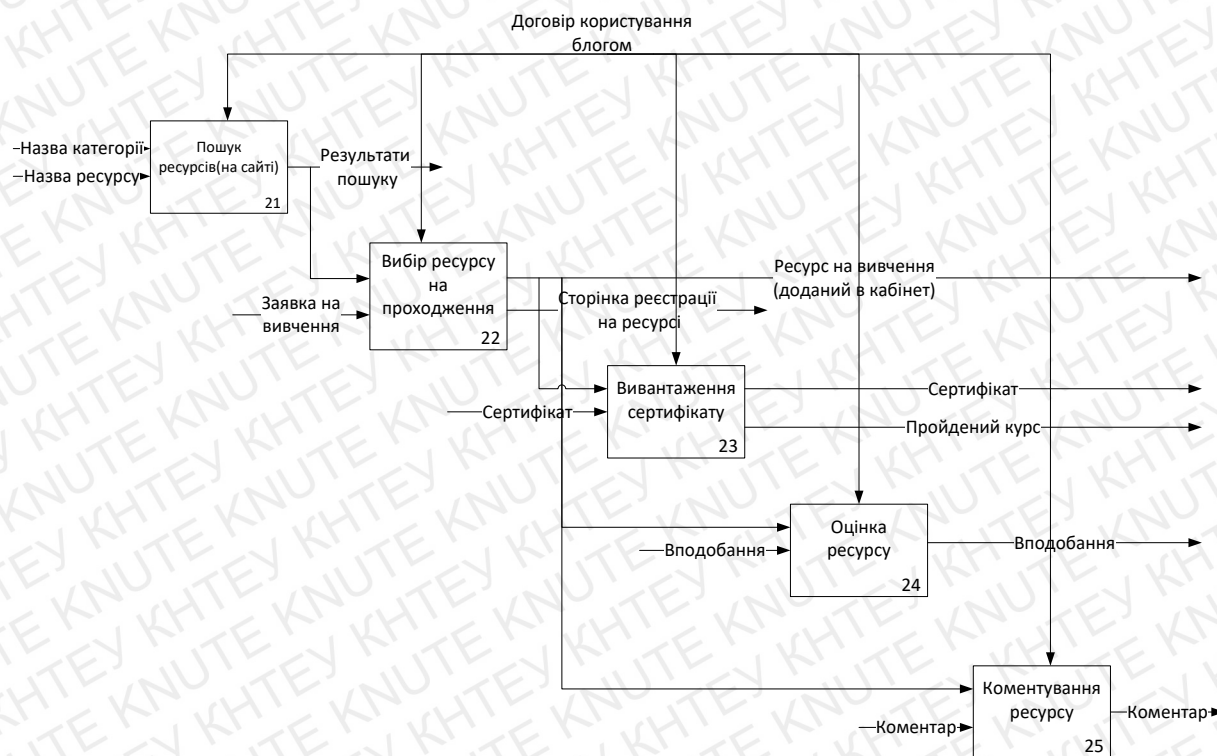


Рис. 2.4. Діаграма SADT 1-го рівня для процесу «Управління навчанням»

2.1.3. Діаграма DFD. Діаграми потоків даних (Data Flow Diagrams - DFD) є ієрархію функціональних процесів, пов'язаних потоками даних. Мета такого подання - продемонструвати, як кожен процес перетворить свої вхідні дані у вихідні, а також виявити відносини між цими процесами [9].

Для побудови DFD традиційно використовуються дві різні нотації, які відповідають методам Йордона-ДеМарко і Гейна-Серсона. Ці нотації незначно відрізняються один від одного графічним зображенням символів.

Основними компонентами діаграм потоків даних є:

- зовнішні сутності;
- системи і підсистеми;
- процеси;
- накопичувачі даних;
- потоки даних.

Зовнішня сутність є матеріальним об'єктом або фізична особа, що є джерелом або приймачем інформації, наприклад, замовники, персонал, постачальники, клієнти, склад.

Процес являє собою перетворення вхідних потоків даних у вихідні відповідно до певного алгоритму.

Накопичувач даних - це абстрактне пристрій для зберігання інформації, яку можна в будь-який момент помістити в накопичувач і через деякий час витягнути, причому способи приміщення і вилучення можуть бути будь-якими.

Потік даних визначає інформацію, передану через деякий з'єднання від джерела до приймача [10].

Діаграма DFD нульового рівня розроблюваної системи представлена в додатку А. Діаграми DFD першого рівня для бізнес-процесу «Управління курсами» представлена на рис. 2.5.

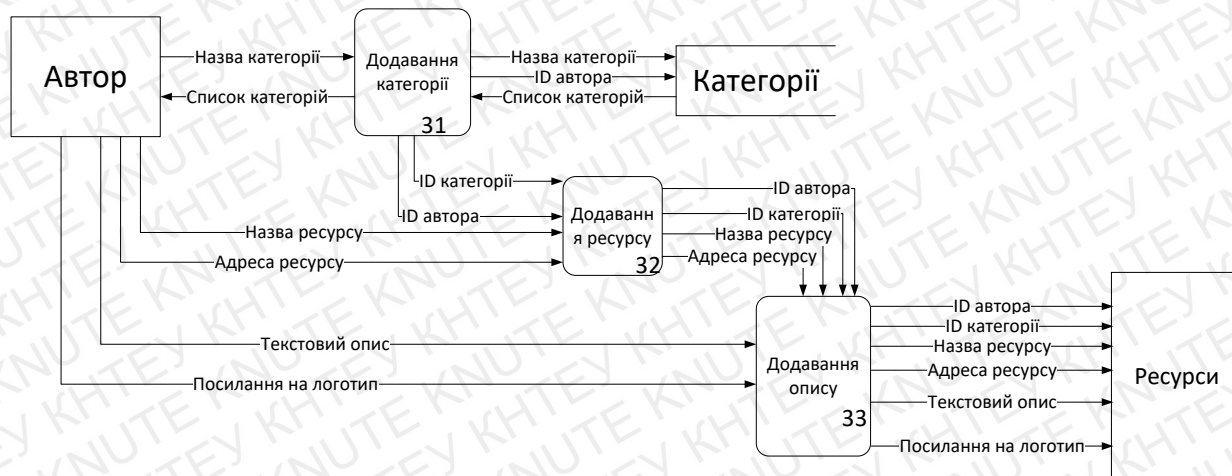


Рис. 2.5. Діаграма DFD 1-го рівня для процесу «Управління курсами»

2.2. Об'єктно-орієнтоване проектування

Опис системи у вигляді об'єктів більше відповідає змістовному опису предметної області. Наприклад, при використанні структурного підходу БД повинна задовольняти вимогам нормалізації, відповідно до яких дані по одному і тому ж об'єкту (сутності з реального світу) можуть зберігатися в кількох таблицях.

Сутності реального світу, як правило, володіють поведінкою, що в об'єктно-орієнтованому проектуванні відбивається за допомогою визначення методів класу. У структурному підході дані (атрибути) і алгоритми (методи) існують окремо один від одного.

Об'єднання атрибутів і методів в об'єкті (класі), а також інкапсуляція дозволяють домогтися більшої внутрішньої і меншої зовнішньої зв'язності між компонентами системи. Це полегшує вирішення проблем:

- адаптації системи до зміни існуючих або появи нових вимог;
- супроводу системи на різних стадіях життєвого циклу;
- повторного використання компонентів.

Об'єктно-орієнтований підхід дозволяє легше організувати паралельні обчислення, оскільки кожен об'єкт володіє власними значеннями характеристик (атрибутів) і поведінкою, за рахунок чого можна домогтися його автономної роботи.

2.2.1. Діаграма прецедентів. Діаграми прецедентів дозволяють візуалізувати поведінку системи, підсистеми або класу, щоб користувачі могли зрозуміти, як їх використовувати, а розробники - реалізувати відповідний елемент.

Прецеденти складаються з безлічі сценаріїв (послідовності кроків описують взаємодію між користувачем і системою), об'єднаних разом деякої спільною метою користувача. Складна система містить кілька десятків прецедентів, кожен з яких може розгортатися в кілька десятків сценаріїв. Для будь-якого прецеденту можна виділити основні сценарії, які описують найважливіші послідовності, і допоміжні, що описують альтернативні послідовності. У кожного прецеденту є передумова і післяумова.

Прецеденти бувають двох типів: **бізнес-прецеденти** (описують функціональність на верхньому рівні і призначена для замовника програмної системи) і **системні прецеденти** (описують функціональність на нижньому рівні, будуються для розробника програмної системи). У будь-якого прецеденту має бути ім'я, унікальне в рамках пакета [11].

Діаграма прецедентів представлена на плакаті 2.

2.2.2. Діаграма діяльності. Для моделювання процесу виконання операцій у мові UML використовують **діаграму діяльності**, яка зображується графом, вершинами якого є стани (дій і/або видів діяльності), а

дугами – переходи від одного стану (дій/виду діяльності) до іншого стану (дій/виду діяльності).

Головним напрямом використання діаграм діяльності є візуалізація особливостей реалізації операцій класів, коли необхідно зобразити алгоритми їхнього виконання. При цьому кожен стан дії може бути виконанням операції деякого класу, а вид діяльності – послідовністю таких дій.

На етапі моделювання прецедентів діаграми діяльності зображають потоки функцій керування, зазначають, які гілки процесу керування можуть виконуватися паралельно, і визначають альтернативні шляхи досягнення мети. Потоки керування можуть містити декілька прецедентів або протікати у рамках одного прецеденту.

У контексті мови UML вид діяльності є деякою сукупністю окремих елементарних обчислень. Щодо цього окремі елементарні обчислення можуть спричинювати до деякого результату або дії. На діаграмі діяльності відображається логіка або послідовність переходу від однієї діяльності до іншої, у цьому випадку увага фіксується на результаті діяльності, який може спричинити зміну стану системи або повернення деякого значення.

Елементи діаграми діяльності: стани видів діяльності, стани дій, переходи, точки прийняття рішення або галуження, вертикальна/горизонтальна смуги синхронізації і доріжки [12].

На рис. 2.6 і рис. 2.7 представлені діаграми діяльності для бізнес-процесів “Управління навчанням” і “Додавання ресурсу” відповідно.



Рис. 2.6. Діаграма діяльності для процесу «Управління навчанням»

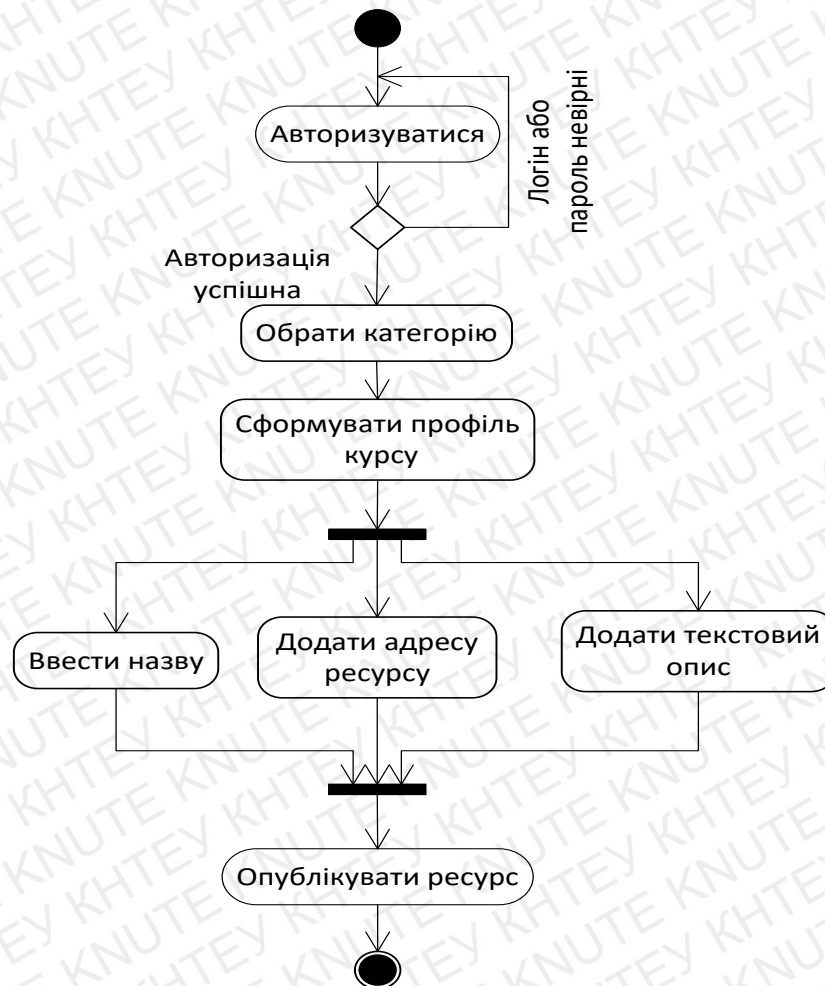


Рис. 2.7. Діаграма діяльності для процесу «Додавання курсу»

2.2.3. Діаграма послідовності. На діаграмі послідовності зображуються виключно ті об'єкти, які безпосередньо беруть участь у взаємодії і не показуються можливі статичні асоціації з іншими об'єктами. Для діаграми послідовності ключовим моментом є саме динаміка взаємодії об'єктів в часі. При цьому діаграма послідовності має як би два виміри. Одне - зліва направо у вигляді вертикальних ліній, кожна з яких зображує **лінію життя** окремого об'єкта, який бере участь у взаємодії. Графічно кожен об'єкт зображується прямокутником і розташовується у верхній частині своєї лінії життя. Усередині прямокутника записуються ім'я об'єкту і ім'я класу, розділені двокрапкою. При цьому вся запис підкреслюється, що є ознакою об'єкта, який, як відомо, являє собою екземпляр класу. Кожен об'єкт існує

тільки 1 раз від моменту створення до знищення. Деякі об'єкти вважаються вже створеними (показуються вгорі) [13].

Діаграма послідовності розроблюваної системи представлено на рис. 2.8.

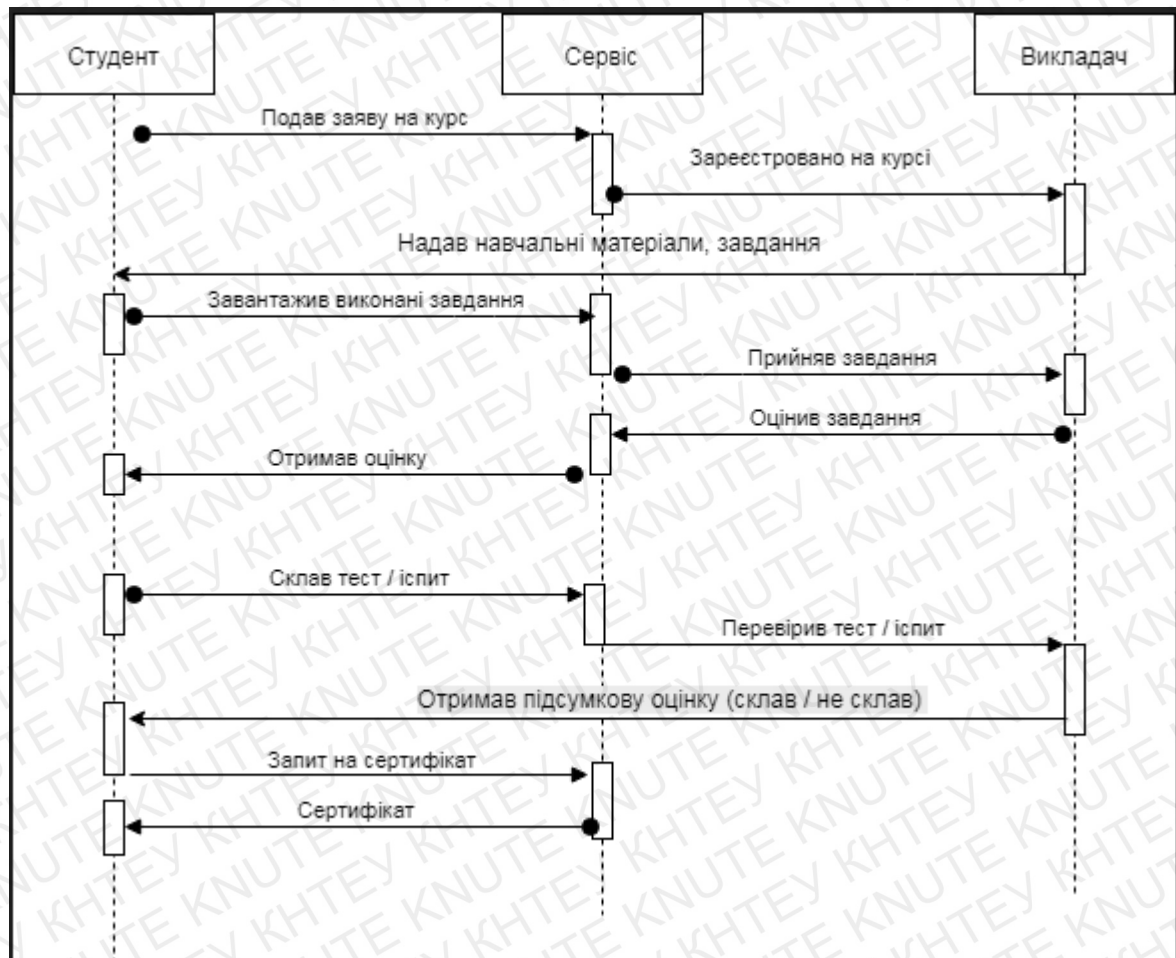


Рис. 2.8. Діаграма послідовності

2.4 Висновки до розділу 2

Для більш детального моделювання було використано мову UML – це стандартний інструмент для допомоги в розробці ПЗ, що використовується для візуалізації, специфікації, конструювання й документування артефактів ПЗ. UML підходить для моделювання будь-яких систем – від складних інформаційних систем до розподілених Web-додатків.

РОЗДІЛ 3.

ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМИ

Інформаційне забезпечення — це сукупність форм документів, нормативної бази і реалізованих рішень щодо обсягу, розміщення і форм організації інформації, яка циркулює в системі автоматизованого оброблення економічної інформації чи в інформаційній системі.

Основними принципами створення інформаційного забезпечення є: цілісність; достовірність; контроль; захист від несанкціонованого доступу; єдність і гнучкість; стандартизація та уніфікація; адаптивність [14].

3.1. Побудова логічної моделі даних

3.1.1. Загальні відомості про ER-Діаграму. Модель «сутність-зв'язок» (ER-модель) — модель даних, яка дозволяє описувати концептуальні схеми за допомогою узагальнених конструкцій блоків. ER-модель — це метамодель даних, тобто засіб опису моделей даних. Головними поняттями ER-моделі є сутність, зв'язок і атрибут. У діаграмах ER-моделі сутність подається у вигляді прямокутника, що містить ім'я сутності. При цьому ім'я сутності - це ім'я типу, а не деякого конкретного екземпляра цього типу. Для кращого розуміння ім'я сутності може супроводжуватися прикладами конкретних об'єктів цього типу.

Кожний екземпляр сутності повинен відрізнятися від будь-якого іншого екземпляра тієї ж сутності (ця вимога до повної міри аналогічна вимозі відсутності кортежів-дублікатів у реляційних таблицях).

Зв'язок - це асоціація, що графічно зображується та установлюється між двома сутностями. Ця асоціація завжди є бінарною і може існувати між двома різними сутностями або між сутністю і нею ж самою.

					КНТЕУ 121 06з-15.МР				
Зм.	№	№ докум	Підпис	Дата	«Розробка освітньої інформаційної системи “Evolution”» Інформаційне забезпечення системи	Стадія	Аркуш	Аркушів	
Зав. каф.		Криворучко О.В		19.10.20		РЗ	24	41	
Керівник		Жирова Т.О.		19.10.20		Факультет інформаційних технологій 2м курс, 6з група			
Гарант		Криворучко		19.10.20					
Розробив		Усик С.А.		19.10.20					

3.1.2. Побудова ER-діаграми. На плакаті 3 зображено логічну модель даних побудовану за допомогою програмного продукту Erwin Data Modeler.

При побудові даної моделі, були виділені такі ключові сутності:

- Категорія (Category) – описує інформацію про певну область знань. Основні атрибути: назва, опис та кількість курсів в категорії.
- Курс (Course) – описує інформацію про онлайн-курс. Основні атрибути: назва, посилання на курс, логотип, дата публікації, опис, автор і унікальний ідентифікатор категорії.
- Персона (Person) – описує спільну інформацію всіх типів користувачів системи. Основні атрибути: ім'я, прізвище, електронна пошта, логін, пароль і дискримінатор – тип персони.
- Студент (Student) – описує інформацію про тип користувачів, який може додавати курси на проходження, коментувати, оцінювати і вивантажувати сертифікати до них. Всі атрибути наслідуються від сутності Персона.
- Автор (Author) – описує інформацію про тип користувачів, який може публікувати, редагувати та видаляти онлайн-курси. Всі атрибути наслідуються від сутності Персона.
- Модератор (Moderator) – описує інформацію про тип користувачів, який перевіряє заявки на авторство, на їх основі приймає рішення про присвоєння статусу автора, а також переглядає звітну інформацію по категоріям, онлайн-курсам і авторам. Всі атрибути наслідуються від сутності Персона.
- Коментар (Comment) – описує інформацію про відгуки, що залишають студенти до певного курсу. Основні атрибути: унікальний ідентифікатор онлайн-курсу, унікальний ідентифікатор користувача, дата коментування і текст коментаря.
- Кабінет (Cabinet) – описує інформацію про персональний кабінет, куди додаються усі онлайн-курси, на які підписується Студент. Основні

атрибути: унікальний ідентифікатор онлайн-курсу, унікальний ідентифікатор користувача, дата додавання і статус проходження.

- Заявки (Cabinet) – описує інформацію про заявку, яку подає Студент у разі, якщо бажає отримати статус автора. Основні атрибути: унікальний ідентифікатор користувача, дата подання, статус перевірки і відповідь Модератора.

3.2. Вибір інструментарію для створення бази даних та роботи з нею

3.2.1. Вибір системи управління базою даних. SQL Server — вільна об'єктно-орієнтована система управління базами даних(СУБД).

Оскільки дана СУБД є об'єктно-орієнтованою, тому вона має певні переваги:

- Наявність механізму успадкування;
- Відповідність моделі даних в додатку і в базі даних(БД);
- Зручність в відображенні складних об'єктів;
- Всі об'єкти на рівні джерела даних строго типізовані;
- Велика кількість типів даних і можливість визначення нових типів даних.

3.3. Фізична модель бази даних

Фізична модель бази даних представлена рис. 3.1.

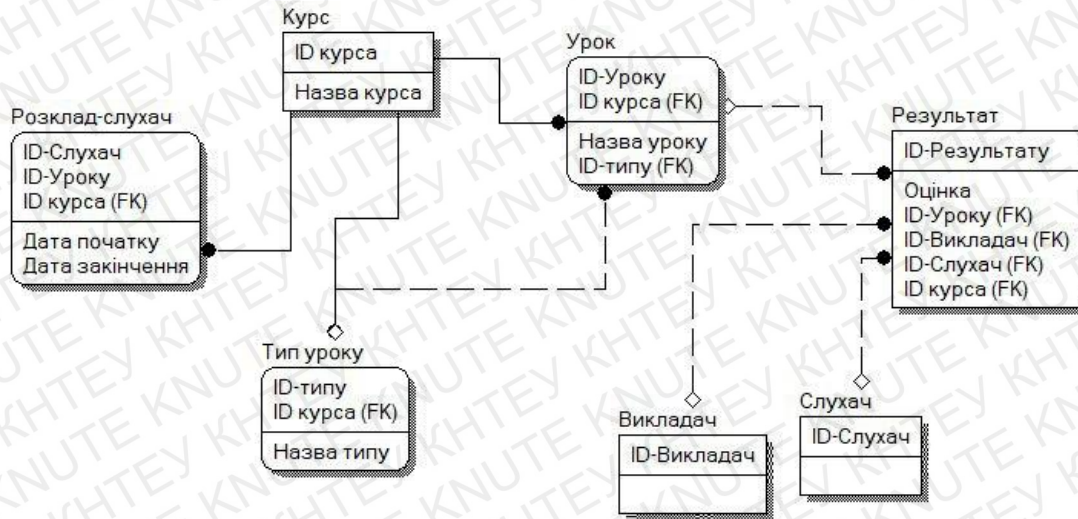


Рис. 3.1. Фізична модель бази даних

3.4 Висновки до розділу 3

В даному розділі було побудовано логічну модель даних, та створено фізичну модель даних. Моделі були побудовані для опису потреб інформації чи типу інформації, що зберігатиметься в базі даних. Та описано вибір системи управління базою даних.

РОЗДІЛ 4

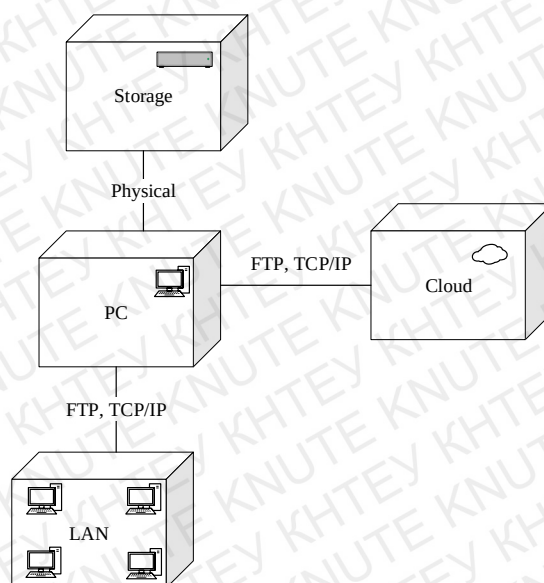
РЕКОМЕНДАЦІЇ ЩОДО ВПРОВАДЖЕННЯ ТА ЕКСПЛУАТАЦІЇ СИСТЕМИ

4.1. Впровадження програмного забезпечення

4.1.1. Діаграма розгортання. Діаграма розгортання відображає фізичні взаємозв'язку між програмними і апаратними компонентами системи. Вона є хорошим засобом для того, щоб показати маршрути переміщення об'єктів і компонентів в розподіленій системі.

Кожен вузол на діаграмі розгортання являє собою певний **тип обчислювального пристрою**, в більшості випадків - частина апаратури.

Компоненти на діаграмі розгортання становлять фізичні модулі програмного коду. Як правило, вони в точності відповідають компонентам на діаграмі компонентів. Таким чином, діаграма розгортання відображає виконання кожного компонента в системі.



Діаграма розгортання представлена на плакаті 4.

					<i>КНТЕУ 121 063-15.МР</i>			
Зм.	№	№ докум	Підпис	Дата	«Розробка освітньої інформаційної системи "Evolution"» Рекомендації щодо впровадження та експлуатації системи			
Зав. каф.		Криворучко О.В.		19.10.20				
Керівник		Жирова Т.О.		19.10.20				
Гарант		Криворучко		19.10.20				
Розробив		Усик С.А.		19.10.20				
					Факультет інформаційних технологій 2м курс, 63 група			
					Стадія	Аркуш	Аркушів	
					Р4	28	41	

Визначення вимог до системи

Функціональні вимоги визначають, що повинна робити система, вони визначаються в постановці завдання. Не функціональні вимоги визначають, як система повинна працювати в нормальних умовах, тому для даної програми визначені такі не функціональні вимоги:

1. Реєстрація студента повинна займати не більше як 10хв часу
2. У разі виникнення збоїв система має повідомляти про помилку і рішення даної помилки, якщо якісь дані були вказані не вірно.
3. Система не повинна призводити до помилок, що завдадуть шкоди вашому пристрою.
4. Система повинна бути для користувача зрозумілою, тобто повідомляти на кожному кроці, що необхідно користувачу або що відбувається в поточний момент.
5. Текст програми повинен бути структурованим і легко читатися

4.1.3. Вимоги до апаратного та програмного забезпечення користувача. Для користування системою користувачеві необхідно мати персональний комп'ютер, планшет або мобільний пристрій з браузером і доступом до мережі Інтернет. Єдиною вимогою до апаратного і програмного забезпечення клієнтських пристроїв є підтримка оновлених та стабільних версій будь-яких браузерів.

4.1.4. Вимоги до апаратного та програмного забезпечення серверу.

Сервер має відповідати таким мінімальним вимогам:

- **Процесор:** "Xeon E5-2620V2" 1 ядро;
- **ОЗП:** "DDR3 ECC Reg" 1 ГБ;
- **Жорсткий диск:** "SAS RAID 10" 16 ГБ;
- **Порт:** 25 Мбіт/с;
- **IP-адреса:** IPv4/IPv6.
- **Операційна система:**

Windows/Linux/Mac OS;

4.1.5 Інсталяційний пакет. Інсталяційний пакет програмного забезпечення складається з:

- **Інтерпретатор:** Python 3.x;
- **Менеджер пакетів:** Python Index Packages(PIP)

4.2. Програмне забезпечення

4.2.1. Архітектура програмного забезпечення. Організаційна структура інформаційної системи представлена у вигляді діаграми пакетів.

Діаграми пакетів уніфікованої мови моделювання(UML) відображають залежності між пакетами, з яких і складається модель.

Пакет(package) – елемент моделі, який використовують для групування інших елементів моделі. Елементи моделі, які входять у склад певного пакету, називаються членами пакету .

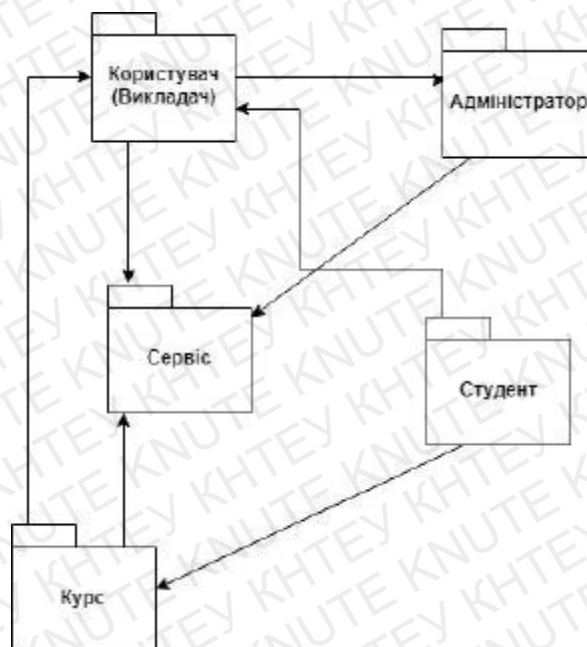


Рис. 4.1. Діаграма пакетів

4.2.2. Організаційна структура. Програмний сервіс для обслуговування кабінету викладача системи online навчання

Веб-застосунок— розподілений застосунок, в якому клієнтом виступає браузер, а сервером — веб-сервер. Браузер може бути реалізацією так званих тонких клієнтів — логіка застосунку зосереджується на сервері, а функція браузера полягає переважно у відображенні інформації, завантаженої мережею з сервера, і передачі назад даних користувача. Однією з переваг такого підходу є той факт, що клієнти не залежать від конкретної операційної системи користувача, тому веб-застосунки є міжплатформовими сервісами. Унаслідок цієї універсальності і відносної простоти розробки веб-застосунки стали широко популярними в кінці 1990-х — початку 2000-х років [17].

4.2.3. Вибір інструментарію для розробки ПЗ. Для написання серверної частини веб-застосунку було обрано скрипкову мову Python. Код серверної частини представлено в додатку В.

Клієнтський частина написана на мові JavaScript. Графічний інтерфейс побудований за допомогою HTML5 і CSS3.

4.2.2. Інтерфейс роботи з БД. Для роботи з базою даних використовується модуль SQL Server.

SQL Server – комерційна система керування базами даних, що розповсюджується корпорацією Microsoft. Мова, що використовується для запитів — Transact SQL, створена спільно Microsoft та Sybase. Transact-SQL є реалізацією стандарту ANSI / ISO щодо структурованої мови запитів SQL із розширеннями. Використовується як для невеликих і середніх за розміром баз даних, так і для великих баз даних масштабу підприємства. Багато років вдало конкурує з іншими системами керування базами даних.

4.2.3. Мова програмування JavaScript. JavaScript — це особлива мова програмування, який базується на об'єктному представленні браузера. Він потрібен для того, щоб надати сайту більше інтерактивності в порівнянні зі звичайним статичним HTML-документом.

Призначення JavaScript:

- За допомогою JavaScript створюються динамічні документи HTML
- JavaScript пов'язує воедино всі будівельні блоки програми, це як би засіб побудови фундаменту.
- JavaScript здійснює перевірку полів форм HTML до того, як вони передалися на сервер.
- Управління програмою на даній мові програмування йде через локальний введення інформації.

Користувач має можливість бачити в окремих вікнах повідомлення-застереження, які виводяться за допомогою JavaScript [19].

4.2.4. Мова розмітки HTML5. HTML5 - це інструмент для упорядкування Web-контенту. Він призначений для спрощення Web-проектування і Web-розробки за рахунок мови розмітки, що забезпечує стандартизований і інтуїтивно зрозумілий інтерфейс. HTML5 надає розробнику кошти для секціонування і структуризації Web-сторінок, а також дозволяє створювати відокремлені компоненти. Мова HTML5 реалізує підхід до проектування Web-сайтів, заснований на відображенні інформації, оскільки він втілює саму суть відображення інформації - поділ і маркування інформації для спрощення її використання і розуміння. Саме в цьому полягає величезна семантична та естетична цінність HTML5. HTML5 надає дизайнерам і розробникам всіх рівнів можливості для надання в публічний

					<i>КНТЕУ 121 063-15.МР</i>	32
Зм.	Арк.	№ докум.	Підпис	Дата		

доступ буквально будь-якого контенту - від простих текстів до мультимедійно насичених інтерактивних матеріалів.

HTML5 надає ефективні інструменти для управління даними, для малювання, для відтворення відео- та аудіоконтенту. HTML5 полегшує розробку крос-браузерних Web-додатків, а також додатків для мобільних пристроїв. HTML5 відноситься до числа технологій, які стимулюють розвиток мобільних сервісів на основі хмарних обчислень. Крім того, HTML5 сприяє підвищенню гнучкості - завдяки можливості створення вражаючих і інтерактивних Web-сайтів. І, нарешті, HTML5 пропонує нові теги і вдосконалення, в числі яких такі: елегантна структура, органи управління формами, API-інтерфейси, мультимедійні функції, підтримка баз даних, істотно збільшена швидкість обробки [20].

4.2.5. Таблиця каскадних стилів CSS3. CSS (Cascading Style Sheets) з англійської перекладається як каскадні таблиці стилів, - це мова, яка використовується як засіб оформлення веб-сторінок, а саме для роботи з шрифтами, кольорами, полями, таблицями, картинками, розташуванням елементів.

Основною метою, яка послужила для розробки мови CSS, було розділення розмітки вмісту (написаного на HTML або іншій мові програмування) та подання документа (написаного на CSS). Результат цього поділу допомагає збільшити доступність документа, надати велику гнучкість, а також зменшити складність і повторюваність в структурному вмісті, створює зручне опис дизайну та стилістики web-сторінки і її вмісту. Ще CSS дає можливість надати один документ в різних стилях або методах виведення.

4.3. Функціональне тестування

Для початку користування системою, користувачу необхідно перейти на головну сторінку ресурсу. У верхній частині екрану доступні посилання для переходу на форму реєстрації і авторизації, а також різноманітні вкладки. Для пошуку курсу або викладача необхідно натиснути на вверху для переходу. Вигляд домашньої сторінки представлений на рис. 4.2.

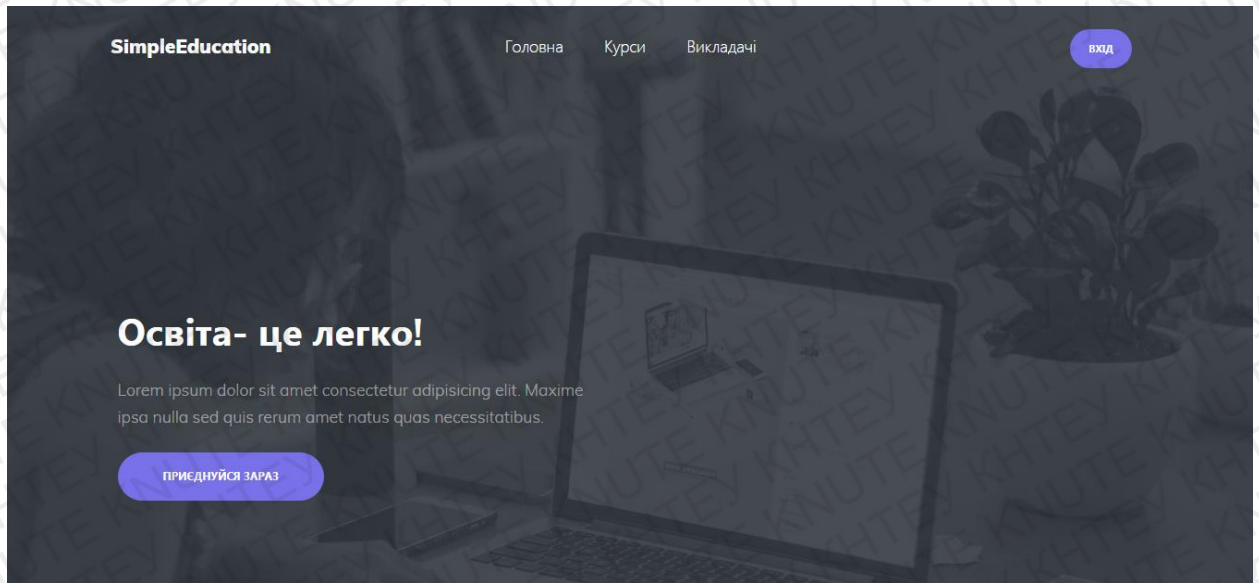


Рис. 4.2. Головна сторінка веб-застосунку

Перехід на форму реєстрації представлений на рис. 4.3.

Рис. 4.3. Сторінка реєстрації

Після реєстрації користувач повинен пройти авторизацію.
Авторизацію представлено на рис. 4.4.

Рис. 4.4. Авторизація

Якщо ви зареєструвались як викладач вас після цього ви заходите на особистий кабінет викладача (рис. 4.5)

Рис. 4.5. Кабінет викладача

При натисненні «Створити курс» ви переходите до форми створення курсу . Вигляд представлений на рис. 4.6.

Рис. 4.6. Створення курсу

У розділі «Завантажені роботи» відображаються усі роботи які завантажив студент для перевірки . Вигляд представлений на рис. 4.7.

Рис.4.7. Завантажені роботи

У розділі «Список завдань» Викладач завантажує лекції завдання. Вигляд роботи представлено на рис. 4.8.

Рис.4.8. Список завдань

4.4 Висновки до розділу 4

Даний розділ описує апаратне, програмне забезпечення, вимоги визначають, як система повинна працювати в нормальних умовах. Було вибрано інструментарій для розробки ПЗ. Також зображено функціонал роботи системи із всіма етапами користування , для полегшення користування нею.

ВИСНОВКИ ТА ПРОПОЗИЦІЇ

У дипломному проекті було розглянуто етапи створення програмного продукту системи онлайн навчання. Так як на сьогоднішній день дистанційне навчання набирає популярність, було створено систему у якій викладач може вільно розмістити курси та навчальні матеріали. Також можна зареєструватися як учень, і шукати цікаві собі курси, при цьому підтримувати зв'язок з викладачем.

Після проходження курсів буде отримано сертифікат про його проходження, а у випадку високих результатів студента – про його успішне завершення. Такі сертифікати є чудовим доказом володіння навичкою, багато-хто навіть включає сертифікати з освітніх платформ у свої резюме.

Для автора розміщення свого курсу на платформі також може бути дуже вигідним. Наприклад, платформа сама займається просування курсу аж до його продажу, а автор просто отримує гонорар із закритої угоди.

Звісно ж, платформа бере певний відсоток, але і бере на себе усі справи з продажу, не залучаючи автора у цей процес. Таким чином, автор має популяризацію свого контенту та пасивний прибуток одночасно.

Також автор самостійно оцінює вартість свого курсу, проте це треба робити, спираючись на певні дані, а не називати першу ліпшу цифру, інакше платформі може бути важко продати курс за суму, що не відповідає його наповненню.

Розроблена інформаційна система може використовуватися у вільному доступі для усіх користувачів мережі Інтернет.

					<i>КНТЕУ 121 063-15.МР</i>			
Зм.	№	№ докум	Підпис	Дата	<i>«Розробка освітньої інформаційної системи “Evolution”»</i>	Стадія	Аркуш	Аркушів
Зав. каф.	Криворучко О.В.		30.10.20	ВП		38	41	
Керівник	Жирова Т.О.		30.10.20	Факультет інформаційних технологій 2м курс, 6з група				
Гарант	Криворучко О.В.		30.10.20					
Розробив	Усик С.А.		30.10.20					
					<i>Висновки та пропозиції</i>			

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Гаврилюк Н. Неформальне професійне навчання осіб [Електронний ресурс] // Сайт Дніпропетровського центру професійно-технічної освіти державної служби зайнятості. – Режим доступу до сайту: <http://cpto.dp.ua/index.php/home/280-2014-02-17-15-03-30>

2. Все про масові відкриті онлайн-курси [Електронний ресурс] // сайт “Освіта.UA”. – Режим доступу до сайту: <http://osvita.ua/news/topics/50757/>

3. Примаченко І. Літній путівник світом безкоштовного онлайн-навчання [Електронний ресурс] // Інформаційний портал “Рух онлайн-навчання в Україні”. – Режим доступу до порталу: <http://mooc.in.ua/articles/litnij-putivnyk-svitom-bezkoshtovnogo-onlajn-navchannya-2>

4. Арлов Д. Що таке UML? [Електронний ресурс] : підручн. / Д. Арлов, А. Нейштадт // UML 2 і Уніфікований процес – 2007. – С. 624. – Режим доступу до підручн. : http://cmcstuff.esyr.org/vmkbotvar15/курс/Семестр/ООАП/Книги/Arlou_i_Neyshtadt_UML_i_UP.pdf

5. Денісова О. Автоматизація об’єктно-орієнтованого проектування інформаційних систем [Електронний ресурс] // Бібліотека підручників та статей “Posibniki”. – Режим доступу: <http://posibniki.com.ua/post-avtomatizaciya-obektno-orientovanogo-proektuvannya-informaciynih-sistem>

6. Інформаційні системи і технології на підприємствах [Електронний ресурс] // сайт “Навчальні ресурси СумДУ”. – Режим доступу: http://elearning.sumdu.edu.ua/free_content/lectured:de1c9452f2a161439391120eef364dd8ce4d8e5e/20160217112601/200847/index.html

7. Палмер С. Функціонально-орієнтоване відслідкування [Електронний ресурс] : навч. посіб. / С. Палмер, Р. Фелсінг, М. Джон //

					КНТЕУ 121 063-15.МР			
Зм.	№	№ докум	Підпис	Дата	«Розробка освітньої інформаційної системи “Evolution”» Список використаних джерел	Стадія	Аркуш	Аркушів
Зав. каф.		Криворучко О.В.		24.01.20		СВД	39	41
Керівник		Жирова Т.О.		24.01.20				
Гарант		Криворучко О.В.		24.01.20		Факультет інформаційних технологій		
Розробив		Усик С.А.		24.01.20		2м курс, 63 група		

Практичний посібник по функціонально-орієнтовній розробці – 2002. – С. 304. – Режим доступу до навч. посіб. : <https://books.google.com.ua/books?id=3MU9WAQ9MIMC&pg=PA22&lpg=PA22&dq=fdd>

8. Нотація IDEF0 // сайт “Нотатки управлінця” [Електронний ресурс]. – Режим доступу: http://www.businessstudio.com.ua/bp/notation_idef0.php

9. Діаграма потоків даних (DFD) [Електронний ресурс] // навчальний портал “Е-educ”. – Режим доступу: <http://e-educ.ru/bd14.html>

10. Калянов Г. Складові діаграми потоків даних [Електронний ресурс] : підруч. / Г. Калянов, А. Калашян // Структурні моделі бізнеса: DFD технології – 2009. – С. 256. – Режим доступу до підруч. : <http://www.ozon.ru/context/detail/id/1512370/>

11. Діаграма прецедентів. Роль прецедентів при розробці ПС. Види прецедентів і відносини між ними. Правила опису [Електронний ресурс] // сайт “Учбові Матеріали для студентів і школярів України”. – Режим доступу: <http://um.co.ua/8/8-2/8-213194.html>

12. Діаграми діяльності [Електронний ресурс] // сайт “Студопедія”. – Режим доступу: http://studopedia.su/16_5481_omps-diagrami-diyalnosti.html

13. Діаграма послідовності і правила її побудови. Види повідомлень і правила їх позначень [Електронний ресурс] // сайт “Учбові Матеріали для студентів і школярів України”. – Режим доступу: <http://um.co.ua/9/9-2/9-29950.html>

14. Інформаційні системи і технології в статистиці [Електронний ресурс] // сайт “Бібліотека економіста”. – Режим доступу: <http://library.if.ua/book/80/5658.html>

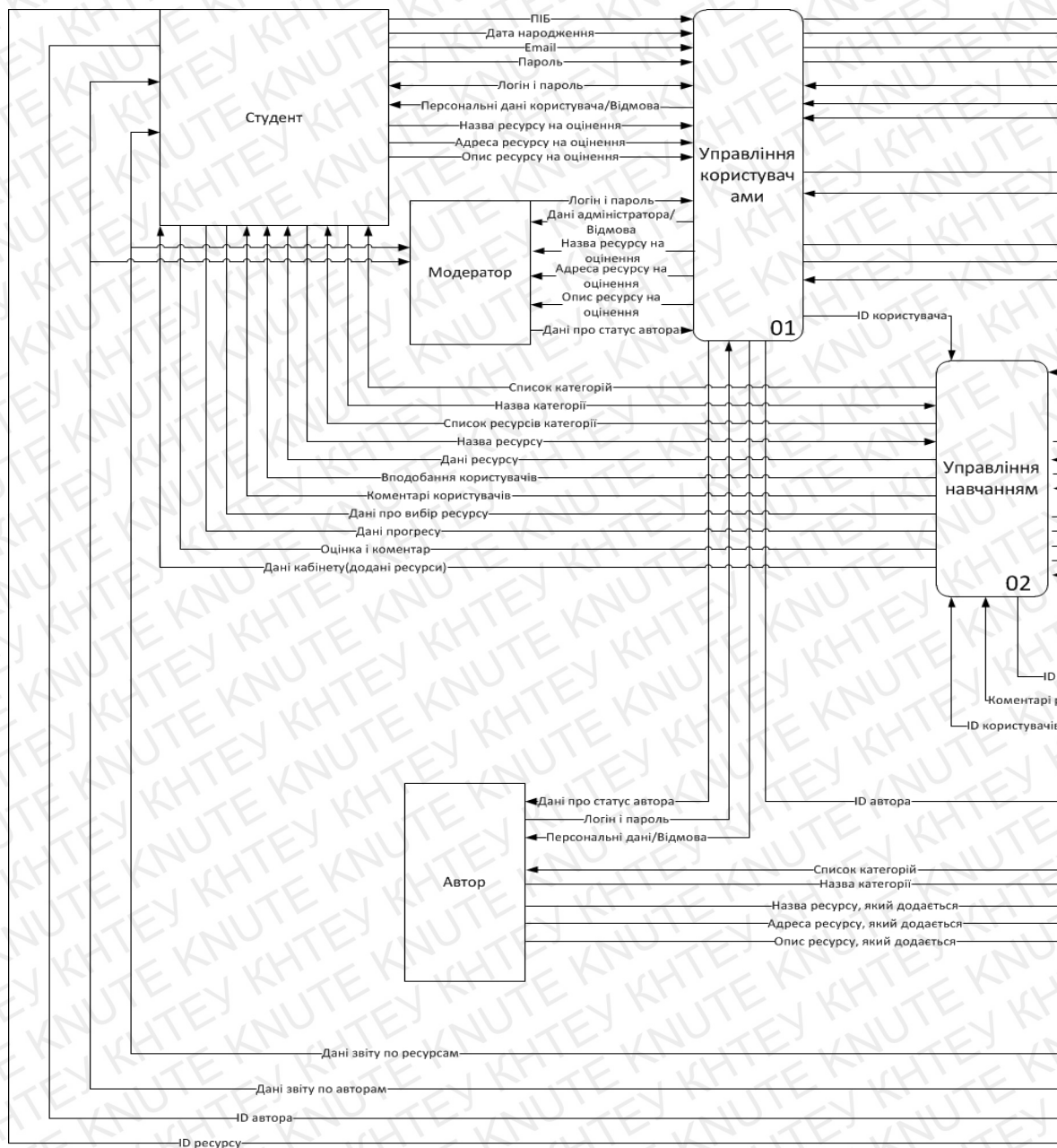
15. Діаграми компонентів і розміщення [Електронний ресурс] // сайт “Учбові Матеріали для студентів і школярів України”. – Режим доступу: <http://um.co.ua/8/8-13/8-132563.html>

16. Уніфікована мова моделювання [Електронний ресурс] // сайт бібліотеки МГПУ ім. Богдана Хмельницького. – Режим доступу: <http://um.co.ua/8/8-13/8-132563.html>
17. Варченко-Троценко Л. Веб-застосунок [Електронний ресурс] // Вікі Київського університету імені Бориса Грінченка // Режим доступу: <http://wiki.kubg.edu.ua/Веб-застосунок>
18. Косенко Є. Мова програмування Python [Електронний ресурс] // Сайт “Енциклопедія мов програмування” // Режим доступу: <http://progopedia.ru/language/python/>
19. Побода В. Веб-розробка [Електронний ресурс] // Блог Віталія Пободи / Режим доступу: <http://www.vitaliypodoba.com/2015/06/webdev-howto/>
20. Уолкер Г. Основи HTML: Частина 1 [Електронний ресурс] // сайт “developerWorks” / Режим доступу: <https://www.ibm.com/developerworks/ru/library/wa-html5fundamental>

Додатки

Додаток А

ДІАГРАМА ПОТОКІВ ДАНИХ



СТВОРЕННЯ ОБ'ЄКТІВ БАЗИ ДАНИХ

```
CREATE TABLE `courses` (  
  `id` int(10) NOT NULL,  
  `title` varchar(255) NOT NULL,  
  `alias` varchar(255) NOT NULL,  
  `description` varchar(255) NOT NULL,  
  `teacher_id` int(30) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

Структура таблиці `lections`

```
CREATE TABLE `lections` (  
  `id` int(11) NOT NULL,  
  `title` varchar(255) NOT NULL,  
  `file` varchar(255) NOT NULL,  
  `course_id` int(30) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

-- Структура таблиці `tasks`

```
CREATE TABLE `tasks` (  
  `id` int(11) NOT NULL,
```



```
`title` varchar(255) NOT NULL,  
`file` varchar(255) NOT NULL,  
`user_id` int(10) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

Структура таблиці `user`

```
CREATE TABLE `user` (  
  `id` int(10) NOT NULL,  
  `firstname` varchar(255) NOT NULL,  
  `lastname` varchar(255) NOT NULL,  
  `email` varchar(255) NOT NULL,  
  `password` varchar(255) NOT NULL,  
  `role` enum('teacher','student') NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

- Структура таблиці `works`

```
CREATE TABLE `works` (  
  `id` int(11) NOT NULL,  
  `title` varchar(255) NOT NULL,  
  `file` varchar(255) NOT NULL,  
  `course_id` int(10) NOT NULL,  
  `user_id` int(10) NOT NULL  
)ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

```

-- Процедура для додавання нового курсу
--
CREATE OR REPLACE FUNCTION add_course(name
VARCHAR(30), course_url VARCHAR(100), logotype VARCHAR(100),
describe TEXT, pub_date DATE, avg_rate INTEGER, video_url
VARCHAR(200), category_id INTEGER)
RETURNS void AS $$
BEGIN
INSERT INTO course VALUES (name, course_url, logotype, describe,
pub_date, avg_rate, video_url, category_id);
END;
$$ LANGUAGE plpgsql;
--
-- Процедура для додавання курсу в персональний кабінет
--
CREATE OR REPLACE FUNCTION join_course(student_id
INTEGER, course_id INTEGER, join_date DATE, progress_status
VARCHAR(50))
RETURNS void AS $$
BEGIN
INSERT INTO cabinet VALUES (student_id, course_id, join_date,
progress_status);
END;
$$ LANGUAGE plpgsql;
--
-- Процедура для додавання коментаря до курсу
--
CREATE OR REPLACE FUNCTION join_course(person_id
INTEGER, course_id INTEGER, comment_date DATE, rate INTEGER, text
VARCHAR(50))
RETURNS void AS $$
BEGIN
INSERT INTO cabinet VALUES (student_id, course_id,
comment_date, rate, comment_text);
END;
$$ LANGUAGE plpgsql;

```

```

-- Тригер видалення курсів при видаленні категорії
--
CREATE OR REPLACE FUNCTION del_courses() RETURNS trigger
AS
$$
BEGIN
if (SELECT COUNT(*) FROM cabinet WHERE category_id=OLD.id)>0
    THEN DELETE FROM course WHERE category_id=OLD.id;
end if;
    return OLD;
END;
$$ LANGUAGE 'plpgsql';

CREATE TRIGGER delete_courses
BEFORE DELETE
ON category
FOR EACH ROW
EXECUTE PROCEDURE del_courses();
--
-- Тригер видалення курсів з кабінету при видаленні профілю
користувача
--
CREATE OR REPLACE FUNCTION del_cabinet() RETURNS trigger AS
$$
BEGIN
if (SELECT COUNT(*) FROM cabinet WHERE student_id=OLD.id)>0
    THEN DELETE FROM cabinet WHERE student_id=OLD.id;
end if;
    return OLD;
END;
$$ LANGUAGE 'plpgsql';

CREATE TRIGGER delete_cabinet
BEFORE DELETE
ON person
FOR EACH ROW
EXECUTE PROCEDURE del_cabinet();
COMMIT;

```


КОД ПРОГРАМИ

```
from django.shortcuts import render, get_object_or_404, redirect
from django.views import View
from django.contrib import messages
from django.db.models import Q
from django.core.paginator import Paginator, EmptyPage, PageNotAnInteger
from django.contrib.contenttypes.models import ContentType
from django.utils.decorators import method_decorator
from django.contrib.auth.mixins import UserPassesTestMixin
from django.contrib.auth.decorators import login_required
from django.http import HttpResponseRedirect

from comments.forms import CommentForm
from comments.models import Comment
from cabinets.forms import SubscribeForm
from .models import Category, Course
from .forms import CourseForm, CategoryForm

class AuthorOrStaffMixin(UserPassesTestMixin):
    def test_func(self):
        author_group = Group.objects.raw("SELECT * FROM authors WHERE
name='authors'")
        if self.request.user.is_staff or self.request.user.is_superuser:
            return True
        elif author_group in self.request.user.groups.all():
            return True
        else:
            return False

class IndexView(View):
    def get(self, *args, **kwargs):
        category_list = Category.objects.raw("SELECT * FROM categories")
        context = {
            "category_list": category_list,
        }

        return render(self.request, 'blog/index.html', context)

class CategoryView(View):
    def get(self, *args, **kwargs):
        category_list = Category.objects.raw("SELECT * FROM categories")
        category_obj = Category.objects.raw("SELETC * FROM categories
slug=kwargs['slug']")
```

```

        course_list = Course.objects.filter("SELECT * FROM courses WHERE
category_id=category_obj.id")
        paginator = Paginator(course_list, 10) # Show 10 contacts per page
        page_request_var = "page"
        page = self.request.GET.get(page_request_var)

    try:
        course_list = paginator.page(page)
    except PageNotAnInteger:
        # If page is not an integer, deliver first page.
        course_list = paginator.page(1)
    except EmptyPage:
        # If page is out of range (e.g. 9999), deliver last page of results.
        course_list = paginator.page(paginator.num_pages)

    context = {
        "category": category_obj,
        "category_list": category_list,
        "course_list": course_list,
        "page_request_var": page_request_var
    }

    return render(self.request, 'blog/category.html', context)

@method_decorator(login_required, name='post')
class CourseDetail(View):
    template_name = "blog/course.html"
    def get(self, *args, **kwargs):
        category_list = Category.objects.raw("SELECT * FROM categories")
        course_obj = Course.objects.raw("SELECT * FROM courses WHERE
slug=kwargs['slug']")
        comment_list = Comment.objects.raw("SELECT * FROM comments WHERE
course_id=course_obj.id")

        initial_data = {
            "content_type": course_obj.get_content_type,
            "object_id": course_obj.id,
        }

        comment_form = CommentForm(initial=initial_data)
        subscribe_form = SubscribeForm(initial=initial_data)

        is_subscribed = False
        if self.request.user.is_authenticated():
            is_subscribed = Course.is_subscribed(course_obj, self.request.user)

        is_author = False
        if self.request.user.is_authenticated() and self.request.user == course_obj.author:
            is_author = True

        context = {

```

```

        "category_list": category_list,
        "course": course_obj,
        "comments": comments,

        "comment_form": comment_form,
        "subscribe_form": subscribe_form,
        "is_subscribed": is_subscribed,
        "is_author": is_author,
    }

    return render(self.request, self.template_name, context)

def post(self, *args, **kwargs):
    course_obj = Course.objects.raw("SELECT * FROM course WHERE
slug=kwargs['slug']")
    comment_form = CommentForm(self.request.POST)
    if comment_form.is_valid():
        c_type = comment_form.cleaned_data.get("content_type")
        content_type = ContentType.objects.get(model=c_type)
        obj_id = comment_form.cleaned_data.get("object_id")
        text = comment_form.cleaned_data.get("text")
        parent_obj = None
        try:
            parent_id = int(self.request.POST.get('parent_id'))
        except KeyError:
            parent_id = None
        except TypeError:
            parent_id = None

        if parent_id:
            parent_qs = Comment.objects.filter(id=parent_id)
            if parent_qs.exists() and parent_qs.count() == 1:
                parent_obj = parent_qs.first()

            new_obj = Comment.objects.raw("INSERT INTO comments (user_id,
object_id, text, parent_id) VALUES
(self.request.user, obj_id, text, parent_obj)")
        )
        if created:
            return redirect(new_obj.content_object.get_absolute_url())

    return redirect(course_obj.get_absolute_url())

class CourseCreate(AuthorOrStaffMixin, View):
    template_name = "blog/course_form.html"

    def get(self, *args, **kwargs):
        category_list = Category.objects.raw("SELECT * FROM categories")

```



```

course_form = CourseForm()
context = {
    "category_list": category_list,
    "course_form": course_form,
}

return render(self.request, self.template_name, context)

def post(self, *args, **kwargs):
    course_form = CourseForm(self.request.POST, self.request.FILES)

    if course_form.is_valid():
        course_obj = course_form.save(commit=False)
        course_obj.author = self.request.user

        if self.request.user.is_staff or self.request.user.is_superuser:
            course_obj.check_status = True
        else:
            course_obj.check_status = False

        course_obj = Course.objects.raw("INSERT INTO courses(name, category_id,
logotype, describe, url, author_id, pub_date) VALUES(name, category_id, logotype, describe,
url, author_id, pub_date)")
        messages.success(self.request, "Курс додано.")
        return redirect(course_obj.get_absolute_url())
    else:
        messages.error(self.request, "Невірні дані.")
        context = {
            "course_form": course_form
        }
        return render(self.request, self.template_name, context)

class CourseDelete(View):
    def get(self, *args, **kwargs):
        course_obj = Course.objects.raw("SELET * FROM slug=kwargs[\"slug\"]")

        if course_obj.author != self.request.user:
            response = HttpResponse()
            response.status_code = 403
            return response

        course_del = Course.objects.raw("DELETE * FROM id=course_obj.id")
        messages.success(self.request, "Курс видалено.")
        return redirect(category.get_absolute_url())

```