МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМ. В. ДАЛЯ
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА ЕЛЕКТРОНІКИ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК ТА ІНЖЕНЕРІЇ

До захисту допускається
Завідувач кафедри
_____ Скарга-Бандурова І.С.
«____»_____ 20__ р.

МАГІСТЕРСЬКА РОБОТА

НА ТЕМУ:

Дослідження та розробка сервісу інтеграції комп'ютерних систем дистанційного навчання у традиційну систему освіти

Освітньо-кваліфікаційний рівень "Магістр"
Спеціальність 123 "Комп'ютерна інженерія"

Науковий керівник роботи:	_____ (підпис)	С.О. Сафонова (ініціали, прізвище)
Консультант з охорони праці:	_____ (підпис)	Я.О.Критська (ініціали, прізвище)
Студент:	_____ (підпис)	В.Є. Коваленко (ініціали, прізвище)
Група:		КІ-17дм

Сєвєродонецьк 2019

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВОЛОДИМИРА ДАЛЯ

Факультет Інформаційних технологій та електроніки
Кафедра Комп'ютерноих наук та інженерії
Освітньо-кваліфікаційний рівень "магістр"
Спеціальність 123 – "Комп'ютерна інженерія"
(шифр і назва)
Спеціалізація
(шифр і назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри КНІ
д.т.н., доц. *І.С. Скарга-Бандурова*
«_____» _____20____р.

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ

Коваленко Віталію Євгеновичу
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження та розробка сервісу інтеграції комп'ютерних систем дистанційного навчання у традиційну систему освіти

керівник проекту (роботи) Сафонова Світлана Олександрівна, к.т.н., доцент
(прізвище, ім.'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від «18» 10 2018 р. № 220/4
2. Строк подання студентом роботи 10.01.2019
3. Вихідні дані до роботи Матеріали науково-дослідної практики.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Огляд існуючих систем дистанційного навчання.
2. Аналіз можливості використання результатів СДН.
3. Розробка моделі взаємодії СДН з традиційною системою освіти.
4. Розробка прототипу системи інтеграції СДН
5.Охорона праці та безпека в надзвичайних ситуаціях.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Електронні плакати

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці	*Критська Я.О.*		

7. Дата видачі завдання *18.10.2018*

Керівник _____
(підпис)

Завдання прийняв до виконання _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Формування технічного завдання	*10.09.18-15.09.18*	
2	Збір необхідної інформації щодо існуючих систем дистанційного навчання	*16.09.18-16.10.18*	
3	Дослідження та аналіз існуючих методів вирішення завдання	*17.10.18-1.11.18*	
4	Розробка моделі взаємодії СДН з традиційною системою освіти	*2.11.18-23.11.18*	
5	Розробка критеріїв відповідності	*24.11.18-4.12.18*	
6	Вибір технологій для розробки прототипу системи інтеграції СДН	*5.12.18-10.12.18*	
7	Розробка прототипу системи інтеграції СДН	*11.12.18-23.12.18*	
8	Охорона праці	*24.12.18-28.12.18*	
9	Оформлення пояснювальної записки	*29.12.18-18.01.19*	

Студент _____ *Коваленко В.Є.*
(підпис) (прізвище та ініціали)

Науковий керівник _____ *Сафонова С.О.*
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Коваленко В.Є. Дослідження та розробка сервісу інтеграції комп'ютерних систем дистанційного навчання у традиційну систему освіти

Метою магістерської роботи є розробка моделі взаємодії систем дистанційного навчання з традиційною системою освіти та розробка прототипу системи, що реалізує розроблену модель. В результаті проекту була розроблена модель взаємодії систем дистанційного навчання з традиційною системою освіти та здійснено програмну реалізацію веб-клієнта і серверного додатка, що демонструє практичну можливість роботи цієї моделі.

Ключові слова: Coursera, веб-клієнт, інтеграція, онлайн система освіти, Java, Intellij IDEA, XML.

АННОТАЦИЯ

Коваленко В.Е. Исследование и разработка сервиса интеграции компьютерных систем дистанционного обучения в традиционную систему образования.

Целью магистерской работы является разработка модели взаимодействия систем дистанционного обучения с традиционной системой образования и разработка прототипа системы, реализующего разработанную модель. В результате проекта была разработана модель взаимодействия систем дистанционного обучения с традиционной системой образования и осуществлена программная реализация веб-клиента и серверного приложения, которое демонстрирует практическую возможность работы этой модели.

Ключевые слова: Coursera, веб-клиент, интеграция, онлайн система образования, Java, Intellij IDEA, XML.

ABSTRACT

Kovalenko V.E. Kovalenko V.E. Research and development of service for the integration of computer systems of distance learning in the traditional education system.

The aim of the project is to develop a model for the interaction of distance learning systems with the traditional education system and the development of a prototype system that implements the developed model. As a result of the project, a model was developed for the interaction of distance learning systems with the traditional education system and a software implementation of the web client and server application was implemented, which demonstrates the practical possibility of this model.

Key words: Coursera, web client, integration, online education system, Java, Intellij IDEA, XML.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

HTML – Мова гіпертекстової розмітки

JAVA – Мова програмування

MySQL – Система управління базою даних

IT – Інформаційні технології

ВНЗ – Вищий навчальний заклад

ОС – Операційна система

ПЗ – Програмне забезпечення

WWW – World Wide Web

СДН – Система дистанційного навчання

ПНС – Персональне Навчальне Середовище

LMS – Learning Management System

ВСТУП

Термін "дистанційна освіта" означає таку організацію навчального процесу, при якій викладач розробляє навчальну програму, що головним чином базується на самостійному навчанні студента. Таке середовище навчання характеризується тим, що учень в основному, а часто і зовсім відділений від викладача в просторі або в часі, в той же час студенти та викладачі мають можливість здійснювати діалог між собою за допомогою засобів телекомунікації. Дистанційна освіта дозволяє вчитися жителям регіонів, де немає інших можливостей для професійної підготовки або отримання якісної вищої освіти, немає університету потрібного профілю або викладачів необхідного рівня кваліфікації.

З середини 70-х років у багатьох країнах стали з'являтися навчальні заклади нового типу, так звані «відкритий», «дистанційний» університет; «електронний», «віртуальний» коледж. Вони мають оригінальну організаційну структуру, використовують своєрідний набір педагогічних прийомів, економічних механізмів функціонування.

Використання в якості інструментів відеотелеконференції, Інтернет та інших систем передачі даних «зблизить» викладача і студента, що знаходяться далеко один від одного, наблизить дистанційну освіту до традиційної, до безпосереднього спілкування викладача зі студентом, лектора з аудиторією, групових семінарських занять, апробованими століттями. Саме тому дистанційну освіту часто називають формою навчання XXI століття.

Стає очевидним, що науково-дослідна і практична робота над проблемами дистанційного навчання, методичного, методологічного і технічного забезпечення дистанційної освіти повинна бути постійною і безперервною. Можна бути впевненими, що результати такої роботи принесуть реальні плоди в сучасний освітній процес, а разом з тим в розвиток систем телекомунікацій.

Темою даного проекту є проектування клієнт-серверного додатка для інтеграції систем веб освіти в традиційну освітню систему. Для студента система дозволяє автоматизувати процес підтвердження проходження курсу, передбаченого навчальним планом у різних міжнародно-визнаних системах дистанційної освіти, використовуючи веб-клієнт. Це програмне забезпечення дозволяє частково винести процес навчання у всесвітню мережу.

Для викладача або співробітника деканату система дозволяє проконтролювати процес проходження конкретним студентом різних онлайн курсів, відповідних навчальних планів, автоматизувати процес зачитування пройдених курсів в якості контролю вивчення дисципліни.

Можливості, які відкриває онлайн-освіта перед студентами нашого часу, важко переоцінити. Рівний доступ до інформації, низькі бар'єри отримання знань, мобільність, широкі можливості вибору тем і спеціалізацій і постійно оновлюваний матеріал – це основні переваги платформ, які в перспективі замінять традиційні інститути. А поки цього не сталося, дозволяє студентам самим вибирати курс, який вони хочуть прослухати, в той же час стимулює викладача бути конкурентоспроможним не тільки в рамках власного університету, а й в рамках всього світу.

1 ДИСТАНЦІЙНА ОСВІТА

1.1 Історія розвитку систем дистанційного навчання

Ідея вчитися у інших на відстані далеко не нова. Деякі вчені стверджують, що священні послання Святого Павла, що розсилаються по храмах, служать ілюстрацією окремих ключових положень дистанційної освіти. У 1840 році Ісаак Пітман (Isaac Pitman) за допомогою поштових відправлень почав навчати стенографії студентів в Об'єднаному Королівстві, ставши, таким чином, родоначальником першого дистанційного освітнього курсу. У 50-ті роки XIX століття в Німеччині Густав Лангеншайдт (Gustav Langenscheidt) опублікував свої Leh rbriefe (букв. «Навчальні листи») як самовчитель по мові для дорослих.

Можливість здобувати вищу освіту на відстані з'явилася в 1836 році, коли в Об'єднаному Королівстві був заснований Лондонський Університет (University of London). Студентам, які навчалися в акредитованих навчальних закладах, було дозволено здавати іспити, що проводяться Університетом.

Починаючи з 1858 року ці іспити стали відкритими для кандидатів з усього світу, незалежно від того, де і яким чином вони здобували освіту. Такий стан справ призвів до виникнення низки коледжів, які пропонували курси навчання по пошті відповідно до університетської програми.

У 70-ті роки XIX століття в Америці також було здійснено ряд кроків з організації дистанційного навчання. Так, в 1873 році Анна Еліот Тікнор (Anna Eliot Ticknor) створила систему навчання поштою для жінок під назвою Товариство Тікнор (Ticknor's Society), взявши за основу англійську програму «Суспільство підтримки домашнього навчання» («Society for the Encouragement of Home Study»). У 1874 році програму навчання поштою запропонував Університет штату Іллінойс (Illinois State University).

У Пенсильванії щоденна газета під назвою «Кол'єрі Інжініе» стала публікувати навчальні матеріали, спрямовані на поліпшення техніки гірських розробок і запобігання нещасним випадкам на рудниках. Ці публікації мали такий величезний успіх, що в 1891 році був розроблений самостійний курс, що послужив моделлю для програм навчання поштою різних предметів. Вільям Рейні Харпер (William Rainey Harper), що вважається в Америці «батьком навчання поштою» (Mackenzie and Christensen, 1971, стор. 7), в 1892 році заснував перше університетське відділення дистанційного навчання в Університеті Чикаго

(University of Chicago), почавши експериментувати з позакласним викладанням в Баптистській теологічній семінарії. У 1906 році викладання поштою було введено в Університеті штату Вісконсін (University of Wisconsin).

Досить рано дистанційне навчання з'явилося в Австралії. У 1911 році почали свою роботу курси університетського рівня в Квінслендському університеті (University of Queensland) в Брісбені. У 1914 році було організовано навчання поштою за програмою початкової школи дітей, які живуть далеко від звичайних шкіл. Студенти педагогічного коледжу в Мельбурні проводили свої уроки, використовуючи пошту. Подібна практика незабаром поширилася на середні школи і технічні училища. Аналогічні системи для школярів стали використовуватися в Канаді та Новій Зеландії. У 1938 році у Вікторії (Британська Колумбія, Канада) відбувся перший з'їзд Міжнародної Ради по освіті поштою (International Council for Correspondence Education).

Після революції 1917 року дистанційна освіта стала розвиватися в Росії. Тут пропонувалися різні курси на самих різних рівнях. У Радянському Союзі була розроблена особлива, «консультаційна» модель дистанційної освіти, назва якої буквально означала «освіту без візуального контакту» (заочна освіта). До 60-х років XX століття в СРСР було 11 заочних університетів і безліч заочних факультетів в традиційних вищих навчальних закладах. Після Другої світової війни прикладом СРСР пішли інші країни Центральної та Східної Європи.

Що стосується Західної Європи, то в 1939 році у Франції для навчання поштою дітей, позбавлених можливості відвідувати школу, був створений Державний центр дистанційного навчання (Centre National d'Enseignement a Distance, CNED). В даний час цей центр став найбільшим навчальним закладом дистанційної освіти у Європі. У 1946 році на дистанційні форми навчання перейшов Південноафриканський університет (University of South Africa, UNISA).

Величезний вплив на систему дистанційної освіти здійснило відкриття у 1969 році Відкритого університету Великобританії (Open University of the United Kingdom, UKOU): навчальні заклади, що ведуть навчання на відстані, з'явилися в цілому ряді країн переважно Європи та Азії. Серед них Universidad Nacional de Educacion a Distancia (UNED) в Іспанії (1972), Allama Iqbal Open University (AIOU) в Пакистані (1974), Sukhothai Thammathirat Open University (STOU) в Таїланді (1978), Корейський державний відкритий університет (Korea National Open University, KNOU) (1982), Universitas Terbuka (UT) в Індонезії (1984) і Державний відкритий університет ім. Індіри Ганді (Indira Gandhi National Open University, IGNOU) в Індії (1985).

У Китаї замість закритих в період культурної революції традиційних вищих навчальних закладів в 1979 році була створена Національна мережа радіо і телевізійних університетів (Central Radio and TV University, CRTVU). Навчання тут організовано з використанням супутникового мовлення і телевізійних університетів в провінції.

XXI століття – справжня віха в історії дистанційного навчання. Інтернет-технології привнесли нові можливості, які зробили навчальний процес більш зрозумілим і доступним. При цьому в арсеналі учнів є:

- рукописні та друковані матеріали (рукописи, підручники, радіокурси, телекурси);
- комплексний підхід в дистанційній освіті, початок двосторонньої взаємодії (спеціалізовані посібники, консультації у викладачів, експрес-курси);
- прогресивне використання інформаційних, а також комунікаційних технологій (синхронний і асинхронний режим двостороннього зв'язку: відеоконференції, Інтернет, листування по електронній пошті).

Але життя не стоїть на місці і, цілком можливо, незабаром буде новий етап в історії дистанційного навчання, адже йому пророкують появу інноваційних освітніх засобів з активним використанням штучного інтелекту. Це дозволить модернізувати існуючі технології освітньої системи і зробити віддалену освіту безмежною.

1.2 Сучасні принципи побудови систем дистанційного навчання

Сучасна система дистанційного навчання повинна забезпечувати:

- централізоване автоматизоване управління навчанням;
- швидке і ефективне розміщення і надання навчального контенту учням;
- єдину платформу для вирішення основних завдань у рамках планування, проведення та управління всіма навчальними заходами в організації;
- підтримку сучасних стандартів в сфері технологій дистанційного навчання;
- персоналізацію навчального контенту і можливість його багаторазового використання;
- широкий діапазон засобів організації взаємодії між усіма учасниками навчального процесу.

Весь функціонал, яким володіють сучасні системи дистанційного навчання, можна розділити на три основні блоки: управління навчанням, забезпечення взаємодії учасників навчального процесу, розробка навчального контенту [1].

В рамках блоку управління навчанням системи дистанційного навчання надають такі основні функціональні можливості:

- управління компетенціями;
- автоматизоване формування навчальних програм;
- управління профілями користувачів;
- управління доступом до дистанційних курсів та тестів;
- журналювання діяльності користувачів;
- забезпечення технічної та методичної підтримки користувачів;
- формування звітів;
- аналіз процесу навчання.

В рамках блоку забезпечення взаємодії учасників навчального процесу системи дистанційного навчання надають такі основні засоби організації спілкування користувачів:

- форум;
- чат;
- відеоконференція;
- блог;
- Wiki;
- віртуальна класна кімната і т.д.

Третій блок (розробка навчального контенту) містить набір інструментів, які вирішують широкий спектр завдань: від створення простих тестів для проведення тестування слухачів до розробки складних мультимедійних курсів. Однак, необхідно відзначити, що не всі системи включають в себе засоби розробки навчального контенту, справедливо припускаючи, що в якості засобів розробки можуть використовуватися програмні продукти сторонніх виробників [1].

До основних тенденцій в сфері систем дистанційного навчання можна віднести:

- включення в функціонал систем дистанційного навчання елементів Web 2.0;
- розвиток інструментів управління компетенціями;
- розширення персоналізації навчального контенту;
- надання Систем дистанційного навчання відповідно до принципу SaaS (Software as a Service).

Необхідно відзначити, що в своєму розвитку системи дистанційного навчання значно випередили засоби розробки навчального контенту. Зараз не існує проблеми вибору і впровадження системи дистанційного навчання, яка покриє основні потреби функціонального замовника. Однак, основною проблемою з якою зіткнеться організація, яка захоче

використовувати технології дистанційного навчання, стануть труднощі з розробкою необхідного навчального контенту (висока вартість і тривалість розробки) [1].

Таким чином, подальший розвиток систем дистанційного навчання багато в чому залежить від змін, які відбуватимуться в сфері розробки навчального контенту.

1.3 Класифікація систем дистанційного навчання

Оскільки дистанційне навчання – це не тільки і не стільки технологія, скільки втілення в освітній сфері процесів віртуалізації і деталізації інформації, комунікацій та професійної діяльності, які змінюють все, то його класифікація – складне завдання і може розглядатися з абсолютно різних сторін: дидактичної, організаційної, технологічної і т. д..

Існують різні підходи до організації систем дистанційного навчання. Ці підходи, перш за все, відрізняються завдяки використанню різних способів доставки навчального матеріалу та методичними прийомами, які визначаються контекстом навчання (цілі, контингент, умови). Кожне конкретне застосування систем дистанційного навчання вимагає точної ідентифікації варіанту, який найбільшою мірою підходить для вирішення поставлених завдань.

Таким чином, для визначення варіанту побудови систем дистанційного навчання, найбільш прийнятного для конкретного випадку, необхідна класифікація набору критеріїв, що визначають архітектуру системи.

Архітектурні варіанти систем дистанційного навчання на основі засобів автоматизації визначаються наступним: методичне, організаційне забезпечення; програмне та інформаційне забезпечення; апаратне забезпечення [2]. Класифікація систем дистанційного навчання також необхідна і в тому випадку, коли для їх реалізації не використовуються програмно-апаратні засоби.

Однак дистанційне навчання річ неоднорідна, його можна класифікувати, принаймні, на чотири типи:

1. Навчання при взаємодії учня виключно (або в великою мірою) з освітніми ресурсами, з мінімальною (або відсутньою) участю в навчанні викладача. Цей метод реалізується завдяки великій кількості друкованих, інтерактивних, відео та аудіо матеріалів. Такий спосіб навчання можна охарактеризувати як самонавчання. При ряді недоліків він все ж користується чималою популярністю.

2. Індивідуальне викладання, засноване на особистому контакті учня і викладача. Це метод також дуже популярний в дистанційній освіті і може здійснюватися за допомогою голосового зв'язку через Інтернет, чат, електронну пошту і т.д. Даний спосіб навчання дуже

дієвий, враховуючи, що при сучасних технологіях учень може не тільки чути, але й бачити викладача, що, безумовно, є позитивним фактором.

3. Метод запрограмованого навчання. Даний тип навчання зустрічається досить часто і іноді є доповненням до стаціонарного. Класичний приклад – курс аудіолекцій, записаний викладачем за принципом «один до багатьох». Також часто можна зустріти відеокурси, спеціальні комп'ютерні програми і т.д. Здебільшого, такого роду навчання комбіноване, і проводиться не одним, а всіма можливими засобами цієї підгрупи. Нерідко навчання такого типу формується навколо якогось навчального сайту, можливо, закритого типу, а також можливі хоч і рідкісні, але все ж реальні з'їзди для навчання.

4. У цю підгрупу можна віднести методи, характерною рисою яких є навчання групою осіб один одного чомусь корисному. Наприклад, є спеціалізовані сайти, на яких носії різних мов вступають в контакт з представниками іншої мовної групи з метою оволодіння тією чи іншою мовою.

1.4 Недоліки традиційної системи освіти

Дистанційне навчання через Інтернет відрізняється від звичного заочного навчання своєю безперервністю. Якщо студент-заочник осягає ази знань самостійно за допомогою підручників, а задати питання викладачам може тільки три рази в рік під час настановних і лабораторно-екзаменаційних сесій, то людина, яка навчається за допомогою дистанційних технологій через Інтернет, має можливість постійно, протягом навчального року контактувати з викладачами.

На заочному навчанні темп засвоєння знань фіксований та єдиний для всієї групи, а при дистанційному навчанні через Інтернет у кожного студента є можливість присвятити більше сил і часу для поглибленого опрацювання більш складних і важливих для нього дисциплін. Зрозуміло, за кожну вивчену дисципліну студенти звітують не менше суворо, ніж при отриманні освіти за класичною заочною формою.

Дистанційне навчання через Інтернет має безліч вагомих переваг перед традиційними формами навчання: можливість займатися в зручний для Вас час і в зручному місці; постійна доступність навчального матеріалу, спілкування з викладачами та однокурсниками; можливість оволодіння сучасними комп'ютерними і телекомунікаційними технологіями.

Якщо порівнювати витрати на навчання за традиційними формами освіти і витрати на дистанційне навчання, то можна прийти до висновку, що дистанційне навчання економічно більш ефективно. Так, наприклад, із загальних витрат на навчання можна відняти витрати на проїзд до місця навчання, проживання та харчування в період навчання. Крім того, час для

вивчення матеріалу вибирається студентом самостійно, що дозволяє людині більш гнучко використовувати свій час, значно економити його для інших важливих справ і роботи.

У традиційній освіті існують певні обмеження на час навчання, так звані семестри. В цей термін студент повинен вкластися і здати необхідні дисципліни. Однак, якщо студент готовий скласти іспити значно раніше закінчення семестру, існують певні проблеми і складності в здійсненні цього задуму. У дистанційному навчанні таких обмежень немає. Навчальні матеріали, тести та іспити доступні студенту в будь-який час. Він вільний самостійно вибирати час і порядок дисциплін і може, наприклад, за досить короткий термін інтенсивних занять вивчити і здати дисципліну, внаслідок чого достроково скласти іспити за весь навчальний рік. Це особливо зручно для людей, які вже мають базовий рівень освіти, які вирішили розширити свої знання і навчаються в скорочені терміни.

Студент не обмежений територіально тим місцем, де він повинен здобути дистанційну освіту. Існує лише одне обмеження – можливість доступу до середовища передачі інформації. Електронні методичні посібники та тести на базі інтернет-технологій можна переглядати і здавати в будь-якому місці. Кожен з нас звик засвоювати інформацію і знання по-своєму. Одному необхідно довго повторювати новий матеріал, іншому достатньо побіжного погляду, щоб зрозуміти суть, третій схильний до вивчення за методом від часткового до загального, четвертий, навпаки, вважає за краще спочатку вивчити теорію, а потім приклади. В дистанційній освіті завдяки великому спектру можливостей і технологій у студента з'являється можливість самостійно вибрати, яким способом він вивчатиме обрану дисципліну.

Можливо, хтось спочатку захоче познайомитися з навчальними матеріалами, потім вивчити теорію, закріпивши тестом на практиці, а хтось все зробить навпаки, почне з тренувального тесту, щоб усвідомити обсяг необхідних знань, потім вивчить ці знання на основі теоретичних матеріалів, а потім перегляне практичні приклади на відео. Свобода вибору способу вивчення дисциплін – особливість дистанційної освіти.

Студентам, які навчаються дистанційно, доводиться мати справу з новітніми технологіями подання та обробки інформації, такими, як комп'ютерні технології, мультимедіа-технології, широкомовні кабельні і супутникові системи, Інтернет тощо. Тому їм мимоволі доводиться освоювати ці технології, отримуючи додаткові навички та вміння, які значно підвищують загальноосвітній і технічний рівень студента. Однак не варто переживати про недостатньо високий рівень кваліфікації в цій області на момент початку навчання, робота з системою дистанційної освіти інтуїтивно зрозуміла і не вимагає серйозних знань в цих областях.

Безліч освітніх установ готують і випускають фахівців з певним профілем. Найчастіше спеціалізація залежить від потреб регіону, в якому розташований ВНЗ. Що ж робити студенту, який бажає самостійно вивчити непопулярні в цьому регіоні дисципліни і науки? Студент знаходиться перед вибором – або їхати вчитися у віддалений навчальний заклад, або за допомогою доступних йому технологій навчатися в онлайн університеті. І, як показує досвід, багато хто вибирає шлях отримання знань через дистанційну освіту.

1.5 Кібер-університет

Використання дистанційних систем навчання разом з традиційною системою освіти можна вважати одним з етапів побудови системи кібер-університету.

Розумний кібер-університет – метрична культура соціально-технологічних відносин, яка об'єднує в мережу кадри і розумну інфраструктуру, для виконання актуальних наукових досліджень і підготовки затребуваних ринком фахівців з академічними і науковими ступенями шляхом адекватного моніторингу та хмарного управління оцифрованими науково-освітніми процесами і явищами в цілях залучення інвестицій і досягнення високої якості життя співробітників.

Рисунок 1.1 – Розумний кібер-університет

Метрика вченого міжнародного рівня включає: знання високих технологій, іноземних мов і наукові досягнення, підтверджені публікаціями з індексами наукометричних баз даних. Метрика іміджу університету:

а) відсутність корупції і високий рівень зарплати співробітників;

б) міжнародний рівень наукових досягнень і якості випускників;

в) наявність критичної маси авторитетних в світі вчених;

г) виступи науковців на провідних міжнародних конференціях;

д) організація та проведення власних конференцій, спонсорованих суспільством IEEE, ACM;

е) членство студентів і вчених в згаданих організаціях.

Інноваційні сервіси, що формують розумний кібер-університет як структурний прототип глобального науково-освітнього віртуального кіберпростору Global Smart Cyber University:

а) Хмарний кібер-сервіс захищеного електронного документообігу для цифрового моніторингу та інтелектуального кіберуправління науково-освітніми процесами (створення, реалізація та утилізація документа), в форматі замкнутого циклу: «факт - вимір - оцінка - дія», який повністю виключає паперові носії шляхом використання Cloud-Mobile Service Computing, баз даних, цифрового підпису, ID-card, пошти та мобільного телефону.

б) Хмарний кібер-сервіс мобільного голосування e-voting для моніторингу громадської думки; реалізації студентських опитувань; прийняття рішень на оперативних нарадах, засіданнях вченої ради, конференціях трудового колективу; проведення виборів експертів, студентського сенату, керівного та науково-педагогічного складу при заміщенні вакантних посад.

в) Хмарний кібер-сервіс управління персоналом на основі online моніторингу, вимірювання, рейтингування та накопичення цифрових метрик компетенцій для оцінювання діяльності студентів і всіх категорій співробітників з метою вироблення прозорих регуляторних моральних і матеріальних стимулів, вибору переможців з претендентів на вакантні позиції керівників і науково-педагогічні посади.

г) Хмарний кібер-сервіс управління структурним підрозділом на основі online моніторингу, вимірювання та накопичення цифрових метрик компетенцій кафедри, пов'язаних з науково-освітнім процесом для вироблення регуляторних дій, що управляють, і генерування пакету документів, необхідного для життєдіяльності.

д) Хмарний кібер-сервіс оцінки якості освітніх процесів і компонентів, online тестування знань і умінь, що виключає нелегітимні відносини між викладачем і студентом при здачі іспитів і заліків.

е) Хмарний кібер-сервіс управління науковими процесами на основі цифрового оцінювання діяльності вчених, підрозділів, наукових результатів, проектів і пропозицій по метриках, розроблених експертами, з метою прозорого та легітимного розподілу фінансових, кадрових і часових ресурсів між підрозділами і співробітниками.

є) Хмарний кібер-сервіс надання освітніх послуг у вигляді MOOC online і onsite курсів, а також управління освітнім процесом на основі прозорого розподілу фінансових і тимчасових (кредитних) ресурсів між підрозділами і співробітниками в суворій відповідності з метричним оцінюванням вкладу кожного суб'єкта в актив і імідж університету.

ж) Хмарний кібер-сервіс моніторингу та управління науково-освітнім процесом студента в реальному масштабі часу, генерування і зберігання електронних документів для його супроводу в часі і просторі за допомогою створення персонального віртуального кабінету, пов'язаного з мобільним пристроєм і e-mail.

з) Хмарний кібер-сервіс вимірювання і супроводу бакалаврських, магістерських та дисертаційних робіт, а також конкурсних проектів на основі інтеграції міжнародних метрик оцінювання наукової та практичної значущості результатів проведених досліджень з внутрішніми критеріями якості, розробленими експертами.

и) Хмарний кібер-сервіс ліцензування та акредитації спеціальностей на основі вимірювання науково-освітньої діяльності кафедр і подальшого генерування пакета документів, необхідного для зовнішнього оцінювання якості навчальних процесів.

і) Хмарний кібер-сервіс електронного 24/7 доступу та моніторингу присутності співробітників і студентів в інфраструктурних аудиторіях університету на основі використання мобільних пристроїв і ID-card, а також електронний банкінг для оплати освітніх послуг і використання корпоративних кафедральних карт для придбання товарів і послуг в межах зароблених кафедрою коштів.

ї) Хмарний кібер-сервіс захисту інформаційно-фізичного простору університету і санкціонування електронного доступу в усі кіберфізічні компоненти і процеси, пов'язані з життєдіяльністю університету.

Рисунок 1.2 – Інноваційні сервіси розумного кібер-університету

1.6 Постановка завдання

Об'єкт дослідження – система дистанційного навчання в рамках традиційної системи освіти.

Предмет дослідження – моделі і методи автоматизації процесу підтвердження проходження курсу, передбаченого навчальним планом у різних міжнародно-визнаних системах дистанційної освіти.

Мета дослідження – розробка методу та засобів автоматизації процесу підтвердження проходження курсу, передбаченого навчальним планом у різних системах дистанційної освіти.

Для досягнення цілей атестаційної роботи необхідно вирішити наступні завдання:

- проаналізувати предметну область, розглянути існуючі системи дистанційного навчання та способи їх впровадження в традиційну систему освіти;
- дослідити можливості використання систем дистанційного навчання в рамках вищої освіти в Україні і вибрати процедури контролю вивчення дисципліни, передбаченою навчальним планом;
- вибрати стек технологій та середовище розробки програмного продукту;
- розробити прототип програмного забезпечення для впровадження системи дистанційного навчання в традиційну систему освіти;
- сформулювати висновки.

Виходячи з означених завдань, можна сформулювати наступну концепцію проекту: для студента система повинна дозволяти автоматизувати процес підтвердження проходження курсу, передбаченого навчальним планом у різних дистанційних системах навчання, використовуючи веб-клієнт; для викладача або співробітника деканату система повинна спростити процес контролю проходження студентом дисциплін, передбачених навчальним планом в дистанційних системах навчання, а також автоматизувати процес зачитування пройдених курсів в якості контролю вивчення дисципліни.

2 АНАЛІЗ ІСНУЮЧИХ ДИСТАНЦІЙНИХ СИСТЕМ НАВЧАННЯ

Для огляду і аналізу були обрані дві найбільш відомі англомовні системи дистанційної освіти – Coursera і Udacity і російськомовна система ІНТУІТ (Інтернет-університет інформаційних технологій).

2.1 Coursera

Coursera – проект в сфері масової онлайн-освіти, заснований професорами інформатики Стенфордського університету Ендрю Ином та Дафною Коллер. В його рамках існує проект з публікації освітніх матеріалів в Інтернеті у вигляді набору безкоштовних онлайн-курсів [3].

Проект співпрацює з університетами, які публікують і ведуть в системі курси з різних галузей знань. Слухачі проходять курси, спілкуються з однокурсниками, здають тести та іспити безпосередньо на сайті Coursera, також поширюється офіційно мобільний додаток для iPhone і Android. На лютий 2017 року Coursera зареєстровано 24 млн користувачів і більше 2000 курсів і 160 спеціалізацій від 149 освітніх установ.

У проекті представлені курси з фізики, інженерних дисциплін, гуманітарних наук і мистецтва, медицини, біології, математики, інформатики, економіки і бізнесу. Тривалість курсів приблизно від шести до десяти тижнів, з 1-2 годинами відеолекцій на тиждень. Курси містять завдання, щотижневі вправи і іноді заключний проект або іспит.

На відміну від таких проектів, як Academic Earth, в проекті пропонуються не окремі лекції, а повноцінні курси, які включають відеолекції з субтитрами, текстові конспекти, домашні завдання, тести та підсумкові іспити. Доступ до курсів обмежений за часом; кожне домашнє завдання або тест має бути виконано тільки в певний період часу. Після закінчення курсу, за умови успішного складання проміжних завдань і заключного іспиту, слухачеві видається сертифікат про закінчення.

Станом на 2014 рік основна частина курсів представлена англійською мовою, є курси китайською, іспанською, французькою, російською, португальською (більше десятка), є по кілька курсів і на інших мовах. При цьому активно додаються субтитри на багатьох мовах світу, які створюються слухачами на добровільних засадах. Для створення субтитрів російською мовою запущений проект "Переведемо Coursera", в якому на початок 2015 року зареєстроване 15 тисяч учасників і переведено 30 курсів.

Серед курсів відомих лекторів, опублікованих в проекті – «Машинне навчання» (Ендрю Ин), «Імовірнісні графічні моделі» (Дафна Колер), «Теорія автоматів» (Джеффрі Ульман), «Принципи функціонального програмування на мові Scala» (Мартін Одерски), «Керівництво для початківців з ірраціональної поведінки» (Ден Аріелі).

У 2012 році Coursera почала працювати зі Стенфордом, Прінстоном, університетом Мічігану і Пенсільванським університетами. 12 освітніх установ-партнерів були додані в липні 2012 року, і ще 17 – у вересні 2012 року.

У лютому 2013 року проект повідомив про ще 29 партнерських університетів. Станом на 2014 рік кількість партнерів – 108. Серед університетів, які співпрацюють з проектом – Університет Джонса Хопкінса, Каліфорнійський технологічний інститут, Единбурзький університет, Університет Торонто, Колумбійський університет, Пенсільванський університет, Московський фізико-технічний інститут, Вища школа економіки, Новосибірський державний університет .

У січні 2013 року Coursera повідомила, що Американська освітня рада схвалила п'ять курсів, рекомендованих для заліку в коледжах США:

- «Алгебра» від Каліфорнійського університету в Ірвіні;
- «Початок математичного аналізу» від Каліфорнійського університету в Ірвіні;
- «Введення в генетику і еволюцію» від Дюкського університету;
- «Біоелектрика: кількісний метод» від Дюкського університету;
- «Математичний аналіз, функції однієї змінної» від Пенсільванського університету.

Після закінчення таких курсів проводиться очний онлайновий іспит (за допомогою веб-камери) в акредитованій екзаменаційній службі. Послуга коштує $60 – $90.

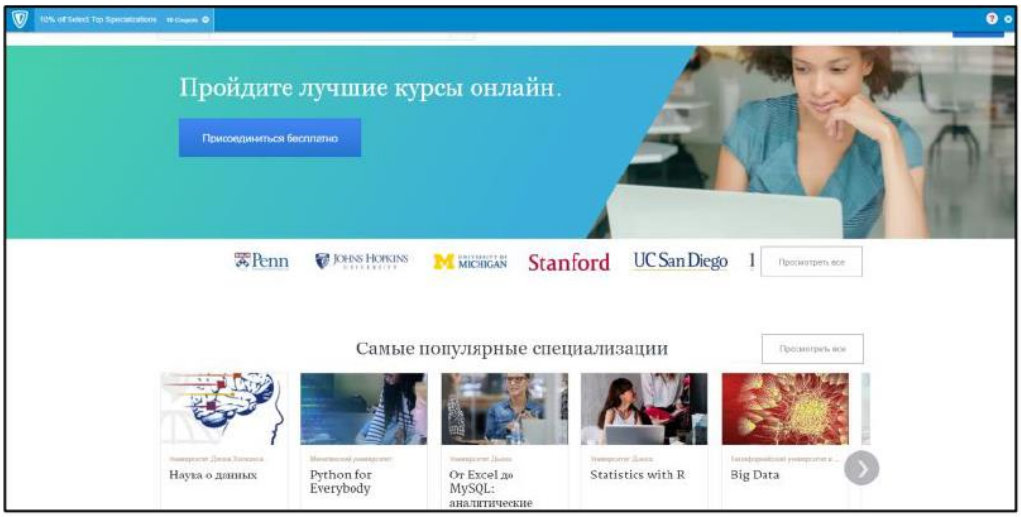


Рисунок 2.1 – Головна сторінка Coursera

2.2 Udacity

Udacity – приватна освітня організація, заснована Себастіаном Труном, Девідом Ставенсом (David Stavens) та Майком Сокольськи (Mike Sokolsky), з метою демократизації освіти. Компанія виникла в результаті розширення програми з інформатики Стенфордського університету. Дистанційні курси доступні безкоштовно через Інтернет, прослухати їх може будь-хто. Спочатку пропонувалося шість курсів. Станом на 1 жовтня 2012 року Udacity пропонує 14 курсів. Число студентів становить десятки тисяч людей [4].

Про заснування Udacity було оголошено в 2012 році на конференції Digital Life Design.

У 2012 році Себастьян Трун був відзначений газетою The Guardian як людина, що внесла істотний внесок в розвиток відкритого Інтернету.

Перші два курси – «CS 101: Створення пошукової системи» (викладач Дейв Еванс з Віргінського університету), і «CS 373: Програмування безпілотних автомобілів» (викладач С. Тран) почалися 20 лютого 2012р. В обох курсах використовується мова програмування Python.

Станом на листопад 2013 року сайт пропонує 24 курси, серед яких, наприклад:

- CS101: Створення пошукової системи;
- CS212: Розробка комп'ютерних програм;
- CS215: Алгоритми: аналіз соціальних мереж;
- CS253: Розробка веб-додатків;
- CS262: Мови програмування;
- CS373: Програмування безпілотних автомобілів;
- CS387: Прикладна криптографія.

Відео-лекції англійською мовою з субтитрами в поєднанні з вбудованими тестами і подальшими домашніми роботами засновані на моделі «вчитися на практиці». Кожна лекція включає в себе вбудований тест, щоб допомогти студентам зрозуміти пропоновані концепції та ідеї.

До штучного інтелекту, робототехніки та фізики додалися курси з ведення блогів і створення стартапів, але не все йде гладко. Багато ідей провалилися, як, наприклад, курс з дискретної математики – рівень виявився занадто низьким порівняно з іншими. З іншого боку, на форумах досі багато незадоволених: курси, на їхню думку, перебувають за межею можливостей і штучно ускладнені. Команда проекту працює над цим – простіше, звичайно, не

буде, але способи надання матеріалу і тести постійно переглядаються, щоб якомога більше студентів змогли освоїти предмет, незважаючи на складність.

Студенти можуть зареєструватися на один або кілька класів до дати здачі першого домашнього завдання. Після закінчення курсу студенти безкоштовно отримують сертифікат про закінчення, підписаний викладачами [4].

Починаючи з серпня 2012 року для деяких курсів можливе виконання очного іспиту в Pearson VUE, вартість якого складає близько 90 доларів США.

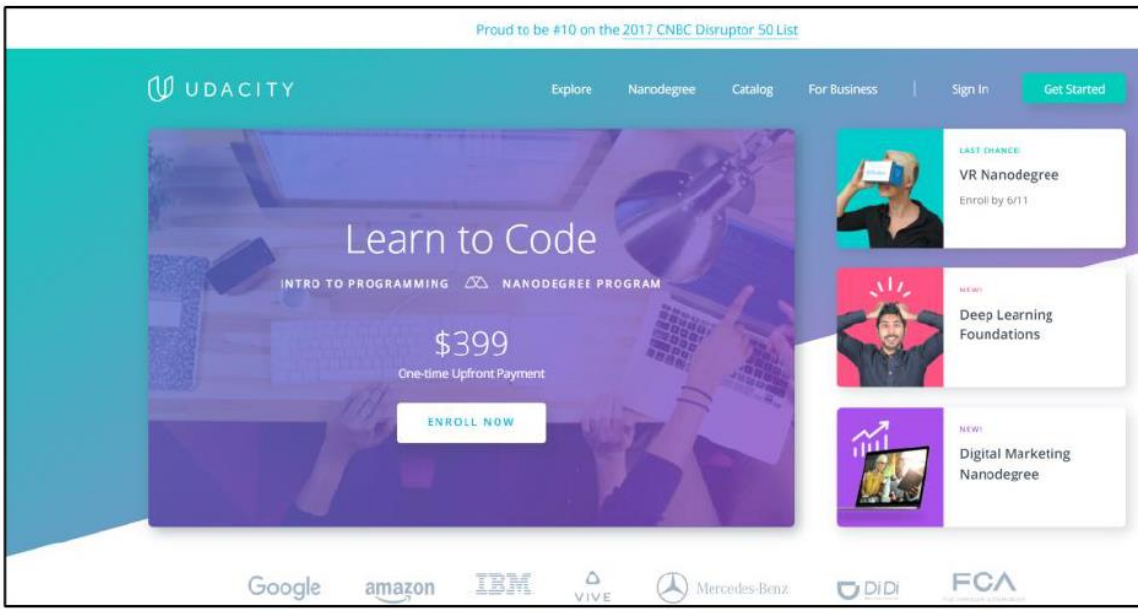


Рисунок 2.2 – Головна сторінка Udacity

2.3 ІНТУІТ

Недержавна освітня приватна установа додаткової професійної освіти «Національний Відкритий Університет « ІНТУІТ» – це освітній проект, головними цілями якого є вільне поширення знань у Всесвітній Мережі та надання послуг дистанційного навчання [5].

На сайті проекту представлені у відкритому і безкоштовному доступі понад 800 навчальних курсів за тематиками комп'ютерних наук, інформаційних технологій, математики, фізики, економіки, менеджменту і іншим областям сучасних знань.

Книжкові серії «ІНТУІТ» «Основи інформаційних технологій», «Основи інформатики та математики» (спільно з МГУ ім. М.В. Ломоносова), «Основи економіки та менеджменту» (спільно з ВШБІ НДУ ВШЕ) об'єднують кілька сотень книг і електронних підручників .

«ІНТУІТ» організовує зйомку відеокурсів і лекцій в провідних університетах та в телестудії. Відтак проект налічує кілька тисяч годин лекцій відомих професорів і доповідей вчених.

Проект співпрацює з навчальними закладами, навчальні матеріали «ІНТУІТ» активно використовуються в навчальному процесі більше ніж у 500 вишах Російської Федерації та в інших країнах.

Проект неодноразово відзначався нагородами регіональних і загальнонаціональних конкурсів, в тому числі і «Національною премією Рунет». Проект є одним з найпопулярніших освітніх ресурсів і має великий потенціал зростання [5].

Розглянемо історію розвитку онлайн-університету ІНТУІТ.

4 березня 2016 р. Отримано ліцензію на право ведення освітньої діяльності за програмами додаткової професійної освіти.

31 січня 2012 р. Оголошено перший набір на програми вищої та другої вищої освіти.

6 грудня 2011 р. Отримано ліцензію на право ведення освітньої діяльності за програмами вищої освіти.

25 травня 2011 р. Отримана нова безстрокова ліцензія на освітню діяльність на програми додаткової освіти.

10 жовтня 2010 р. Отримано ліцензію на право ведення освітньої діяльності за програмами додаткової освіти.

15 квітня 2010 р. З'явилися перші курси компанії Intel.

9 квітня 2010 р. Всі курси ІНТУІТ стали виходити в форматі для електронних книг.

23 вересня 2009 р. Вийшли перші підручники в серії «Ліцей інформаційних технологій».

27 лютого 2009 р. Оголошено перший спільний з компанією Microsoft конкурс з розробки навчальних курсів з інформаційних технологій.

28 жовтня 2008 р. Всі курси ІНТУІТ переведені в формат SCORM2004 для систем дистанційного навчання.

20 жовтня 2008 р. Відкрито Інтернет-Університет Суперкомп'ютерних Технологій.

21 травня 2008 р. Вийшов перший відеокурс.

20 березня 2008 р. ІНТУІТ проводить перші очні курси у Вищій школі економіки.

7 серпня 2007 р. Вийшов перший підручник в серії «Основи економіки та менеджменту», що видається спільно з Вищою школою бізнес-інформатики Державного Університету «Вища школа економіки».

19 жовтня 2006 р. Вийшов сотий навчальний курс.

31 січня 2006 р. Вийшла мобільна версія сайту.

26 листопада 2005 р. ІНТУІТ став лауреатом національної премії Російської Федерації за внесок в розвиток російського сегмента мережі Інтернет в номінації "Наука і освіта".

10 листопада 2005 р. Перший випуск студентів за спільною програмою з ФІТ НГУ.

25 травня 2005 р. Вийшли перші підручники нової серії «Основи інформатики та математики», що видається разом з МДУ ім. М.В. Ломоносова.

5 травня 2005 р. ІНТУІТ став переможцем всеросійського конкурсу освітніх ресурсів «ІТ-освіта в Росії».

25 серпня 2004 р. Підписано угоду про стратегічне партнерство з факультетом інформаційних технологій Новосибірського Державного Університету (ФІТ НДУ).

10 вересня 2003 р. Вийшла версія навчальних курсів на CD.

10 квітня 2003 р. Відкриття проекту, запущений сайт Інтернет-Університету Інформаційних Технологій.

3 квітня 2003 р. Вийшов перший підручник «Основи web-технологій» (автори Храмцов П.Б. і ін.) в серії «Основи інформаційних технологій».

31 січня 2003 р. Створено перший навчальний курс.

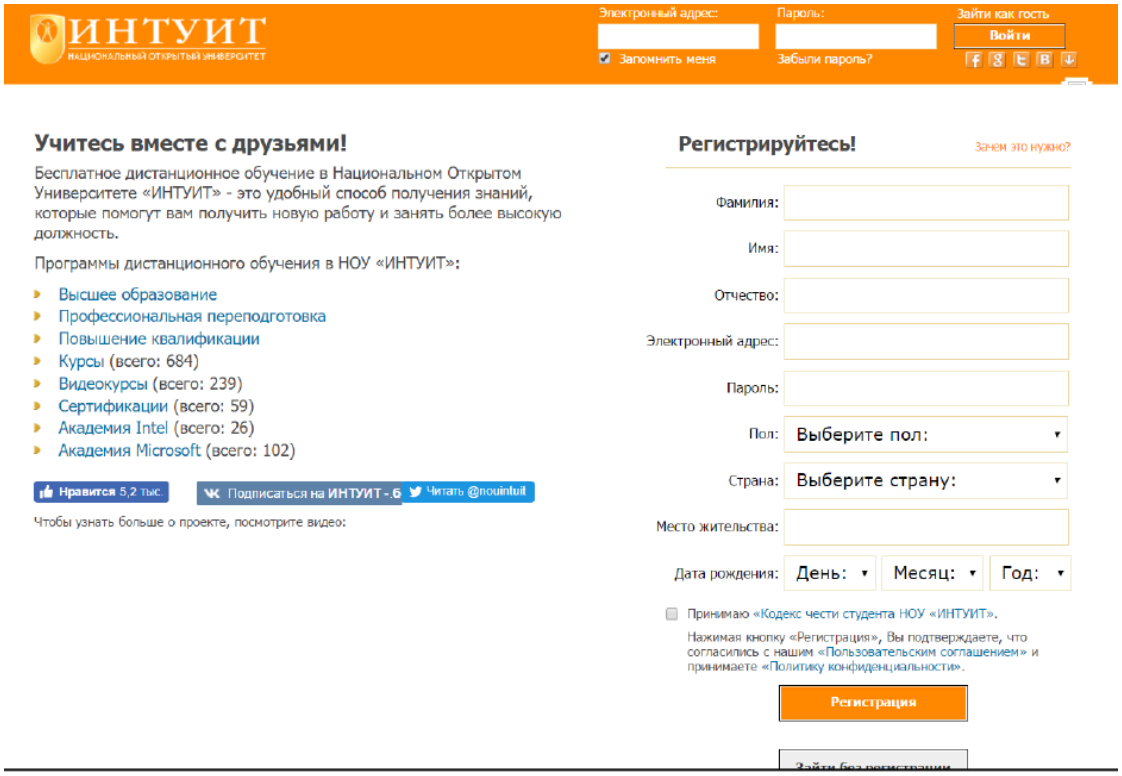


Рисунок 2.3 – Головна сторінка ІНТУЇТ

2.4 Верифікація проходження курсу

Більшість наведених вище систем після успішного проходження учням курсу видають сертифікат, що підтверджує це проходження. У різних системах навчання верифікація відбувається по-різному, наприклад, в системі Coursera посилання для верифікації валідності сертифікатів вбудоване в сам сертифікат, в той же час в системі ІНТУІТ посилання для валідації сертифіката знаходиться в особистому кабінеті студента, і цим посиланням можна поділитися, якщо необхідно. Посилання на верифікаційні сторінки можуть виглядати наступним чином – на Рисунку 2.5 представлено приклад сертифіката для Coursera і https://www.intuit.ru/verifydiplomas/101163201 для ІНТУІТ.

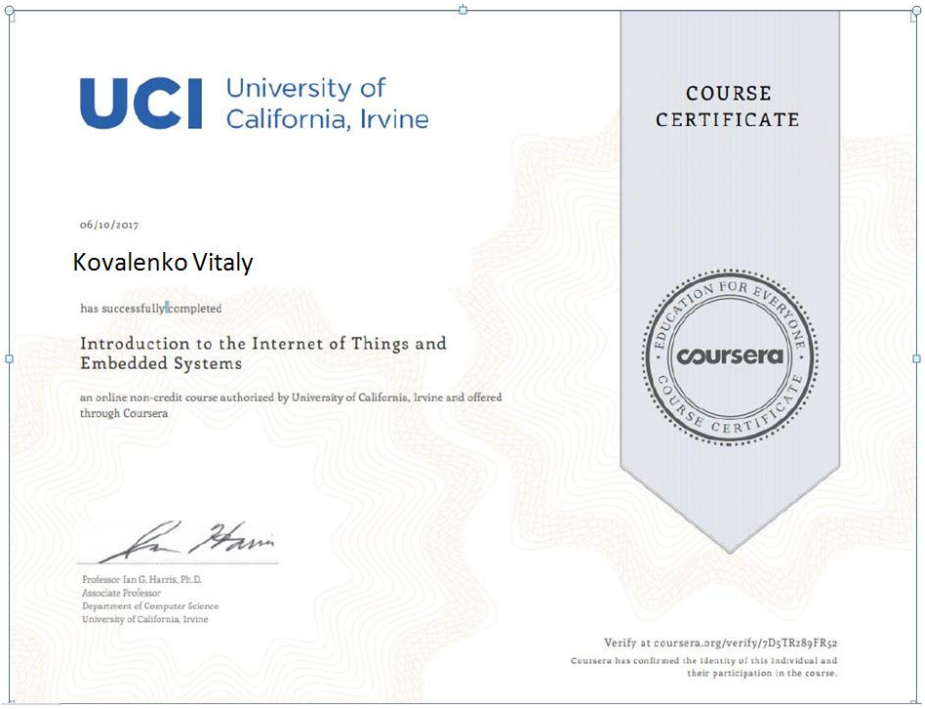


Рисунок 2.5 – Приклад сертифіката в системі Coursera

2.5 Побудова персонального навчального середовища

Перехід від суспільства інформаційного до суспільства знань, яке не потребує накопичення знань про запас, а прагне до безперервної освіти і самонавчання протягом усього життя людини, тягне за собою неминучі зміни в навчанні [4].

Нова якість підготовки майбутніх професіоналів вимагає змін в організації освітнього процесу, оновлення технологій навчання у ВНЗ, зміни форм взаємодії студентів і

викладачів. І акцент в навчанні зміщується в бік самостійної роботи студентів, яка, безумовно, проходить з використанням ресурсів Інтернет і дистанційних освітніх технологій.

На даний момент існує два підходи до впровадження дистанційного навчання: за допомогою LMS і за допомогою хмарних обчислень («програмне забезпечення як послуга»), що утворюють Персональне Навчальне Середовище. Оптимальний варіант в дистанційному навчанні – дотримуватися підходу, який заснований на інтеграції LMS з Персональними Навчальними Середовищами, як студента, так і викладача.

Термін Персональне Навчальне Середовище (англ. Personal Learning Environment (PLE)) – з'явився в західній літературі в 2004 році. Під ПНС розуміється сукупність «соціальних сервісів», програм, інформаційних матеріалів, що забезпечують віддаленому користувачеві (наприклад, студенту і викладачу) комфортні умови навчання. ПНС викладача і студента, використання персональної навчальної мережі, по суті, є показником професійного зростання педагога 21 століття. Відбір інструментів для середовища – справа суто особиста, залежить від цілей автора і чим багатше ПНС, тим більше навчальних можливостей з'являється у користувача Інтернет.

Тому рекомендується його постійно розвивати з урахуванням нових можливостей соціальних сервісів. Персональне навчальне середовище дозволяє студентам краще знати свої цілі в процесі навчання для того, щоб мати можливість вибрати відповідні засоби і включити їх в ПНС. Для цього студенти повинні мати навички саморегульованого навчання і мета-когнітивного мислення.

Студенти повинні створити середовище навчання, яке найкращим чином задовольняє їхні потреби в навчанні, використовуючи їх власний вибір соціальних сервісів. Супровід процесу формування ПНС студента повністю лягає на викладача. На додаток до саморегульованого навчання, ПНС забезпечує студентам активну роль в якості адміністратора власної освіти [5].

Основна проблема в реалізації ПНС студентів – це несформованість регулятивних навичок навчання, тобто недостатність знань про власні способи навчання, методи, які найкращим чином задовольняють їхні потреби. ПНС можна розглядати як засіб підтримки у студентів мета-пізнавального мислення і розвиток навичок саморегульованого навчання. Щоб побудувати ПНС, студенти не тільки повинні знати про свої способи і методи навчання, але і вибрати найбільш відповідні інструменти (соціальні сервіси) для їх освітніх потреб і цілей. Студенти повинні вміти формулювати свої цілі в процесі навчання. Вони повинні мати можливість переглянути свої методи навчання і при необхідності перейти на метод, який краще відповідає їх цілі навчання.

ПНС персоніфікована, тож склад ПНС залежить від різних чинників.

а) Вид функцій, які виконують інструменти:

1) для спілкування (e-mail, чати, форуми і т.д.);

2) для відбору інформаційного матеріалу і його структурування (закладки, агрегатори, Evernote і т.д.);

3) для аналізу матеріалу та пострефлексії (ментальні карти, блоги і т.д.);

4) креативні – для створення контенту (Power Point);

5) контролю і діагностики знань (опитування, тестування).

б) Від зовнішніх чинників – формується з існуючих на даному етапі хмарних обчислень, що відповідають результатам еволюції Web 2.0.

в) Від особистості автора середовища, особистих переваг, комфортності, каналів сприйняття інформації.

г) Цілей автора ПНС – для чого він використовує цю середу:

1) для індивідуального навчання (самовдосконалення, зростання);

2) для навчання інших (для роботи).

Поява нових сервісів в віртуальному просторі з різними функціями, модернізація існуючих, дозволяють створювати більш комфортні умови для навчання студентів і супроводу навчання студентів викладачами.

3 РОЗРОБКА МОДЕЛІ ВЗАЄМОДІЇ СДН З ТРАДИЦІЙНОЮ СИСТЕМОЮ ОСВІТИ

Впровадження LMS у вищий навчальний заклад вирішує не тільки проблему організації дистанційної освіти, а й стає потужною технологічною підтримкою для підвищення освітніх стандартів в усіх напрямках. Інтегруючи e-learning, ВНЗ нарощує конкурентоспроможність, в тому числі за рахунок збільшення кількості випускників і престижності диплома.

3.1 Сценарії використання дистанційних систем освіти в навчальному процесі

Дистанційні системи навчання мають кілька значних переваг та можуть використовуватись наступним чином.

Для школярів проведення відкритих лекцій викладачами університету дозволить зацікавити потенційних абітурієнтів у виборі даного навчального закладу, а також буде сприяти ранньому професійному самовизначенню. А можливість використовувати в процесі навчання матеріали викладачів кращих університетів світу пропонує конкурентну перевагу перед іншими навчальними закладами.

Для студентів заочної форми навчання СДН дає можливість підтримувати комунікації з викладачем не тільки в період сесій, а також спростити процес самоосвіти за рахунок літератури та інших навчальних засобів, що надаються в електронному вигляді. А можливості електронного тестування дозволяють провести контроль даних у великої кількості учнів в короткий термін, що істотно знижує навантаження на викладачів.

Робота аспірантів та здобувачів передбачає повну самоорганізацію. Для цієї категорії СДН дозволяє нівелювати проблеми, пов'язані з щільним академічним графіком аспірантів і їх наукових керівників, а також дозволяє використовувати найбільш великі області знань.

Використання LMS дозволяє молодим вченим і викладачам створювати єдине електронне наукове середовище, яке дає можливість підтримувати міжфакультетські проекти, проводити наукові конференції, забезпечувати функціонування електронної бібліотеки. Також використання в процесі навчання вже існуючих матеріалів дозволяє викладачам найбільш повно розкривати область знань, використовувати вже наявні напрацювання інших учених і в підсумку підбирати найбільш якісну методику навчання.

Завдяки e-learning студенти очної форми навчання отримують нові освітні засоби. У процесі самостійного вивчення матеріалу в домашніх умовах учні можуть розраховувати на

дистанційну допомогу викладача. Це дозволяє підвищити якість засвоєння знань. Також засобами LMS може проводитися оцінка знань, наприклад, тестування, яка надає більш незалежний і неупереджений зріз знань. Додатково в процесі навчання студенти зможуть отримувати сертифікати оцінки якості знань від вчених з інших університетів світу.

Для людей, що здобувають другу вищу освіту, LMS може значно спростити процес освіти, бо, як правило, друга освіта здобувається або без відриву від основної роботи, або в процесі очного навчання. Дистанційна освіта в цьому випадку буде оптимальним рішенням і з точки зору формування навчального графіка, і з позиції якості навчання.

Курси підвищення кваліфікації при університетах актуальні для працюючих людей. З причини короткостроковості цих освітніх програм та їх специфіки, вони повністю можуть бути переведені у віддалений режим за допомогою інтеграції СДН.

Для абітурієнтів впровадження LMS дозволяє організувати доуніверситетську підготовку на дистанційній або частково дистанційній основі. Це особливо актуально для абітурієнтів, що планують вступ до ВНЗ, але проживають в іншому населеному пункті.

3.2 Проблеми інтеграції СДН з традиційною системою освіти

На сьогоднішній день існує величезна кількість всіляких джерел знань. Інтернет же надає практично необмежений доступ до них. Дуже швидкими темпами відбувається розвиток вже існуючих, таких як Coursera, Udacity та ін., і поява нових систем дистанційної освіти. За своєю суттю, обсяг наданих знань у традиційній університетській системі освіти і знання, що надаються в СДН, відрізняються лише підходом до подачі навчального матеріалу та системою оцінювання, хоча зараз вже починають зустрічатися все більш різноманітні засоби оцінки знань в СДН.

Виникає закономірне питання, чому при однаковому або практично однаковому обсязі знань у студента немає можливості будь-яким чином вибирати те, в якій формі він хоче отримати даний обсяг знань? Для того, щоб вирішити дану проблему спочатку потрібно виявити проблеми, пов'язані з інтеграцією сторонніх незалежних від конкретного університету СДН:

1. Проблема визначення якості отриманих в сторонньої СДН знань. Суть проблеми полягає в тому, що університетські курси розробляються за певною методикою і повинні охоплювати необхідний обсяг знань і навчити студента певним навичкам. Тобто при отриманні позитивної оцінки викладач гарантує, що студент засвоїв в певному обсязі необхідні знання з даної дисципліни і даний обсяг знань визначено викладачем. Необхідно розробити механізм, щоб викладач міг формалізувати обсяг необхідних знань для отримання

позитивної оцінки за курс, і в свою чергу студент міг, виходячи із заданого набору формальних параметрів, обирати курс в СДН, який би відповідав даним формальними показниками.

2. Друга проблема – проблема відповідності оцінки, тобто потрібно поставити у відповідність Болонську систему оцінювання і систему оцінювання в конкретній СДН. На щастя, ця проблема не стоїть настільки гостро через те, що практично всі дистанційні системи освіти виставляють оцінку або за 100-бальною, або відсотковою шкалою, що досить просто приводиться до Болонської системи оцінювання.

3. Проблема контролю якості знань. Необхідно яким-небудь чином упевнитися, що та оцінка, яку заявив студент як отриману – відповідає дійсності. Адже при впровадженні даної системи не можна повністю ламати існуючі механізми контролю та заохочення студентів, і система оцінювання повинна бути об'єктивною. На сьогоднішній день переважна більшість систем дистанційної освіти після закінчення навчання видають сертифікат про закінчення курсу та надають засоби верифікації даних сертифікатів, що є гарантією того, що студент отримав необхідний обсяг знань і СДН гарантує те, що студент пройшов всі контролі якості на достатній рівень.

4. Ще одна проблема – зайва формалізованість. Все ж таки процес навчання пов'язаний з певним ступенем свободи в вивченні матеріалу, його подачі і інформації, яка міститься в ньому, тому не можна вимагати 100% відповідності курсу в певній СДН матеріалам, запропонованим для вивчення викладачем. Більш того, може виникнути ситуація, коли матеріали і спосіб подачі пропонований в дистанційній системі освіти виявиться кращим та якіснішим, ніж матеріал, запропонований викладачем. Але вимагаючи 100% збігу, студент не зможе підняти якість отримуваних знань або зможе, але це займе додаткових часових витрат, тому що залишиться необхідність здавати курс за університетськими матеріалами. А викладач, в свою чергу, не зможе підняти рівень якості дисципліни викладання, що позначиться на якості знань студентів в майбутньому, вбиває конкуренцію і розвиток матеріалів викладача. Тому необхідно надати механізм якогось підтвердження викладачем курсів в СДН, які підходять для вивчення дисципліни в разі невідповідності заданих викладачем формальних ознак.

5. Остання проблема – та, що інтеграція СДН не повинна значно збільшувати навантаження як на викладача, так і на студента і співробітника деканату, тож система інтеграції повинна бути легкою в роботі та не мати ускладнений та важкий для роботи та розуміння інтерфейс.

Виходячи з вищеописаних проблем, необхідно розробити концепцію підходу, який може бути імплементований в певній автоматизованій системі і здатний вирішувати описані вище проблеми і завдання.

3.3 Розробка критеріїв відповідності навчальних матеріалів у СДН навчальній програмі

Нормативний зміст підготовки фахівців в освітньо-професійних та освітньо-наукових програмах (ОПП та ОНП) має формулюватися у термінах компетентностей та результатів навчання, досягнення яких потребує відповідного переліку навчальних дисциплін, який визначає вищий навчальний заклад при формуванні навчальних планів. У нормативних документах компетентність визначається як здатність особи до виконання певного виду діяльності, що виражається через знання, розуміння, уміння, цінності, інші особисті якості. При цьому визначається кваліфікаційний рівень, тобто структурна одиниця Національної рамки кваліфікацій, що визначається певною сукупністю компетентностей, які є типовими для кваліфікацій даного рівня. Виходячи з цього результати навчання (РН) характеризуються сукупністю: знань, умінь, комунікаціями, автономністю і відповідальністю. Компетентності для відповідної спеціальності та рівня вищої освіти знаходять своє відображення при формуванні ОПП та навчального плану, який відображає спосіб набуття результатів навчання через вивчення відповідних навчальних дисциплін.

Дистанційне навчання – це, в першу чергу, взаємодія особи, яка навчається, з викладачем на відстані (дистанційно). При цьому таке навчання відображає практично усі властиві навчальному процесу компоненти (методи, цілі, організаційні форми, зміст, а часто і засоби навчання) та реалізовується специфічними засобами телекомунікаційних технологій, які передбачають інтерактивність процесу навчання. Зовсім недавно все більше стали звертати увагу на компетенції, які отримує особа під час навчання. Саме із компетенцій складається майбутній фахівець певної галузі. Тому дуже важливо використовувати компетентністний підхід у дистанційному навчанні.

При компетентністному підході накопичується та осмислюється досвід рішення не навчальних, а життєвих завдань. Основним результатом навчання будуть не знання, уміння і навички, а осмислений досвід професійної діяльності. Життєвий досвід формується планомірно. Оцінюється накопичений багаж дидактичних одиниць та здатність застосувати його у різних ситуаціях. Саме тому методи і форми навчання мають бути підпорядковані не навчальному змісту, а використовуватися як самостійні засоби досягнення певних педагогічних цілей.

Отже, виходячи з вищеописаного, для того, щоб було можливо зарахувати пройдений студентом курс в одній із СДН, необхідної умовою є те, що цей курс має відповідати заданому набору критеріїв, а саме:

- назва курсу має відповідати дисципліні;
- об’єм курсу має відповідати об’єму дисципліни;
- структура курсу має відповідати структурі дисципліни, тобто мають бути різні типи задач, що використовуються протягом вивчення курсу в СДН;
- тематична наповненість курсу – тематика курсу має в певній мірі співпадати с тематичним наповненням дисципліни;
- має застосовувати компетенційний підхід – тобто, після проходження курсу в СДН студент має здобути компетенції, що були визначені як необхідні для даної дисципліни.

Також слід зазначити, що стовідсоткового співпадіння за всіма цими ознаками досягти занадто важко, якщо не неможливо. Тому додатково є необхідним визначити такий параметр, як критерій відповідності – мінімально достатній рівень співпадіння пройденого студентом курсу в СДН та дисципліни, що вивчається у ВНЗ стаціонарно за наведеними ознаками.

Отже, для успішної розробки системи інтеграції СДН необхідно розробити декілька формалізованих шаблонів, що мають заповнюватись студентом та викладачем, а також дерево прийняття рішень, що буде вирішувати чи відповідає певний курс дисципліні.

Шаблон, що буде заповнюватись викладачем, має визначати набір критеріїв, яким має відповідати курс для того аби він був зарахований в якості відповідного до певної навчальної дисципліни. В загальному випадку цей шаблон може мати наступний вигляд.

Таблиця 3.1 – Шаблон критеріїв відповідності, що заповнюється викладачем

Назва дисципліни	<Назва курсу>	
Об’єм дисципліни	Лекції	<Кількість лекційних годин>
	Лабораторні роботи	<Кількість годин на лабораторні роботи>
	Практичні роботи	<Кількість годин на практичні роботи>
	Самостійне навчання	<Кількість годин для самостійного вивчення>
Тематичний склад	<Тема 1> <Тема 2>	
Компетенції	<Компетенція 1> <Компетенція 2> <Компетенція 3>	
Знання, уміння	<Знання 1> <Знання 2> <Уміння 1> <Уміння 2>	
Мінімальний поріг співпадання критеріїв відповідності	<Мінімальний поріг відповідності у відсотках>	

Заповнивши цей нескладний шаблон, викладач визначає ключові особливості дисципліни, на які можуть спиратися студенти при виборі курсу в будь-якій СДН. Заповнений викладачем екземпляр цього шаблону має бути постійно у вільному доступі для студентів, аби ті мали змогу в будь-який час перевірити відповідність обраного курсу.

На перший погляд цей шаблон виглядає нібито дещо перевантаженим, але слід розуміти, що у випадку інтеграції даної системи з іншими університетськими системами, від викладача може потребуватись лише часткове заповнення даного шаблону, оскільки інформація, наприклад, про назву та об'єм, вже знаходиться у робочій програмі дисципліни або у семестровій програмі напрямку студента.

Складання загального шаблону для студента є дещо складнішим, але матиме схожий вигляд з шаблоном викладача.

Таблиця 3.2 – Шаблон, що заповнюється студентом

Назва дисципліни	<Назва курсу>	
Об'єм дисципліни	Лекції	<Кількість лекційних годин>
	Лабораторні роботи	<Кількість годин на лабораторні роботи>
	Практичні роботи	<Кількість годин на практичні роботи>
	Самостійне навчання	<Кількість годин для самостійного вивчення>
Тематичний склад	<Тема 1> <Тема 2>	
Компетенції	<Чи здобули ві компетенцію 1> <Чи здобули ві компетенцію 2> <Чи здобули ві компетенцію 3>	
Знання, уміння	<Чи здобули ви знання 1> <Чи здобули ви знання 2> <Чи здобули ви уміння 1> <Чи здобули ви уміння 2>	
Здобута оцінка	<Здобута оцінка>	
Посилання на пройдений курс	<Посилання на web сторінку пройденого курсу>	
Посилання на отриманий сертифікат	<Посилання на отриманий сертифікат>	

Основне завдання цього шаблону – це визначення відповідності вивченого студентом курсу тим критеріям, які було обрано викладачем, як ключові. Основна складність при цьому – це об'єктивність визначення відповідності здобутих компетенцій, оскільки невелика кількість курсів в СДН визначає компетенції, які будуть здобуті студентом, а також формулювання цих компетенцій може дещо відрізнятись. Для вирішення цієї проблеми є кілька підходів:

1. Студент відповідає на питання про компетенції у форматі тестування, тобто, викладач складає невеличкий тест для виявлення відповідності знань студента визначеним компетенціям.

2. Студент відповідає на питання у форматі «Чи освоїли ви компетенцію 1?» з відповідями «Так» або «Ні». Звісно це не надто об’єктивний засіб визначення рівня знань, але слід додати, що студент також має надати посилання на вивчений ним курс та посилання на отриманий ним сертифікат, що забезпечить достовірність оцінки, а також усвідомлення того, що в будь-який момент викладач може зайти на сторінку курсу та перевірити його тематичну та компетенційну відповідність. І у випадку виявлення обману оцінка може буде анульована.

3. Третій підхід – це розробка системи з методами машинного навчання, штучного інтелекту та засобів кросмовного аналізу для автоматичного визначення компетенцій, що надає курс, та їх відповідності ознакам сформованих викладачем.

З цих трьох підходів найбільш відповідним здається другий варіант, оскільки він не потребує занадто великих коштів та часу на розробку, як третій варіант, і не потребує додаткових дій від викладача (складання тестів для визначення рівня відповідності студента компетенціям), та є достатньо об’єктивним. Третій підхід є дуже цікавим, але його слід залишити на майбутнє в якості можливих поліпшень розробленої системи, оскільки час та кошти на його реалізацію є занадто високими для системи, робота якої може бути змінена в подальшому у зв’язку з тим, що не була протестована у реальному використанні. Приклад прототипу реалізованої системи, що використовує другий підхід, детально описано у четвертому розділі.

При наявності двох описаних вище шаблонів та визначених на початку проблем стає можливим розробка дерева прийняття рішень, після розробки якого можна переходити до наступного етапу – формування вимог до реалізації заданої системи. Дерево прийняття рішень є інструментом підтримки прийняття рішень, який складається з графічного представлення наявних альтернатив, що генеруються з початкового рішення. Однією з найбільших переваг дерева рішень є можливість трансформації або декомпозиції однієї складної задачі в кілька більш простих підзадач. Рекурсивно нові сформульовані підзадачі знову розбиваються на ще більш прості підзадачі [6].

Для формування графічного представлення дерева рішень, як правило, використовуються лінії для ідентифікації рішень (наприклад, “так” або “ні”) і вузли для визначення питань, по яких потрібно прийняти рішення. Кожна з гілок, утворених лініями і вузлами, закінчується свого роду листом, який ідентифікує найбільш ймовірний результат з послідовності рішень.

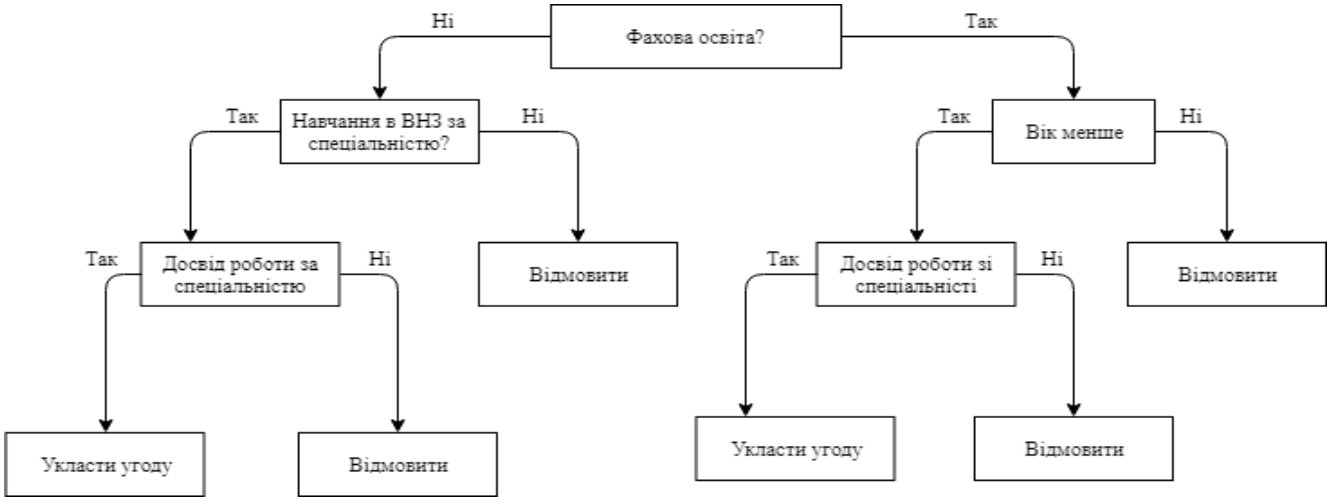


Рисунок 3.1 – Приклад дерева ухвалення рішень

Структура дерева містить такі елементи: «листя» і «гілки». На ребрах («гілках») дерева ухвалення рішення записані атрибути, від яких залежить цільова функція, в «листі» записані значення цільової функції, а в інших вузлах — атрибути, за якими розрізняються випадки. Щоб класифікувати новий випадок, треба спуститися по дереву до листа і видати відповідне значення. Подібні дерева рішень широко використовуються в інтелектуальному аналізі даних. Мета полягає в тому, щоб створити модель, яка прогнозує значення цільової змінної на основі декількох змінних на вході [6].

Кожен лист являє собою значення цільової змінної, зміненої в ході руху від кореня до листа. Кожен внутрішній вузол відповідає одній з вхідних змінних. Дерево може бути також «вивчено» поділом вихідних наборів змінних на підмножини, що засновані на тестуванні значень атрибутів. Це процес, який повторюється на кожній з отриманих підмножин. Рекурсія завершується тоді, коли підмножина у вузлі має ті ж значення цільової змінної. Таким чином, воно не додає цінності для прогнозувань. Процес, що йде «згори донизу», індукція дерев рішень (TDIDT), є прикладом поглинаючого «жадібного» алгоритму [6]. І на сьогодні є найбільш поширеною стратегією дерев рішень для даних, але це не єдина можлива стратегія. В інтелектуальному аналізі даних дерева рішень можуть бути використані як математичні та обчислювальні методи, щоб допомогти описати, класифікувати і узагальнити набір даних, які можуть бути записані таким чином:

$$(x, Y) = (x_1, x_2, x_3 \dots x_k, Y)\ , \qquad (3.1)$$

де залежна змінна Y є цільовою змінною, яку необхідно проаналізувати, класифікувати й узагальнити. Вектор x складається з вхідних змінних x1, x2, x3, тощо, які використовуються для виконання цього завдання.

Переваги методу:

- простий в розумінні та інтерпретації. Люди здатні інтерпретувати результати моделі дерева прийняття рішень після короткого пояснення;
- не потребує підготовки даних (інші техніки вимагають нормалізації даних, додавання фіктивних змінних, а також видалення пропущених даних);
- здатний працювати як з категоріальними, так і з інтервальними змінними (інші методи працюють лише з тими даними, де присутній лише один тип змінних);
- використовує модель «білого ящика» – якщо певна ситуація спостерігається в моделі, то її можна пояснити за допомогою булевої логіки (прикладом «чорного ящика» може бути штучна нейронна мережа, так як результати даної моделі піддаються поясненню важко);
- дозволяє оцінити модель за допомогою статистичних тестів – це дає можливість оцінити надійність моделі;
- є надійним методом – метод добре працює навіть в тому випадку, коли були порушені початкові припущення, включені в модель;
- дозволяє працювати з великим об'ємом інформації без спеціальних підготовчих процедур – даний метод не вимагає спеціального обладнання для роботи з великими базами даних.

Недоліки методу:

- проблема отримання оптимального дерева рішень є NP-повною з точки зору деяких аспектів оптимальності навіть для простих завдань (таким чином, практичне застосування алгоритму дерев рішень засноване на евристичних алгоритмах, таких як алгоритм «жадібності», де єдине оптимальне рішення вибирається локально в кожному вузлі – такі алгоритми не можуть забезпечити оптимальність всього дерева в цілому);
- ті, хто вивчає метод дерева прийняття рішень, можуть створювати занадто складні конструкції, які недостатньо повно представляють дані – дана проблема називається перенавчанням.

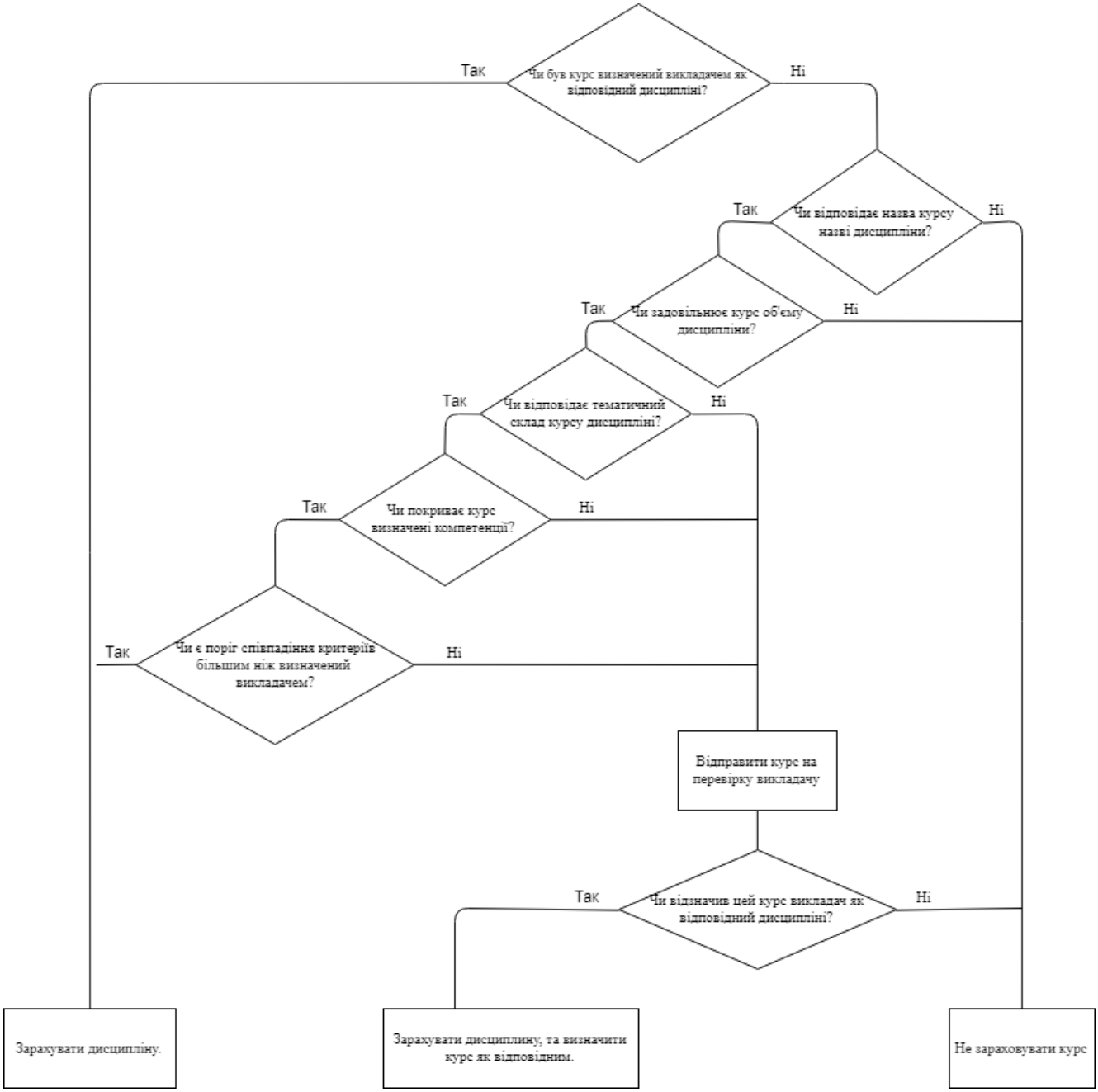


Рисунок 3.2 – Дерево прийняття рішень

На рисунку вище наведено розроблену схему прийняття рішень, при використанні якої стає можливою інтеграція різноманітних систем дистанційного навчання для зарахування різноманітних дисциплін навчальної програми університету.

3.4 Формування вимог до системи інтеграції СДН

Таким чином, після розробки шаблонів та дерева прийняття рішень, є можливим сформулювати вимоги до системи, що треба розробити на першій ітерації.

Для опису вимог найбільш швидким та простим є метод створення історії користувача.

Історії користувача - швидкий спосіб оперування вимогами користувача без необхідності застосування занадто формалізованих документів та виконання адміністративних задач, пов'язаних з опрацюванням цих документів. Наміром, з яким використовують історії користувача, є швидше та менш накладне реагування на швидко змінювані вимоги реального світу [7].

Історії користувача – це неформальний опис вимог до тих пір, поки відсутні відповідні тести прийнятності. Перед тим як реалізовувати історію користувача, відповідна процедура прийняття має бути написана користувачем, що тестуванням чи іншим чином визначає чи задоволені вимоги історії користувача. Деяка формалізація відбувається, коли розробник приймає історію користувача та відповідну процедуру прийнятності.

Історії користувача мають деякі переваги перед іншими засобами.

а) Вони короткі. Вони надають невеликі порції вимог, які можуть бути реалізовані за кілька днів чи тижнів.

б) Дозволяють розробнику та представнику клієнта обговорювати вимоги протягом життєвого циклу проекту.

в) Потребують зовсім мало обслуговування.

г) Дозволяють ближчий контакт зі споживачем.

д) Дозволяють розбивати проект на маленькі кроки.

е) Підходять до проектів, в яких вимоги нестійкі чи малозрозумілі. Ітерації розкриття вимог спрямовують процес вдосконалення продукту.

є) Полегшують оцінку складності розробки.

ж) Завдяки ближчому контакту з замовником реалізуються найважливіші для нього частини проекту.

Тож для системи інтеграції СДН з традиційною системою освіти були створені наступні історії користувача:

Для користувацької ролі «Студент» були визначені наступні історії:

а) Як студент, я хочу мати можливість увійти в систему для її використання.

б) Як студент, я хочу мати можливість бачити перелік дисциплін, що вивчаються у поточному семестрі для того, щоб розуміти, які курси я хочу вивчити в СДН.

в) Як студент, я хочу мати можливість продивитись сформовані викладачем формальні критерії для певної дисципліни, аби мати можливість обрати найбільш відповідний до них курс в СДН.

г) Як студент, я хочу мати можливість додати пройдені мною курси до системи, аби мати можливість вивчати дисципліни в СДН.

д) Як студент, я хочу мати можливість переглядати статус доданих мною дисциплін, щоб розуміти поточний статус дисциплін, що треба вивчити в поточному семестрі.

ж) Як студент, я хочу мати можливість переглядати підтверджені викладачем курси, аби мати змогу обрати курси, що гарантовано відповідають навчальній програмі.

Для ролі «Співробітник деканату» були визначені наступні історії:

а) Як співробітник деканату, я хочу мати можливість завантажувати семестровий план навчання для того, щоб студенти мали можливість вивчати дисципліни в СДН.

б) Як співробітник деканату, я хочу мати можливість переглядати перелік дисциплін, що були зараховані для студентів, аби мати можливість перенести ці дані у документацію університету.

в) Як співробітник деканату, я хочу мати можливість переглядати додані студентом сертифікати у систему для здійснення перевірки та верифікації пройдених студентом дисциплін.

Для користувацької ролі «Викладач» були визначені наступні історії:

а) Як викладач, я хочу мати можливість додавати критерії для дисциплін, що викладаю, аби студенти мали можливість пройти цю дисципліну в СДН.

б) Як викладач, я хочу мати можливість підтверджувати різноманітні курси, що були відправлені студентами на верифікацію для того, щоб підвищити рівень навчальних матеріалів та не обмежувати студентів в їх можливостях.

в) Як викладач, я хочу мати можливість підтверджувати курси, як ті, що підходять для вивчення дисципліни, аби в подальшому студенти мали можливість проходити ці курси в якості навчання дисципліни.

3.5 Додатковий функціонал системи

В подальшому функціонал даної системи може бути дещо розширений наступними можливостями:

- додавання машинного навчання, штучного інтелекту та кросмовного аналізу для доданих студентом курсів, щоб максимально зняти навантаження з викладача;
- додавання інтеграції розробленої системи з LMS Moodle для автоматичного завантаження переліку компетенцій, знань та умінь відповідних дисципліні;
- додавання функціоналу поштової розсилки, щоб викладач та студент могли вчасно реагувати на дії один одного в системі;

- додавання інтеграції з різноманітними СДН для автоматичної верифікації доданих студентами сертифікатів;
- інтеграція з іншими університетськими системами для зменшення одноманітних дій користувачів системи;
- розробка мобільних додатків для найбільш комфортного використання системи всіма учасниками.

4 ПРИКЛАД ПРОГРАМНОЇ РЕАЛІЗАЦІЇ СИСТЕМИ ІНТЕГРАЦІЇ СДН З ТРАДИЦІЙНОЮ НАВЧАЛЬНОЮ СИСТЕМОЮ

4.1 Вибір технології та середовища розробки

На даний момент існує безліч варіантів для написання серверної частини клієнт-серверного додатка. Найбільш популярними в цій сфері є мови Java, C#, PHP і Python.

PHP і Python, не дивлячись на свої переваги у відносній простоті написання коду, втрачають перевагу перед Java або C# в швидкості роботи. З решти двох варіантів, була обрана Java з використанням Spring Framework. Вибір припав на Java з причин, викладених нижче.

Java – об'єктно-орієнтована мова програмування, розроблена компанією Sun Microsystems (в подальшому придбаної компанією Oracle). Програми Java зазвичай транслюються в спеціальний байт-код, тому вони можуть працювати на будь-якій віртуальній Java-машині незалежно від комп'ютерної архітектури.

Перевагою подібного способу виконання програм є повна незалежність байт-коду від операційної системи і обладнання, що дозволяє виконувати Java-додатки на будь-якому пристрої, для якого існує відповідна віртуальна машина. Іншою важливою особливістю технології Java є гнучка система безпеки, в рамках якої виконання програми повністю контролюється віртуальною машиною. Будь-які операції, які перевищують встановлені повноваження програми (наприклад, спроба несанкціонованого доступу до даних або з'єднання з іншим комп'ютером), викликають негайне переривання.

Часто до недоліків концепції віртуальної машини відносять зниження продуктивності. Ряд удосконалень збільшив швидкість виконання програм на Java:

- застосування технології трансляції байт-коду в машинний код безпосередньо під час роботи програми (JIT-технологія) з можливістю збереження версій класу в машинному коді;
- широке використання платформенно-орієнтованого коду (native-код) в стандартних бібліотеках;
- апаратні засоби, що забезпечують прискорену обробку байт-коду (наприклад, технологія Jazelle, підтримувана деякими процесорами фірми ARM).

Основні можливості Java:

- автоматичне керування пам'яттю;
- розширені можливості обробки виняткових ситуацій;

- багатий набір засобів фільтрації введення-виведення;
- набір стандартних колекцій: масив, список, стек і т. п .;
- наявність простих засобів створення мережевих додатків (у тому числі з використанням протоколу RMI);
- наявність класів, що дозволяють виконувати HTTP-запити і обробляти відповіді;
- вбудовані в мову засоби створення багатопоточних додатків;
- уніфікований доступ до баз даних: на рівні окремих SQL-запитів – на основі JDBC, SQLJ; на рівні концепції об'єктів, що володіють здатністю до зберігання в базі даних – на основі Java Data Objects і Java Persistence API;
- підтримка лямбд, замикань, вбудовані можливості функціонального програмування (з 1.8).

Одна з головних переваг мови Java – її незалежність від платформи, на якій виконуються програми. Таким чином, один і той же код можна запускати під управлінням операційних систем Windows, Linux, FreeBSD, Solaris, Apple Mac і ін. Це стає дуже важливим, коли програми завантажуються за допомогою глобальної мережі Інтернет і використовуються на різних платформах.

Java – повністю об'єктно-орієнтована мова. Всі сутності в мові Java є об'єктами, за винятком небагатьох основних типів (primitive types), наприклад, чисел. Свого часу об'єктно-орієнтоване програмування (ООП) замінило структурне програмування.

Вся справа в тому, що розробниками мови Java з компанії Sun був проведений фундаментальний аналіз програм на мові C++. Аналізувалися "вузькі місця" вихідного коду, які і призводять до появи помилок, що важко виявити. Тому було прийнято рішення проектувати мову Java з урахуванням можливості створювати програми, в яких були б приховані найбільш поширені помилки.

Spring (рис.4.1) – це вільно поширюваний фреймворк, створений Родом Джонсоном (Rod Johnson) і описаний в його книзі «Expert One-on-One: J2EE Design and Development». Він був створений з метою усунути складнощі розробки корпоративних додатків і зробити можливим використання простих компонентів JavaBean для досягнення всього того, що раніше було можливим тільки з використанням EJB. Однак область застосування Spring не обмежується розробкою програмних компонентів, що виконуються на стороні сервера. Будь-який Java-додаток може використовувати переваги фреймворка в плані простоти, тестованості і слабкого зв’язку.

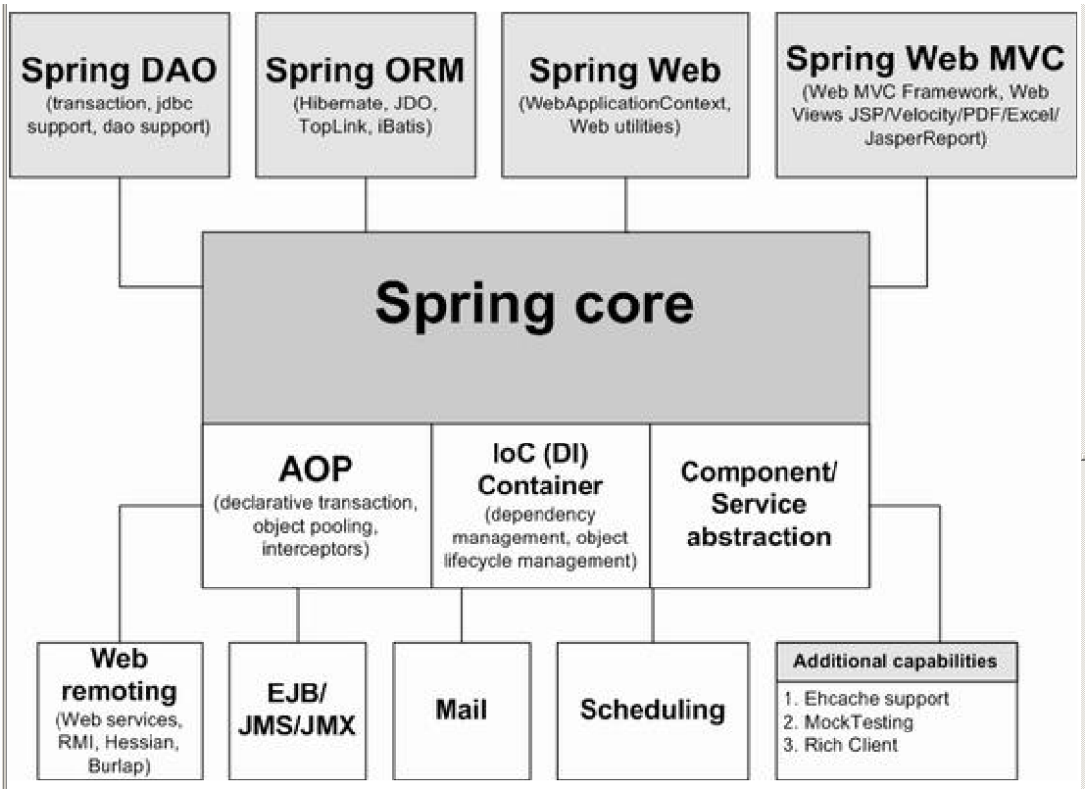


Рисунок 4.1 – Архітектура Spring Framework

У своєму прагненні спростити складності, пов'язані з розробкою на мові Java, фреймворк Spring використовує чотири ключові стратегії:

- легковажність завдяки застосуванню простих Java-об'єктів (POJO);
- слабке зв'язування за допомогою впровадження залежностей і орієнтованості на інтерфейси;
- декларативне програмування через аспекти і загальноприйняті угоди;
- зменшення обсягу типового коду через аспекти та шаблони.

Практично всі можливості фреймворка Spring сягають корінням в ці стратегії. В основі Spring лежить патерн Inversion of control. Основна ідея цього патерну полягає в усуненні залежності компонентів або класів додатків від конкретних реалізацій допоміжних інтерфейсів і делегування повноважень з управління створенням потрібних реалізацій IoC контейнеру. IoC контейнер відповідає за створення потрібної реалізації Product для Consumer. При використанні класу Consumer в інших проектах є можливість замінити реалізацію інтерфейсу Product на більш відповідну, не вносячи змін в код.

Основні переваги IoC контейнерів:

- управління залежностями;
- спрощення повторного використання класів або компонентів;
- спрощення unit-тестування;

- більш "чистий" код (класи більше не займаються ініціалізацією допоміжних об'єктів. Не варто, звичайно "перегинати палицю", керуючи створенням абсолютно всіх об'єктів через IoC. В IoC контейнер найкраще виносити ті інтерфейси, реалізація яких може бути змінена в поточному проекті або в майбутніх проектах).

Spring набагато більше, ніж звичайний IoC контейнер. Framework спрощує розробку J2EE проектів, реалізуючи низькорівневі і найбільш часто використовувані частини корпоративного додатку.

Spring Web MVC – розширювана платформа MVC для створення веб-додатків. Спираючись на шаблон модель – представлення-контролер, фреймворк Spring MVC допомагає будувати веб-додатки, так само гнучкі і слабко пов'язані, як сам фреймворк Spring Framework.

В якості СУБД була обрана MySQL Server.

MySQL – вільна система управління базами даних (СУБД). MySQL є власністю компанії Oracle Corporation, що отримала її разом з купівлею Sun Microsystems, що здійснює розробку і підтримку програми. Поширюється під GNU General Public License або під власною комерційною ліцензією.

За останній час MySQL значно просунулася вперед. Уже зараз ця СУБД стала фактичним стандартом для Інтернет-додатків (веб-магазини, складні сайти, інформаційні портали). Під відкритими системами (Linux, FreeBSD) і навіть під закритими, де немає продукції Microsoft (Sun Solaris, різні комерційні версії Unix), ця СУБД лідирує вже зараз. MySQL є Open Source Software.

Open Source означає, що тексти відкриті для читання і редагування всім бажаючим. Будь-хто може завантажити MySQL з Internet і використовувати його абсолютно безкоштовно. Будь-який бажаючий може вивчати вихідний текст і змінювати його на свій розсуд. MySQL використовує ліцензію GPL (GNU General Public License), щоб визначити те, що можно робити з програмним забезпеченням в різних ситуаціях. Якщо треба впровадити MySQL в комерційну прикладну програму, можливо купити комерційно запатентовану версію у авторів.

Гнучкість СУБД MySQL забезпечується підтримкою великої кількості типів таблиць: користувачі можуть вибрати як таблиці типу MyISAM, що підтримують повнотекстовий пошук, так і таблиці InnoDB, що підтримують транзакції на рівні окремих записів.

Для вирішення задачі об’єктно-реляційного відображення був обраний Hibernate, як найпопулярніша реалізація специфікації JPA. Метою Hibernate є звільнення розробника від значних типових завдань із програмування взаємодії з базою даних. Розробник може використовувати Hibernate як при розробці з нуля, так і для вже існуючої бази даних.

Hibernate піклується про зв'язок класів з таблицями бази даних (і типів даних мови програмування із типами даних SQL), і надає засоби автоматичної побудови SQL запитів й зчитування/запису даних, і може значно зменшити час розробки, який зазвичай витрачається на ручне написання типового SQL і JDBC коду. Hibernate генерує SQL виклики і звільняє розробника від ручної обробки результуючого набору даних, конвертації об'єктів і забезпечення сумісності із різними базами даних.

Hibernate забезпечує прозору підтримку збереження даних, тобто їхньої персистентності (англ. persistence) для «POJO»-об'єктів, для звичайних Java-об'єктів; єдина сувора вимога до класу, що зберігається — конструктор за замовчуванням.

Mapping (зіставлення, буквально — картування) Java класів з таблицями бази даних здійснюється за допомогою конфігураційних XML файлів або Java анотацій. Hibernate може використовувати файл XML або анотації для підтримки схеми бази даних.

Забезпечуються можливості з організації відношення між класами «один-до-багатьох» і «багато-до-багатьох». На додаток до управління зв'язками між об'єктами, Hibernate також може керувати рефлексивними асоціаціями, де об'єкт має зв'язок «один-до-багатьох» з іншими примірниками свого власного типу даних.

Hibernate підтримує відображення користувацьких типів значень. Це робить можливим такі сценарії:

- перевизначення типу за замовчуванням SQL, який Hibernate вибирає при відображенні стовпчика властивості;
- картування перераховуваного типу Java до колонок БД, так ніби вони є звичайними властивостями;
- картування однієї властивості в декілька колонок.

Hibernate забезпечує використання SQL-подібної мови Hibernate Query Language (HQL), яка дозволяє виконувати SQL-подібні запити, записані поряд з об'єктами даних Hibernate. Запити критеріїв надаються як об'єктно-орієнтована альтернатива до HQL.

Hibernate може використовуватись як у самостійних програмах Java, так і в програмах Java EE, що виконуються на сервері (наприклад, сервлети чи EJB session beans). Також він може включатись як додаткова можливість до інших мов програмування.

Для розробки клієнтської частини було прийнято рішення використовувати WEB клієнт, написаний на HTML з використанням адаптивної верстки і фреймворка Twitter Bootstrap.

Адаптивна верстка (Responsive web design або RWD) – підхід до створення дизайну, при якому сайт розробляється з розрахунком на те, щоб забезпечити найбільш просте його

використання: зручний перегляд сайту з мінімумом змін розміру і зайвих прокручувань – на найширшому спектрі пристроїв. Адаптивний дизайн володіє наступними особливостями.

- При верстці адаптивного дизайну використовуються виключно HTML і CSS - без підключення JavaScript для визначення «адаптивності» елементів дизайну.
- Адаптивна верстка визначає, як будуть виглядати елементи сайту на різних пристроях; проте ці елементи не ховаються/не замінюють один, а їх поведінка, так само як і виконувані ними функції, не змінюються.
- Три основних принципи адаптивного дизайну: розташування всіх елементів в рамках модульної сітки; всі елементи верстки та медіа-файли (в т.ч. зображення) є «гумовими» (flexible) – їх розміри залежать від розміру екрану; робота з Media queries – модулем CSS3, що дозволяє задавати різні стилі (або навіть таблиці стилів) в залежності від роздільної здатності екрану, його розмірів і інших характеристик.
- Адаптивна розмітка (adaptive layout) полягає в тому, що на сайті створюється кілька стилів, варіантів розташування елементів на модульній сітці і кілька варіантів стилів елементів дизайну. Ці варіанти змінюють один одного при зміні розмірів екрану, при цьому утворюються якісь точки переходу між різними видами розмітки/стилів сайту.
- Адаптивний дизайн не має на увазі роботи з об'єктною моделлю елементів на сторінці, не має на увазі зміну ієрархії/вкладеності блоків і об'єктів при зміні виду розмітки.

Twitter Bootstrap – це фреймворк для створення сучасних, кросбраузерних і стандартизованих інтерфейсів. Продумана структура коду HTML, JavaScript і CSS надає можливість створювати безліч різноманітних елементів інтерфейсу і сітку сайту.

Основні інструменти Bootstrap:

- сітки – заздалегідь задані розміри колонок, які можна відразу ж використовувати;
- шаблони – фіксований або гумовий шаблон документа;
- типографіка – опису шрифтів, визначення деяких класів для шрифтів таких як код, цитати і т.д.;
- медіа – представляє можливості управління зображеннями і відео;
- таблиці – засоби оформлення таблиць, аж до додавання функціональності для забезпечення можливості сортування;
- форми – класи для оформлення не тільки форм, але і деяких подій, що відбуваються з ними;
- навігація – класи оформлення для вкладок, сторінок, меню і панелей інструментів;
- алерт – оформлення діалогових вікон, підказок та спливаючих вікон.

Перевагою всіх обраних технологій є можливість розгорнути їх на будь-якому сервері з будь-якою ОС, яка підтримує Java і MySQL. На сьогоднішній день всі основні сучасні ОС (Windows, OS X, Linux) підтримують обрані технології.

4.2 Опис розробленого прототипу системи інтеграції СДН

При розробці прототипу системи інтеграції СДН була створена наступна схема БД (рис.4.2):

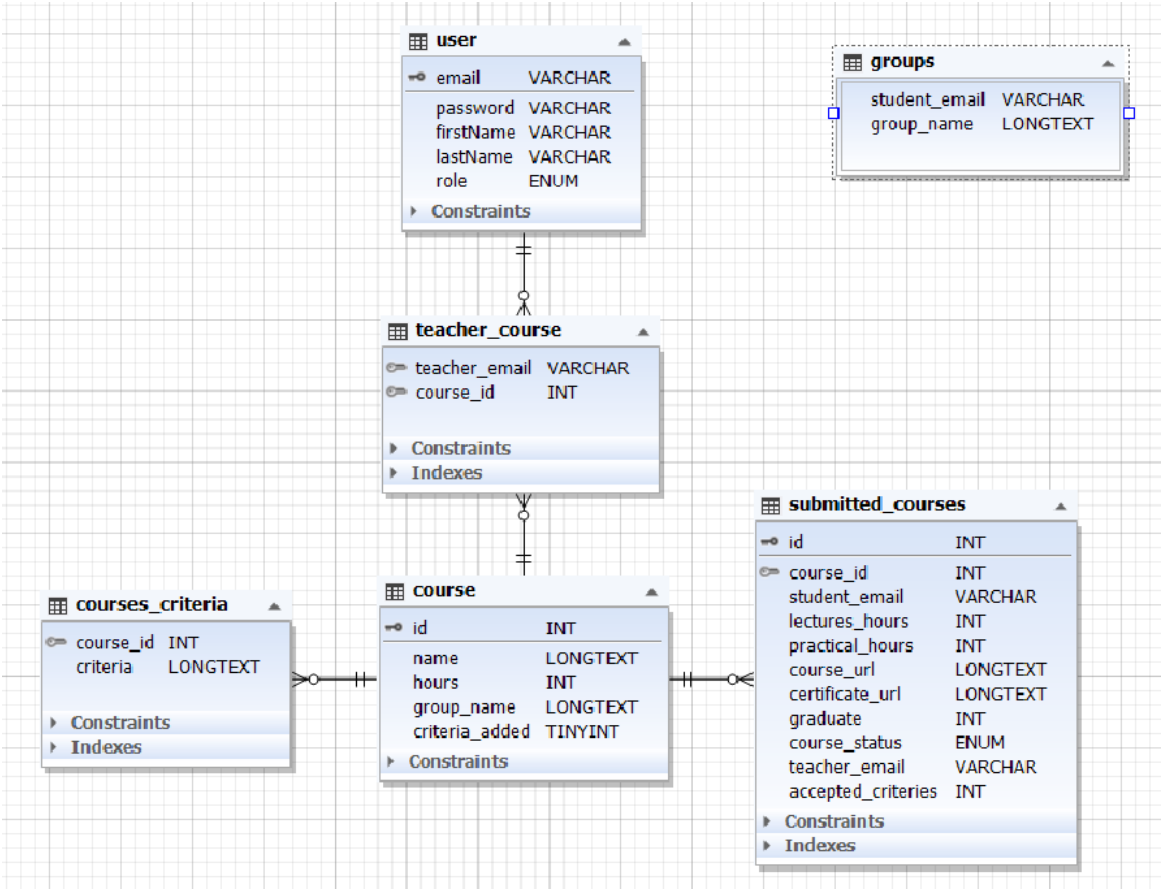


Рисунок 4.2 – Схема розробленої бази даних

На розробленій схемі видно, що йде оперування кількома сутностями: user – на цьому етапі розробки програми, функціонал реєстрації не реалізовувався, оскільки в сервісів сайту вже є інтеграція з поштовим клієнтом від Google, цю інтеграцію лише потрібно перенести в розроблену систему у випадку її реального використання; course – дисципліни, додані у систему співробітником деканату та заповнені викладачем; submitted_course – курси, що були додані студентами як вже перевірені, або ті, що очікують на перевірку.

Також був розроблений веб-клієнт для роботи у системі. Користувач системи може мати одну з трьох ролей: Student – студент, Teacher – викладач або Manager – співробітник

деканату. Авторизація в системі виконується за допомогою сторінки входу та модулю Spring Security, що входить до складу Spring Framework (рис.4.3).

Рисунок 4.3 – Сторінка авторизації користувача

Після цього відбувається переадресація користувача на сторінку відповідно до його ролі в системі, де користувач може робити певні дії відповідно до своєї ролі. Слід зазначити, що інтерфейс є «гумовим» тобто клієнт може однаково зручно користуватись системою з різних пристроїв (ноутбук, стаціонарний комп'ютер, планшет або смартфон) (рис.4.4).

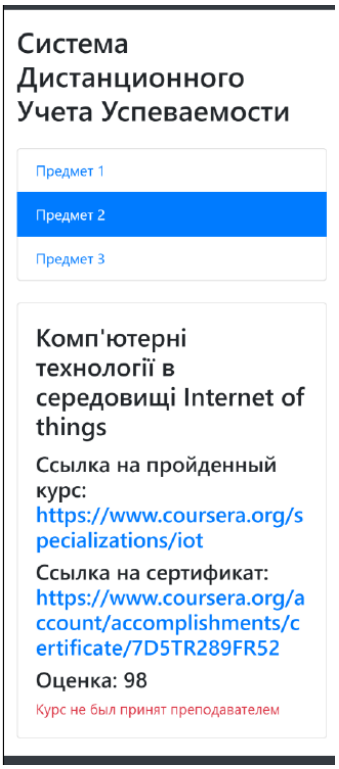


Рисунок 4.4 – Відображення мобільної версії веб-клієнта.

Для співробітника деканату в розробленій системі доступний такий функціонал – завантаження робочої програми у систему для того, щоб система могла частково заповнити критерії відповідності дисципліни та перегляд курсів, що були автоматично зараховані системою або викладачем.

Для завантаження курсу необхідно на сторінці після авторизації обрати файл з робочою програмою певної групи та натиснути кнопку завантаження програми (рис.4.5).

У випадку успішного завантаження, про це буде виведено відповідне повідомлення. Алгоритм, що виконує граматичний розбір завантаженої таблиці з робочою програмою, наведено у лістингу нижче.

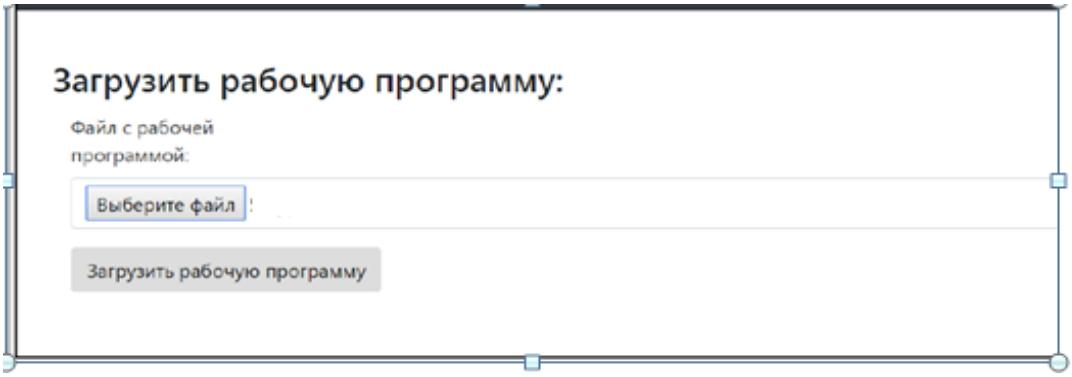


Рисунок 4.5 – Screenshot завантаження робочої програми

Лістинг 4.1 – Код алгоритму граматичного розбору

```
public List<Course> parseProgram(MultipartFile file) {
    Workbook workbook = getWorkbookFromFile(file);
    if (Objects.isNull(workbook)) {
        return Collections.emptyList();
    }
    FormulaEvaluator evaluator =
workbook.getCreationHelper().createFormulaEvaluator();
    List<Course> parsedCourses = new ArrayList<>();
   Sheet sheet = workbook.getSheetAt(0);
   for (Row row: sheet) {
```

```
        String firstCellValue = row.getCell(0).getStringCellValue();
        if (firstCellValue.isEmpty()) {
            continue;
        }
        if ("ВСЬОГО:".equals(firstCellValue)) {            break;
        }
         parsedCourses.add(new Course(row.getCell(1).getStringCellValue(),
        getHours(row, evaluator), row.getCell(32).getStringCellValue()));
    }
    return parsedCourses;
}
```

Для перегляду зарахованих курсів, слід натиснути кнопку у верхньому правому куті (рис.4.6).

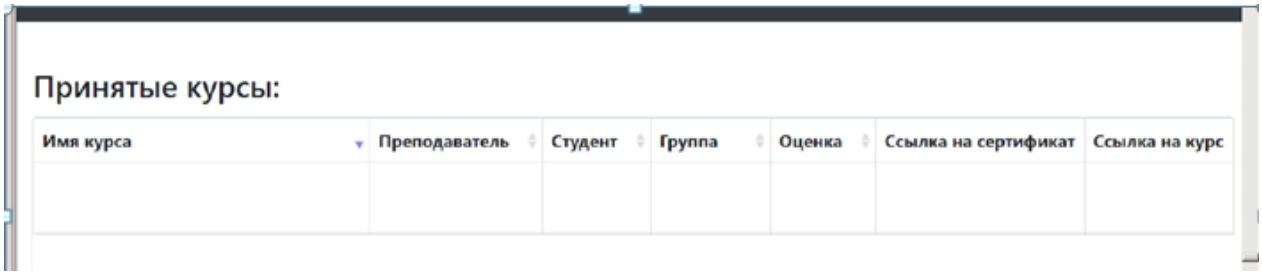


Рисунок 4.6 – Screenshot перегляду зарахованих курсів

Екран перегляду зарахованих дисциплін надає інформацію у вигляді таблиці, що має наступну інформацію: назву зарахованої дисципліни, прізвище та ім’я студента, прізвище та ім’я викладача, групу, у якій навчається студент; отриману студентом оцінку, посилання на пройдений студентом курс та посилання на отриманий студентом сертифікат. Таблиця може бути відсортована за усіма полями окрім посилань.

Для викладача в розробленій системі доступний наступний функціонал – перегляд відправлених студентами курсів на перевірку, що не були автоматично зараховані системою, та додавання переліку компетенції до дисциплін, яким викладач навчає студентів.

Для перегляду курсів, що були відправлені студентами на перевірку, на сторінці після авторизації є перелік таких курсів з додатковою інформацією (назва дисципліни, посилання на

курс та отриманий сертифікат, кількість прослуханих студентом лекційних годин та кількість годин, що були витрачені на практичні завдання; кількість компетенцій що співпали, та отримана студентом оцінка). Після перегляду сертифікату, курсу та оцінки наданої інформації, викладач має прийняти рішення щодо зарахування або відмови у зарахуванні даного курсу (рис.4.7).

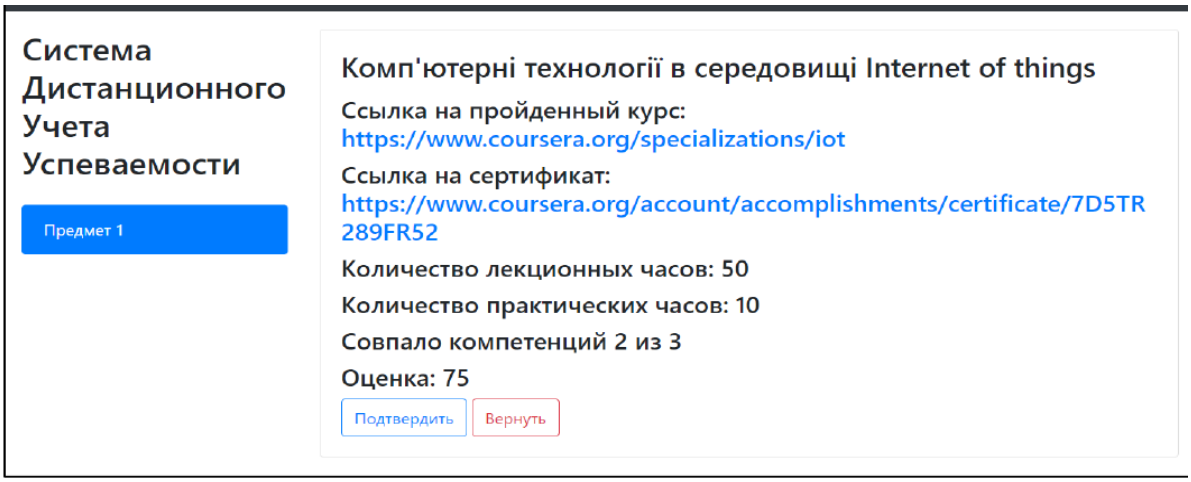


Рисунок 4.7 – Screenshot перегляду відправлених курсів

Для додавання компетенцій до дисципліни, що викладає педагог слід натиснути кнопку у правому верхньому куті, на наступному екрані викладач обирає дисципліну до якої він хоче додати компетенції та переходить на екран додавання. На цьому екрані викладач може додати максимум 15 компетенцій до обраної ним дисципліни.

Після натискання кнопки додавання компетенцій (рис.4.8) – введені викладачем компетенції будуть додані до обраної дисципліни, та ця дисципліна не буде відображена у переліку дисциплін без компетенцій на попередньому екрані.

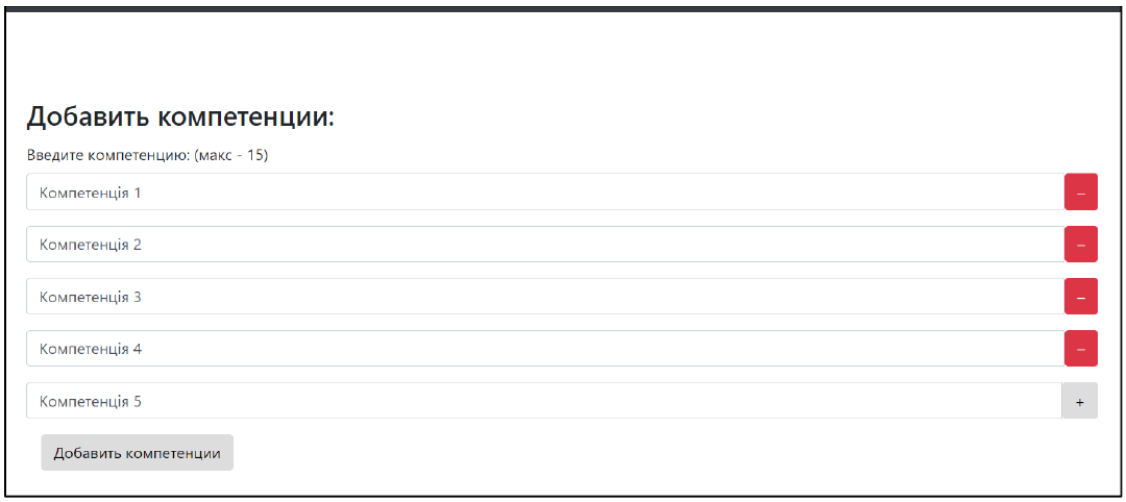


Рисунок 4.8 – Screenshot додавання компетенцій

Для студента система надає наступний функціонал – перегляд відправлених ним курсів, з додатковою інформацією та їхнім поточним статусом (знаходиться на розгляді у викладача, зараховано або відмовлено), та функціонал додавання курсу на перевірку з заповненням відповідних полів та обиранням компетенцій, що були ним засвоєні під час проходження курсу. Екран перегляду переліку відправлених курсів буде відображений одразу після авторизації студента (рис.4.9).

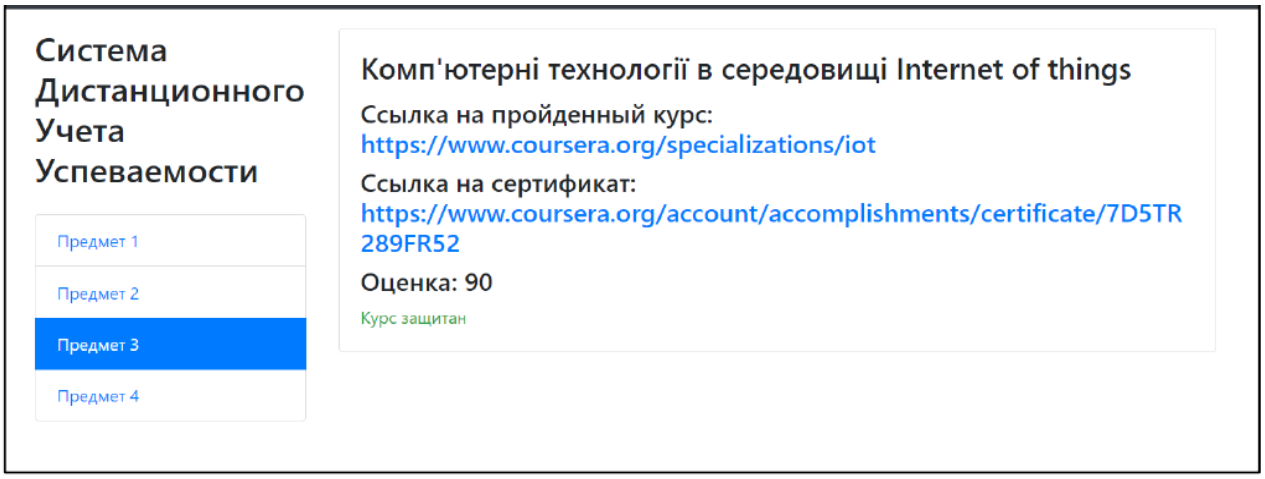


Рисунок 4.9 – Screenshot перегляду відправлених курсів

Функціонал відправлення курсу на перевірку у системі складається з двох послідовних етапів:

1. Перший етап – студент обирає дисципліну, яку він пройшов дистанційно з переліку тих, для яких викладачами були визначені компетенції та заповнює інформацію про пройдений ним курс, а саме – кількість прослуханих лекційних годин, кількість годин витрачених на практичні завдання, посилання на пройдений курс, посилання на отриманий ним сертифікат та отриману ним оцінку з даного курсу в СДН (рис.4.10).

2. Другий етап – студент повинен відмітити ті компетенції з переліку заданих викладачем, що були ним отримані у ході вивчення доданого ним курсу (рис.4.11).

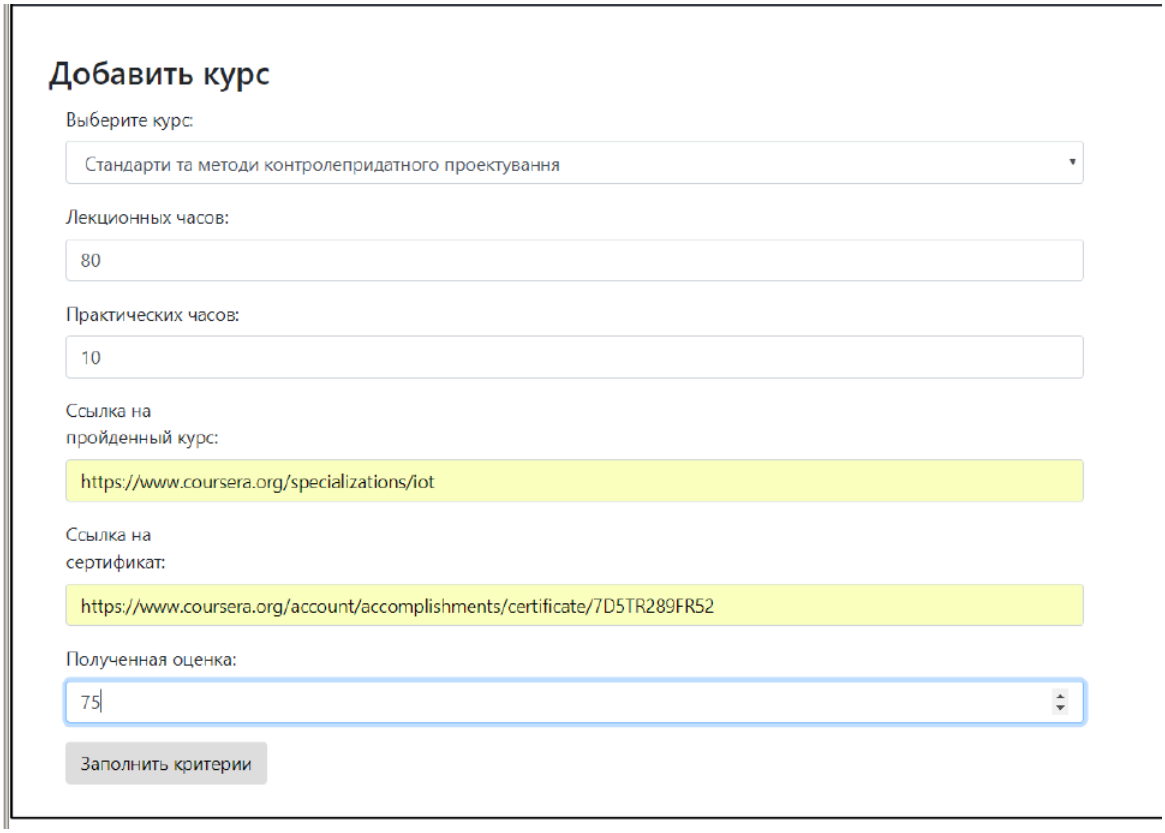


Рисунок 4.10 – Screenshot заповнення інформації про курс

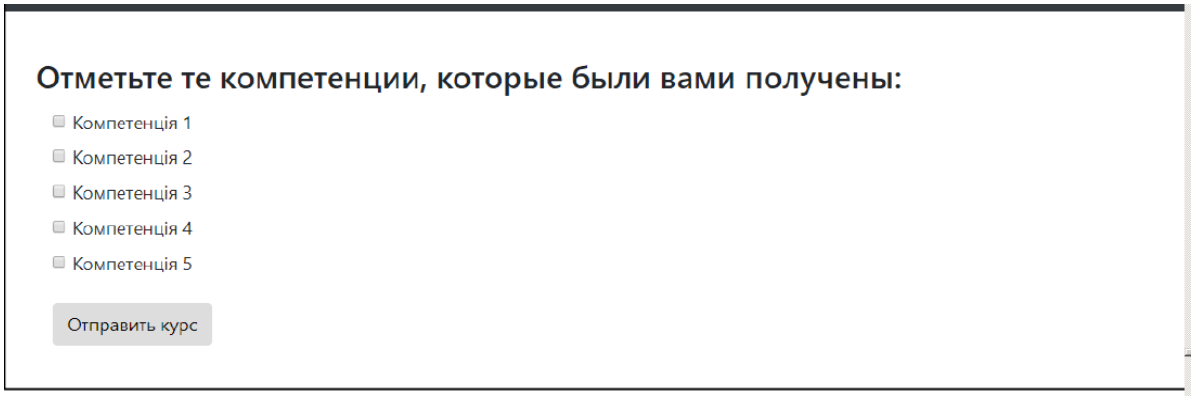


Рисунок 4.11 – Screenshot заповнення компетенцій курсу

Після натискання кнопки відправлення курсу на перевірку заповнена студентом інформація відправляється на сервер, де за спрощеним алгоритмом, який заснований на дереві прийняття рішень з розділу 3, визначається зараховувати цей курс автоматично або відправити на перевірку викладачу.

Алгоритм працює наступним чином – система підраховує суму витрачених студентом годин на практичні завдання та кількість годин, що були витрачені на прослуховування лекційного матеріалу, та перевіряє чи є ця кількість більшою, ніж кількість годин визначених на вивчення дисципліни робочою програмою помножена на визначений в системі

числовий поріг співпадіння, потім перевіряє чи є кількість вивчених студентом компетенцій більшою ніж кількість компетенцій, що були визначені викладачем, помножена на той самий поріг співпадіння. Якщо кількість годин курсу та кількість засвоєних компетенцій задовольняє порогу співпадіння, то курс автоматично зараховується, якщо ні – курс відправляється на перевірку викладачу. У лістингу нижче наведено Java код, що реалізує цей алгоритм.

Лістинг 4.2 – Код алгоритму автоматичного зарахування курсу

```
int submittedCurseHours = particularCourseData.getLecturesHours() +
particularCourseData.getPracticalHours();
boolean isHourAccepted = course.getHours() * AUTO_ACCEPT_CRITERIA <=
submittedCurseHours; List<String> courseCriteria =
teacherService.getCourseCriteria(particularCourseData.getCourseId());
List<String> notAcceptedCriteria = courseCriteria.stream()
    .filter(i -> particularCourseData.getSubmiitedCriteria().contains(i))
    .collect(Collectors.toList());

boolean isCriteriaAccepted = courseCriteria.size() * AUTO_ACCEPT_CRITERIA <=
notAcceptedCriteria.size();
submittedCourse.setCourseStatus(isHourAccepted && isCriteriaAccepted ?
SubmittedCourseStatus.ACCEPTED
    : SubmittedCourseStatus.SUBMITTED);
submittedCourse.setAcceptedCriteries(courseCriteria.size() -
notAcceptedCriteria.size());
```

Таким чином, було розроблено прототип системи інтеграції дистанційн навчальних систем з традиційною системою освіти, наочно доведено можливість створення цієї системи та її використання. Розроблений прототип може стати основою для розробки програмного продукту, що буде інтегрований у вже діючі системи університету.

5 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ. ЕКОЛОГІЯ

В даному розділі проведено аналіз потенційних небезпечних та шкідливих виробничих факторів, причин пожеж. Розглянуті заходи, які дозволяють забезпечити гігієну праці і виробничу санітарію. На підставі аналізу розроблені заходи з техніки безпеки та рекомендації з пожежної профілактики.

Аналіз потенційно небезпечних і шкідливих виробничих чинників виконується для персонального комп'ютера, на якому буде виконуватися розробка.

5.1 Загальні питання з охорони праці

Умови праці на робочому місці, безпека технологічних процесів, машин, механізмів, устаткування та інших засобів виробництва, стан засобів колективного та індивідуального захисту, що використовуються працівником, а також санітарно-побутові умови повинні відповідати вимогам нормативних актів про охорону праці. В законі України «Про охорону праці» визначається, що охорона праці - це система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних і лікувально-профілактичних заходів та засобів, спрямованих на збереження життя, здоров'я і працездатності людини у процесі трудової діяльності.

Основним організаційним напрямом у здійсненні управління в сфері охорони праці є усвідомлення пріоритету безпеки праці і підвищення соціальної відповідальності держави, і особистої відповідальності працівників.

В організації/підприємстві проводиться навчання і перевірка знань з питань охорони праці відповідно до вимог Типового положення про порядок проведення навчання і перевірки знань з питань охорони праці, затвердженого наказом Держнаглядохоронпраці України від 26.01.2005 N 15, зареєстрованого в Міністерстві юстиції України 15.02.2005 за N 231/10511 НПАОП 0.00-4.12-05 [13].

Також впроваджені організаційні заходи з пожежної безпеки - відповідно до вимог Типового положення про інструктажі, спеціальне навчання та перевірку знань з питань пожежної безпеки на підприємствах, в установах та організаціях України, затвердженого наказом Міністерства України з питань надзвичайних ситуацій та у справах захисту населення від наслідків Чорнобильської катастрофи від 29.09.2003 N 368, зареєстрованого в Міністерстві юстиції України 11.12.2003 за N 1148/8469 НАПБ Б.02.005-2003 [25].

Обов’язковими вимогами враховане наступне:

– не слід допускати до роботи осіб, що в установленому порядку не пройшли навчання, інструктаж та перевірку знань з охорони праці, пожежної безпеки;

– на підприємстві/організації, де експлуатуються ЕОМ з відео дисплейними терміналами (ВДТ) і периферійними пристроями (ПП), розробляється інструкція з охорони праці відповідно до Положення про розробку інструкцій з охорони праці [14];

5.2 Аналіз стану умов праці

5.2.1 Вимоги до приміщень

Геометричні розміри приміщення зазначені в табл. 5.1.

Таблиця 5.1 – Розміри приміщення

Найменування	**Значення**
Довжина, м	5
Ширина, м	5
Висота, м	3
Площа, $м^2$	25
Об’єм, $м^3$	75

Згідно з [12] розмір площі для одного робочого місця оператора персонального комп’ютера має бути не менше 6 кв. м, а об’єм — не менше 20 куб. м. Отже, дане приміщення цілком відповідає зазначеним нормам.

5.2.2 Вимоги до організації місця праці

При порівнянні відповідності характеристик робочого місця нормативним основні вимоги до організації робочого місця за [10] (табл. 5.2) і відповідними фактичними значеннями для робочого місця, констатуємо повну відповідність.

Таблиця 5.2 - Характеристики робочого місця

Найменування параметра	Фактичне значення	Нормативне значення
Висота робочої поверхні, мм	750	680 ÷800
Висота простору для ніг, мм	730	не менше 600
Ширина простору для ніг, мм	660	не менше 500
Глибина простору для ніг, мм	700	не менше 650
Висота поверхні сидіння, мм	470	400 ÷500
Ширина сидіння, мм	400	не менше 400
Глибина сидіння, мм	400	не менше 400
Висота поверхні спинки, мм	600	не менше 300
Ширина опорної поверхні спинки, мм	500	не менше 380
Радіус кривини спинки в горизонтальній площині, мм	400	400
Відстань від очей до екрану дисплея, мм	800	700 ÷800

Приміщення кабінету знаходиться на третьому поверсі трьох поверхової будівлі і має об’єм 78 м^3, площу – 25 м^2. У цьому кабінеті обладнано три місця праці, з яких два укомплектовані ПК.

Температура в приміщенні протягом року коливається у межах 18–24°С, відносна вологість — близько 50%. Швидкість руху повітря не перевищує 0,2 м/с. Шум в лабораторії знаходиться на рівні 50 дБА. Система вентилювання приміщення — природна неорганізована, а опалення — централізоване.

5.2.3 Навантаження та напруженість процесу праці

Під час виконання випускної роботи - за фізичним навантаженням робота відноситься до категорії легкі роботи (Іа), її виконують сидячи з періодичним ходінням. Щодо характеру організування виконання дипломної роботи, то він підпадає під нав’язаний режим, оскільки певні розділи роботи необхідно виконати у встановлені конкретні терміни.

Рекомендовано застосування екранних фільтрів, локальних світлофільтрів (засобів індивідуального захисту очей) та інших засобів захисту, а також інші профілактичні заходи

наведені в [10].

Роботу за дипломним проектом визнано, такою, що займає 50% часу робочого дня та за восьмигодинної робочої зміни рекомендовано встановити додаткові регламентовані перерви - для розробників програм тривалістю 15 хв. через кожну годину роботи;

5.3 Виробнича санітарія

На підставі аналізу небезпечних та шкідливих факторів при виробництві (експлуатації), пожежної безпеки можуть бути надалі вирішені питання необхідності забезпечення працюючих достатньою кількістю освітлення, вентиляції повітря, організації заземлення, тощо.

5.3.1 Аналіз небезпечних та шкідливих факторів при виробництві (експлуатації) виробу

Аналіз небезпечних та шкідливих виробничих факторів виконується у табличній формі (табл. 5.3). Роботу, пов'язану з ЕОП з ВДТ, у тому числі на тих, які мають робочі місця, обладнані ЕОМ з ВДТ і ПП, виконують із забезпеченням виконання НПАОП 0.00.-1.28-10 [24], які встановлюють вимоги безпеки до обладнання робочих місць, до роботи із застосуванням ЕОМ з ВДТ і ПП. Основними робочими характеристиками персонального комп'ютера є:

- робоча напруга U=+220В +-5%;
- робочий струм I=2А;
- споживана потужність P=350 Вт.

Таблиця 5.3 – Аналіз небезпечних і шкідливих виробничих факторів

Небезпечні і шкідливі виробничі фактори	Джерела факторів (види робіт)	Кількісна оцінка	Нормативні документи
1	2	3	4
фізичні			
- підвищена температура поверхонь обладнання	експлуатація ЕОМ, принтерів, сканерів	2	[12]
- підвищений рівень шуму на робочому місці	-//-	2	[11]
- підвищена або знижена вологість повітря	-//-	2	[12]
- підвищена або знижена рухливість повітря	-//-	1	[12]
- підвищений рівень іонізуючого випромінення в робочій зоні	-//-	2	[12] [22]

Продовження таблиці 5.3

- підвищений рівень електромагнітного випромінення	-//-	2	[22]
- підвищений рівень напруги електричної мережі, замикання якої може відбутися через тіло людини	-//-	4	[23] [12]
- підвищений рівень статичної електрики	-//-	2	[23]
- підвищена напруженість магнітного поля	-//-	2	[22]
- недостатність природного світла	порушення умов праці (вимог до приміщень)	2	[9]
- недостатнє освітлення робочої зони	порушення гігієнічних параметрів виробничого середовища	3	[9]
- підвищена яскравість світла	порушення умов праці (організації місця праці-налагодження моніторів)	1	[10]
- понижена контрастність	-//-	1	[10]
психофізіологічні:			
- нервово-психічна перевантаження (розумове, перенапруження аналізаторів-зорових)	- пошук інформації для постановки теми; - виконання роботи за темою диплома, тестування; - оформлення роботи	4	[24] [10]
- фізичні (статичне – сидіння)	порушення умов праці (організації місця праці - сидіння користувача) та організації робочого часу - безпервна робота)	2	[24] [10]

Робочі місця мають відповідати вимогам Державних санітарних правил і норм роботи з візуальними дисплейними терміналами електронно-обчислювальних машин, затверджених постановою Головного державного санітарного лікаря України від 10.12.98 N 7 [10].

5.3.2 Пожежна безпека

Небезпека розвитку пожежі на обчислювальному центрі обумовлюється застосуванням розгалужених систем електроживлення ЕОМ, вентиляції і кондиціювання,. Небезпека загоряння пов'язана з особливістю комп'ютерів - із значною кількістю щільно розташованих на монтажній

платі блоках електронних вузлів, схем, електричних і комутаційних кабелів, та ін. При відхиленні реальних умов експлуатації від розрахункових можуть виникнути пожежонебезпечні ситуації.

Висока щільність елементів в електронних схемах призводить до значного підвищення температури окремих вузлів (80…100 ℃), що в умовах їх високої щільності може привести до перегріву, і може служити причиною запалювання ізоляційних матеріалів.

Для відводу теплоти від ЕОМ діє потужна система кондиціонування.

Потенційними джерелами запалювання можуть бути:

а) іскри і дуги короткого замикання;

б) електрична іскра при замиканні і розмиканні ланцюгів;

в) перегріви від тривалого перевантаження;

г) відкритий вогонь і продукти горіння;

д) наявність речовин, нагрітих вище за температуру самозаймання;

е) розрядна статична електрика.

Продуктами згорання, що виділяються на пожежі, є: окис вуглецю; сірчистий газ; окис азоту; синильна кислота; акромін; фосген; хлор і ін. При горінні пластмас, окрім звичних продуктів згорання, виділяються різні продукти термічного розкладання: хлорангідридні кислоти, формальдегіди, хлористий водень, фосген, синильна кислота, аміак, фенол, ацетон, стирол [8].

Згідно [16] таке приміщення, площею 25 м^2, відноситься до категорії "В", в якому передбачено устаткування автоматичною пожежною сигналізацією із застосуванням датчиків-сповіщувачів РІД-1.

5.3.3 Електробезпека

На робочому місці виконуються наступні вимоги електробезпеки: ПК, периферійні пристрої та устаткування для обслуговування, електропроводи і кабелі за виконанням та ступенем захисту відповідають класу зони за ПУЕ (правила улаштування електроустановок), мають апаратуру захисту від струму короткого замикання та інших аварійних режимів. Лінія електромережі для живлення ПК, периферійних пристроїв і устаткування для обслуговування, виконана як окрема групова три- провідна мережа, шляхом прокладання фазового, нульового робочого та нульового захисного провідників. Нульовий захисний провідник використовується для заземлення (занулення) електроприймачів. Штепсельні з'єднання та електророзетки крім контактів фазового та нульового робочого провідників мають спеціальні контакти для

підключення нульового захисного провідника. Електромережа штепсельних розеток для живлення персональних ПК укладено по підлозі поруч зі стінами відповідно до затвердженого плану розміщення обладнання та технічних характеристик обладнання. Металеві труби та гнучкі металеві рукави заземлені. Захисне заземлення включає в себе заземлюючих пристроїв і провідник, який з'єднує заземлюючий пристрій з обладнанням, яке заземлюється - заземлюючий провідник.

5.4 Гігієнічні вимоги до параметрів виробничого середовища

5.4.1 Освітлення

У приміщенні, де розташовані ЕОМ передбачається природне бічне освітлення, рівень якого відповідає [9]. Джерелом природного освітлення є сонячне світло. Регулярно повинен проводитися контроль освітленості, який підтверджує, що рівень освітленості задовольняє ДБН і для даного приміщення в світлий час доби достатньо природного освітлення.

Розрахунок освітлення.

Для виробничих та адміністративних приміщень світловий коефіцієнт приймається не менше -1/8, в побутових – 1/10:

$$S_b = \left(\frac{1}{5} \div \frac{1}{10}\right) \cdot S_n, \qquad (5.1)$$

де S_b – площа віконних прорізів, м^2;

S_n – площа підлоги, м^2.

$S_n = a \cdot b = 5 \cdot 5 = 25$ м^2,

$S = 1/8 \cdot 25 = 3{,}125$ м^2.

Приймаємо 2 вікна площею S=1,6 м^2 кожне.

Розрахунок штучного освітлення виробляється по коефіцієнтах використання світлового потоку, яким визначається потік, необхідний для створення заданої освітленості при загальному рівномірному освітленні.

Розрахунок кількості світильників n виробляється по формулі (5.2):

$$n = \frac{E \cdot S \cdot Z \cdot K}{F \cdot U \cdot M}, \qquad (5.2)$$

де E – нормована освітленість робочої поверхні, визначається нормами – 300 лк;

S – освітлювана площа, м^2; S = 25 м^2;

Z – поправочний коефіцієнт світильника (Z = 1,15 для ламп розжарювання та ДРЛ; Z = 1,1 для люмінесцентних ламп) приймаємо рівним 1,1;

K – коефіцієнт запасу, що враховує зниження освітленості в процесі експлуатації – 1,5;

U – коефіцієнт використання, залежний від типу світильника, показника індексу приміщення і т.п. – 0,575

M – число люмінесцентних ламп в світильнику – 2;

F – світловий потік лампи – 5400лм (для ЛБ-80).

Підставивши числові значення у формулу (А.2), отримуємо:

$$n = \frac{300 \cdot 25 \cdot 1,1 \cdot 1,5}{5400 \cdot 0,575 \cdot 2} \approx 2,0$$

Приймаємо освітлювальну установку, яка складається з 2-х світильників, які складаються з двох люмінесцентних ламп загальною потужністю 160 Вт, напругою – 220 В.

5.4.2 Шум

Рівень шуму, що супроводжує роботу користувачів персональних комп'ютерів (зумовлений як роботою системних блоків, клавіатури, так і друкуванням на принтерах, а також зовнішніми чинниками), коливається у межах 50–65 дБА [11]. У залах опрацювання інформації та комп'ютерного набору рівні шуму не повинні перевищувати 65 дБА.

5.4.3 Вентилювання

У приміщенні, де знаходяться ЕОМ, повітрообмін реалізується за допомогою природної організованої вентиляції (вентиляційні шахти), тобто при V приміщення > 40 м3 на одного працюючого допускається природна вентиляція. Цей метод забезпечує приток потрібної кількості свіжого повітря, що визначається в СНіП.

5.5 Заходи з організації виробничого середовища та попередження виникнення надзвичайних ситуацій

Відповідно до санітарно-гігієнічних нормативів та правил експлуатації обладнання наводимо приклади деяких заходів безпеки.

Заходи безпеки під час експлуатації персонального комп'ютера та периферійних пристроїв передбачають:

- правильне організування місця праці та дотримання оптимальних режимів праці та відпочинку під час роботи з ПК;
- експлуатацію сертифікованого обладнання;
- дотримання заходів електробезпеки;
- забезпечення оптимальних параметрів мікроклімату;
- забезпечення раціонального освітлення місця праці (освітленість робочого місця не перевищувала 2/3 нормальної освітленості приміщення);
- облаштовуючи приміщення для роботи з ПК, потрібно передбачити припливно-витяжну вентиляцію або кондиціювання повітря.

Вимоги безпеки при надзвичайних ситуаціях:

При раптовому припиненні подачі електричної енергії вимкнути всі пристрої ПК в такій послідовності: периферійні пристрої, ВДТ, системний блок, стабілізатор (або блок безперервного живлення). Витягнути вилки з розеток. При наявності ознак горіння (дим, запах горілого) необхідно вимкнути всі пристрої ПК, знайти місце загоряння і виконати всі можливі заходи для його ліквідації, попередивши терміново про це керівництво.

Також застосовують різні **електричні захисні засоби від ураження струмом**:

Основні - володіють ізоляцією, здатної довго витримувати робоче напругу електроустановки і тому ними дозволяється стосуватися струмоведучих частин, знаходячи-трудящих під напругою.

Розрахунок захисного заземлення (забезпечення електробезпеки будівлі).

Згідно з класифікацією приміщень за ступенем небезпеки ураження електричним струмом [17], приміщення в якому проводяться всі роботи відноситься до першого класу (без підвищеної небезпеки). Коефіцієнт використання вертикальних заземлювачів $\eta_в$ в залежності від розміщення заземлювачів та їх кількості знаходиться в межах 0,4…0,99. Взаємну екрануючу дію горизонтального заземлювача (з'єднувальної смуги) враховують за допомогою коефіцієнта використання горизонтального заземлювача η_c.

Послідовність розрахунку.

1) Визначається необхідний опір штучних заземлювачів $R_{шт.з.}$:

$$R_{шт.з.} = \frac{R_д \cdot R_{пр.з.}}{R_{пр.з.} - R_д}, \qquad (5.3)$$

де $R_{пр.з.}$ – опір природних заземлювачів; $R_д$ – допустимий опір заземлення. Якщо природні заземлювачі відсутні, то $R_{шт.з.}=R_д$.

Підставивши числові значення у формулу (А.3), отримуємо:

$$R_{шт.з.} = \frac{4 \cdot 40}{40 - 4} \approx 4 \text{ Ом}$$

2) Опір заземлення в значній мірі залежить від питомого опору ґрунту ρ, Ом·м. Приблизне значення питомого опору глини приймаємо ρ=40 Ом•м (табличне значення).

3) Розрахунковий питомий опір ґрунту, $\rho_{розр}$, Ом·м, визначається відповідно для вертикальних заземлювачів $\rho_{розр.в}$, і горизонтальних $\rho_{розр.г}$, Ом·м за формулою:

$$\rho_{розр..} = \psi \cdot \rho, \qquad (5.4)$$

де ψ – коефіцієнт сезонності для вертикальних заземлювачів I кліматичної зони з нормальною вологістю землі, приймається для вертикальних заземлювачів $\rho_{розр.в}$=1,7 і горизонтальних $\rho_{розр.г}$=5,5 Ом·м.

$$\rho_{розр.в} = 1,7 \cdot 40 = 68 \text{ Ом•м}$$

$$\rho_{розр.г} = 5,5 \cdot 40 = 220 \text{ Ом•м}$$

4) Розраховується опір розтікання струму вертикального заземлювача $R_в$, Ом, за (5.5).

$$R_в = \frac{\rho_{розр.в}}{2 \cdot \pi \cdot l_в} \cdot \left(\ln \frac{2 \cdot l_в}{d_{ст}} + \frac{1}{2} \cdot \ln \frac{4 \cdot t + l_в}{4 \cdot t - l_в} \right), \qquad (5.5)$$

де $l_в$ – довжина вертикального заземлювача (для труб - 2−3 м; $l_в$=3 м);

$d_{ст}$ – діаметр стержня (для труб - 0,03−0,05 м; $d_{ст}$=0,05 м);

t – відстань від поверхні землі до середини заземлювача, яка визначається за ф. (5.6):

$$t = h_в + \frac{l_в}{2}, \qquad (5.6)$$

де $h_в$ – глибина закладання вертикальних заземлювачів (0,8 м); тоді $t = 0,8 + \frac{3}{2} = 2,3$ м

$$R_в = \frac{68}{2 \cdot \pi \cdot 3} \cdot \left(\ln \frac{2 \cdot 3}{0,05} + \frac{1}{2} \cdot \ln \frac{4 \cdot 2,3 + 3}{4 \cdot 2,3 - 3} \right) = 18,5 \text{ Ом}$$

5) Визначається теоретична кількість вертикальних заземлювачів n штук, без урахування коефіцієнта використання $\eta_в$:

$$n = \frac{2 \cdot R_в}{R_д} = \frac{2 \cdot 18,5}{4} = 9,25 \qquad (5.7)$$

І визначається коефіцієнт використання вертикальних електродів групового заземлювача без врахування впливу з'єднувальної стрічки $\eta_в$ =0,57 (табличне значення).

6) Визначається необхідна кількість вертикальних заземлювачів з урахуванням коефіцієнта використання $n_в$, шт:

$$n_в = \frac{2 \cdot R_в}{R_д \cdot \eta_в} = \frac{2 \cdot 18,5}{4 \cdot 0,57} = 16,2 \approx 16 \qquad (5.8)$$

7) Визначається довжина з'єднувальної стрічки горизонтального заземлювача $l_с$, м:

$$l_с = 1,05 \cdot L_в \cdot (n_в - 1), \qquad (5.9)$$

де $L_в$ – відстань між вертикальними заземлювачами, (прийняти за $L_в$ = 3м);

$n_в$ – необхідна кількість вертикальних заземлювачів.

$$l_с = 1,05 \cdot 3 \cdot (16 - 1) \approx 48 \text{ м}$$

8) Визначається опір розтіканню струму горизонтального заземлювача (з'єднувальної стрічки) $R_г$, Ом:

$$R_г = \frac{\rho_{розр.г}}{2 \cdot \pi \cdot l_с} \cdot \ln \frac{2 \cdot l_с^{\,2}}{d_{см} \cdot h_г}, \qquad (5.10)$$

де $d_{см}$ – еквівалентний діаметр смуги шириною b, $d_{см}$ =0,95b, b = 0,15 м;

$h_г$ – глибина закладання горизантальних заземлювачів (0,5 м);

$l_с$ - довжина з'єднувальної стрічки горизонтального заземлювача $l_с$, м

$$R_г = \frac{220}{2 \cdot \pi \cdot 48} \cdot \ln \frac{2 \cdot 48^2}{0,95 \cdot 0,15 \cdot 0,5} = 8,1 \text{ Ом}$$

9) Визначається коефіцієнт використання горизонтального заземлювача $\eta_{с.}$ відповідно до необхідної кількості вертикальних заземлювачів $n_в$.

Коефіцієнт використання з'єднувальної смуги $\eta_с$=0,3 (табличне значення).

10) Розраховується результуючий опір заземлювального електроду з урахуванням з'єднувальної смуги:

$$R_{заг} = \frac{R_в \cdot R_г}{R_в \cdot \eta_с + R_г \cdot n_в \cdot \eta_в} \le R_д. \qquad (5.11)$$

Висновок: дане захисне заземлення буде забезпечувати електробезпеку будівлі, так як виконується умова: $R_{заг}$ <4 Ом, а саме:

$$R_{заг} = \frac{18,5 \cdot 8,1}{18,5 \cdot 0,3 + 8,1 \cdot 16 \cdot 0,57} = 1,9 \le R_д$$

5.6 Вплив на навколишнє середовище

Закон України "Про охорону навколишнього середовища" [26] - визначає правові, економічні, соціальні основи охорони навколишнього середовища. Завдання Закону полягає в регулюванні відносин у галузі охорони праці, використанні та відновленню природних ресурсів, забезпеченні екологічної безпеки, попередженню та ліквідації наслідків негативної дії на навколишнє середовище діяльності людини, збереження природних ресурсів, генетичного фонду нації, ландшафтів й інших природних об'єктів. Під час магістерської роботи у приміщені утворюються відходи у вигляді зношених й відпрацьованих деталей, відходів паперу, люмінесцентні лампи та ін. Всі відходи здаються для подальшої утилізації. Жорсткість вимог до виробництва й матеріалів, а також розробка нових виробничих й утилізаційних технологій дозволяє зменшити антропогенне навантаження на навколишнє середовище.

ЖК-екрани - один з джерел парникових газів, які набагато шкідливіше діоксиду вуглецю. Рідкокристалічні монітори швидко знайшли популярність, прийшовши на зміну громіздким ЕПТ-моделям. За іншим аспектам екологічної безпеки дисплеї на основі рідких кристалів також вважалися проривом, тому що в них не використовувався газ, що містить свинець. Досить довго ніхто не звертав уваги на застосовуваний для чищення РК-панелей трехфторістий азот (NF3), і тільки в середині 2008 року вченими було доведено наявність даної хімічної речовини в атмосфері. Відкриття було вражаючим: порівняно з діоксидом вуглецю (CO2) NF3 має в 17 000 разів більше активного парникового газу, а його атмосферний час напіврозпаду може складати від 550 до 740 світлових років (у CO2 - від 30 до 40 років). Закону, який обмежував би рівень викиду NF3, поки не існує.

Виявлення енерговитрат є таким же проблематичним процесом, як і визначення кількості матеріалів, придатних для вторинної переробки, і важких металів, що містяться в пристроях. Таким чином, надійним показником екологічності залишається тільки рівень енергоспоживання.

Полівінілхлорид, що позначається зазвичай абревіатурою ПВХ, - це різновид пластику, що застосовується в самих різних цілях. З нього зроблена зовнішня оболонка кабелів, якими з'єднуються пристрої, він оточує електричний провід портативного комп'ютера. Це дешевий, міцний і вельми поширений матеріал. Він є причиною виникнення гормонального дисбалансу, проблем в репродуктивної сфері та різних форм раку. Полівінілхлорид практично неможливо правильно утилізувати. При його згорянні утворюється вкрай шкідливий канцерогенний діоксин. Єдиний спосіб правильно утилізувати ПВХ полягає в тому, щоб відправити його в центр небезпечних відходів.

ВИСНОВКИ

У даній роботі були проаналізовані системи дистанційної освіти, їх переваги та недоліки в порівнянні з традиційною системою освіти також проаналізовані різні системи дистанційного навчання.

В результаті проведеної роботи було зроблено аналіз умов праці, шкідливих та небезпечних чинників, з якими стикається робітник. Було визначено параметри і певні характеристики приміщення для роботи над запропонованим проектом, описано, які заходи потрібно зробити для того, щоб дане приміщення відповідало необхідним нормам і було комфортним і безпечним для робітника. Приведені рекомендації щодо організації робочого місця, а також важлива інформацію щодо пожежної та електробезпеки. Були наведені розміри приміщення та наведено значення температури, вологості й рухливості повітря, необхідна кількість і потужність ламп та інші параметри, значення яких впливає на умови праці робітника, а також – наведені інструкції з охорони праці, техніки безпеки при роботі на комп’ютері.

Було розроблено модель взаємодії систем дистанційного навчання з традиційною системою освіти, проаналізовано різні підходи до реалізації цієї моделі. Запропонована модель є значним кроком до удосконалення процесу навчання. Була розроблена програмна реалізація прототипу системи інтеграції СДН у традиційну систему освіти, що наочно демонструє можливість практичного використання розробленої моделі взаємодії.

Прототип, розроблений у результаті цієї роботи, може стати частиною одного з сервісів, що формують розумний кібер-університет, а саме частину послуги надання освітніх послуг у вигляді MOOC онлайн та onsite курсів.

Розроблена система була успішно протестована для перевірки виконання навчального плану, передбаченого програмою навчання магістрів спеціальності «Комп'ютерна інженерія», а також системи дистанційного навчання – Coursera.

ПЕРЕЛІК ПОСИЛАНЬ

1. СИСТЕМА ДИСТАНЦІЙНОГО НАВЧАННЯ (СДН) [Електронний ресурс] / Науково-методологічний центр дистанційного навчання. – Режим доступу: www / URL: - http://www.web-learn.ru/biblioteka-online/34-learning-management-system – 24.11.2018 р – Загл. з екрану.

2. Analytical survey Distance Education for the Information Society: Policies, Pedagogy and Professional Development. Moscow 2000, 86 pp., UNESCO Institute for Information Technologies in Education, pp.3-6

3. Coursera [Електронний ресурс] / Coursera. – Режим доступу www / URL: https://www.coursera.org/ – 24.11.2018 р – Загл. з екрану.

4. Udacity [Електронний ресурс] / Udacity. – Режим доступу www / URL: https://www.udacity.com/ – 24.11.2018 р – Загл. з екрану.

5. ІНТУІТ [Електронний ресурс] / ІНТУІТ. – Режим доступу www / URL: http://www.intuit.ru – 25.11.2018 р – Загл. з екрану.

6. Левитин А. В. Глава 10. Ограничения мощи алгоритмов: Деревья принятия решения // Алгоритмы. Введение в разработку и анализ — М.: Вильямс, 2006. — с. 409–417. — 576 с.

7. Фокіна Т.Н. Персональные учебные среды студента и преподавателя [Текст]: матеріали XI міжнародної науково-методичної конференції «Нові освітні технології у ВНЗ»., 2014. м. Єкатеринбург / – Єкатеринбург : НОТВ-2014.

8. ГОСТ 12.1.044-89 ССБТ. Пожежовибухонебезпека речовин і матеріалів. Номенклатура показників і методи їх визначенняя

9. ДБН В.2.5-28:2015 Природнє і штучне освітлення

10. ДСанПіН 3.3.2.007-98 Гігієнічні вимоги до організації роботи з візуальними дисплейними терміналами електронно-обчислювальних машин

11. ДСН 3.3.6.037-99 Санітарні норми виробничого шуму, ультразвуку та інфразвуку

12. ДСН 3.3.6.042-99 Санітарні норми мікроклімату виробничих

13. НПАОП 0.00-4.12-05 Типове положення про порядок проведення навчання і перевірки знань з питань охорони праці

14. НПАОП 0.00-4.15-98 Про розробку інструкцій з охорони праці

15. НПАОП 0.00-6.03-93 Порядок опрацювання та затвердження власником нормативних актів про охорону праці

16. НАПБ Б.03.002-2007 Норми визначення категорій приміщень, будинків та зовнішніх установок за вибухопожежною та пожежною небезпекою

17. НПАОП 40.1-1.01-97 Правила безопасной эксплуатации электроустановок

18. НПАОП 40.1-1.32-01 Правила устройства электроустановок. Электрооборудование специальных установок

19. ДСН 3.3.6.039-99 Санітарні норми виробничої загальної та локальної вібрації

20. ДСТУ ГОСТ 12.1.012-90 ССБТ. Вібраційна безпека. Загальні вимоги

21. ДБН В.2.5-67:2013 Опалення, вентиляція та кондиціонування

22. ГОСТ 12.1.006-84 ССБТ. Електромагнітні поля радіочастот. Загальні вимоги безпеки. Допустимі рівні на робочих місцях і вимоги до проведення контролю

23. ГОСТ 12.1.030-81 ССБТ. Електробезпечність. Захисне заземлення. Занулення

24. НПАОП 0.00-1.28-10 Правила охорони праці під час експлуатації електронно-обчислювальних машин

25. НАПБ Б.02.005-2003 Типове положення про інструктажі, спеціальне навчання та перевірку знань з питань пожежної безпеки на підприємствах, в установах та організаціях України.

26. Закон України “Про охорону навколишнього природного середовища”.

ДОДАТОК А

Лістинг коду розробленої системи з інтеграції СДН

```
@Controller
public class TeacherController {
@Autowired
private UserService userService;
@Autowired
private TeacherService teacherService;
@RequestMapping(value = "/not-accepted-courses", method =
RequestMethod.GET)
@ResponseStatus(HttpStatus.OK)
public ModelAndView getNotAcceptedCoursesPage(HttpServletResponse
response, Principal principal,
@RequestParam(value =
"submittedCourseId", required = false) Integer submittedCourseId) {
ModelAndView modelAndView = new ModelAndView();
List<SubmittedCourse> submittedCourses =
teacherService.getNotAcceptedCourses(principal.getName());
submittedCourses.sort(Comparator.comparingInt(SubmittedCourse::getCourseId));
SubmittedCourse activeSubmittedCourse;
if (submittedCourseId != null) {
activeSubmittedCourse = submittedCourses.stream()
.filter(submittedCourse1 ->
submittedCourse1.getId().equals(submittedCourseId))
.findFirst().orElse(null);
} else {
activeSubmittedCourse = submittedCourses.stream()
.min(Comparator.comparingInt(SubmittedCourse::getCourseId))
.orElse(null);
}
if (activeSubmittedCourse == null) {
modelAndView.setViewName("not-accepted-courses");
return modelAndView;  }
```

```
32.  modelAndView.addObject("submittedCourses", submittedCourses);
33.  Course course =
34.  userService.getCourseById(activeSubmittedCourse.getCourseId());
35.  NotAcceptedCourse activeCourse = new NotAcceptedCourse();
36.  activeCourse.setCourseName(course.getName());
37.  activeCourse.setCourseUrl(activeSubmittedCourse.getCourseUrl());
38.  activeCourse.setCertificateUrl(activeSubmittedCourse.getCertificateUrl());
39.  activeCourse.setLecturesHours(activeSubmittedCourse.getLecturesHours());
40.  activeCourse.setPracticalHours(activeSubmittedCourse.getPracticalHours());
41.  activeCourse.setAcceptedCriteria(activeSubmittedCourse.getAcceptedCriteries()
42.  );
43.  activeCourse.setAllCriteria(teacherService.getCourseCriteria(activeSubmittedC
44.  ourse.getCourseId()).size());
45.  activeCourse.setGraduate(activeSubmittedCourse.getGraduate());
46.  modelAndView.addObject("activeCourse", activeCourse);
47.  modelAndView.addObject("activeId", activeSubmittedCourse.getId());
48.  modelAndView.setViewName("not-accepted-courses");
49.  return modelAndView;
50.  }
51.  @RequestMapping(value = "/accept-course", method = RequestMethod.GET)
52.  @ResponseStatus(HttpStatus.OK)
53.  public void performCourseOperation(HttpServletResponse response,
54.  Principal principal,
55.  @RequestParam(value =
56.  "submittedCourseId", required = false) Integer submittedCourseId,
57.  @RequestParam(value = "operation",
58.  required = false) String operation) throws IOException {
59.  if (StringUtils.isNotEmpty(operation)) {
60.  if (operation.equals("accept")) {
61.  teacherService.acceptCourse(submittedCourseId);
62.  } else {
63.  teacherService.rejectCourse(submittedCourseId);
64.  }
65.  }
```

```
66.  response.sendRedirect("/dlas/not-accepted-courses");
67.  }
68.  @RequestMapping(value = "/courses-without-criteria", method =
69.  RequestMethod.GET)
70.  @ResponseStatus(HttpStatus.OK)
71.  public ModelAndView getCoursesWithoutCriteriaPage(HttpServletResponse
72.  response, Principal principal) {
73.  ModelAndView modelAndView = new ModelAndView();
74.  List<Course> coursesWithoutCriteria =
75.  teacherService.getCoursesWithoutCriteria(principal.getName());
76.  coursesWithoutCriteria.sort(Comparator.comparingInt(Course::getId));

77.  modelAndView.addObject("coursesWithoutCriteria",
78.  coursesWithoutCriteria);

79.  modelAndView.setViewName("courses-without-criteria");
80.  return modelAndView;
81.  }

82.  @RequestMapping(value = "/add-criteria", method = RequestMethod.GET)
83.  @ResponseStatus(HttpStatus.OK)
84.  public ModelAndView getAddCriteriaPage(@RequestParam(value = "courseId",
85.  required = true) Integer courseId) {

86.  ModelAndView modelAndView = new ModelAndView();

87.  modelAndView.addObject("courseId", courseId);

88.  modelAndView.setViewName("add-criteria");
89.  return modelAndView;
90.  }

91.  @RequestMapping(value = "/add-criteria", method = RequestMethod.POST)
92.  @ResponseStatus(HttpStatus.OK)
93.  public void uploadCriteries(HttpServletResponse response,
```

```
94.  @RequestParam(value = "courseId",
95.  required = true) Integer courseId,
96.  @RequestParam(value = "criteries",
97.  required = true) List<String> criteries) throws IOException {

98.  teacherService.uploadCriteries(courseId, criteries);

99.  response.sendRedirect("/dlas/courses-without-criteria");
100. }
101. }

102. @Controller
103. public class ManagerController {

104. @Autowired
105. private ManagerService managerService;

106. @Autowired
107. private UserService userService;

108. @RequestMapping(value = "/upload-program", method = RequestMethod.GET)
109. @ResponseStatus(HttpStatus.OK)
110. public ModelAndView getManagerHomePage(HttpServletResponse response) {
111. ModelAndView modelAndView = new ModelAndView();
112. modelAndView.setViewName("upload-program");
113. return modelAndView;
114. }

115. @RequestMapping(value = "/accepted-courses", method = RequestMethod.GET)
116. @ResponseStatus(HttpStatus.OK)
117. public ModelAndView getAcceptedCourses(HttpServletResponse response) {
118. ModelAndView modelAndView = new ModelAndView();
119. List<SubmittedCourse> acceptedCourses =
120. managerService.getAcceptedCourses();
```

```
121. acceptedCourses.sort(Comparator.comparingInt(SubmittedCourse::getId));

122. List<AcceptedCourse> acceptedCoursesDto = acceptedCourses.stream()
123. .map(submittedCourse -> {
124. AcceptedCourse acceptedCourse = new AcceptedCourse();
125. Course course =
126. userService.getCourseById(submittedCourse.getCourseId());
127. acceptedCourse.setCourseName(course.getName());
128. acceptedCourse.setGroupName(course.getGroupName());

129. acceptedCourse.setGraduate(submittedCourse.getGraduate());

130. acceptedCourse.setStudentName(userService.getUserNameByEmail(submittedCours
     e.
131. getStudentEmail()));

132. acceptedCourse.setTeacherName(userService.getUserNameByEmail(submittedCour
     se.
133. getTeacherEmail()));
134. acceptedCourse.setCourseUrl(submittedCourse.getCourseUrl());
135. acceptedCourse.setCertificateUrl(submittedCourse.getCertificateUrl());
136. return acceptedCourse;
137. }).collect(Collectors.toList());
138. modelAndView.addObject("acceptedCourses", acceptedCoursesDto);
139. modelAndView.setViewName("accepted-courses");
140. return modelAndView;
141. }

142. @RequestMapping(value = "/upload-program", method = RequestMethod.POST)
143. @ResponseStatus(HttpStatus.OK)
144. public ModelAndView uploadProgram(@RequestParam("upload-file")
145. MultipartFile file) {
146. ModelAndView view = new ModelAndView();
147. view.setViewName("upload-status");
148. view.addObject("isUploaded", managerService.uploadProgram(file));
```

```
149.  return view;
150.  }
151.  }

152.  @Controller
153.  public class StudentController {

154.  @Autowired
155.  private UserService userService;

156.  @Autowired
157.  private ManagerService managerService;

158.  @Autowired
159.  private TeacherService teacherService;

160.  @RequestMapping(value = "/student", method = RequestMethod.GET)
161.  @ResponseStatus(HttpStatus.OK)
162.  public ModelAndView getStudentHomePage(HttpServletResponse response,
163.  Principal principal,
164.  @RequestParam(value =
165.  "submittedCourseId", required = false) Integer submittedCourseId) {
166.  ModelAndView modelAndView = new ModelAndView();

167.  List<SubmittedCourse> submittedCourses =
168.  userService.getSubmittedCourses(principal.getName());

169.  submittedCourses.sort(Comparator.comparingInt(SubmittedCourse::getCourseId));
170.  SubmittedCourse activeSubmittedCourse;
171.  if (submittedCourseId != null) {
172.  activeSubmittedCourse = submittedCourses.stream()
173.  .filter(submittedCourse1 ->
174.  submittedCourse1.getId().equals(submittedCourseId))
175.  .findFirst().orElse(null);
176.  } else {
```

```
activeSubmittedCourse = submittedCourses.stream()

.min(Comparator.comparingInt(SubmittedCourse::getCourseId))
.orElse(null);
}

if (activeSubmittedCourse == null) {
modelAndView.setViewName("student");
return modelAndView;
}

modelAndView.addObject("submittedCourses", submittedCourses);

Course course =
userService.getCourseById(activeSubmittedCourse.getCourseId());
ActiveCourse activeCourse = new ActiveCourse();
activeCourse.setCourseName(course.getName());
activeCourse.setCourseUrl(activeSubmittedCourse.getCourseUrl());

activeCourse.setCertificateUrl(activeSubmittedCourse.getCertificateUrl());
activeCourse.setStatus(activeSubmittedCourse.getCourseStatus().name());
activeCourse.setGraduate(activeSubmittedCourse.getGraduate());
modelAndView.addObject("activeCourse", activeCourse);
modelAndView.addObject("activeId", activeSubmittedCourse.getId());
modelAndView.setViewName("student");

return modelAndView;
}

@RequestMapping(value = "/add-course", method = RequestMethod.GET)
@ResponseStatus(HttpStatus.OK)
public ModelAndView getAddCoursePage(HttpServletResponse response,
Principal principal) {
ModelAndView modelAndView = new ModelAndView();
String groupName = userService.getGroupName(principal.getName());
```

```
205.  if (StringUtils.isNotEmpty(groupName)) {
206.  modelAndView.addObject("courses",
207.  managerService.getCoursesForGroup(groupName));
208.  }
209.  modelAndView.setViewName("add-course");
210.  return modelAndView;
211.  }

212.  @RequestMapping(value = "/fill-criteria", method = RequestMethod.POST)
213.  @ResponseStatus(HttpStatus.OK)
214.  public ModelAndView fillParticularData(Principal principal, HttpSession
215.  session,
216.  @RequestParam("course") Integer courseId,
217.  @RequestParam("lectures-hours") Integer
218.  lecturesHours,
219.  @RequestParam("practical-hours") Integer
220.  practicalHours,
221.  @RequestParam("course-url") String courseUrl,
222.  @RequestParam("certificate-url") String
223.  certificateUrl,
224.  @RequestParam("graduate") Integer graduate) {

225.  AddCourseData particularFilledCourse = new AddCourseData();
226.  particularFilledCourse.setCourseId(courseId);
227.  particularFilledCourse.setLecturesHours(lecturesHours);
228.  particularFilledCourse.setPracticalHours(practicalHours);
229.  particularFilledCourse.setCourseUrl(courseUrl);
230.  particularFilledCourse.setCertificateUrl(certificateUrl);
231.  particularFilledCourse.setGraduate(graduate);
232.  particularFilledCourse.setStudentEmail(principal.getName());

233.  session.setAttribute("addedCourseData", particularFilledCourse);

234.  List<String> courseCriteries =
235.  teacherService.getCourseCriteria(courseId);
```

```
236. ModelAndView modelAndView = new ModelAndView();
237. modelAndView.addObject("courseCriteries", courseCriteries);
238. modelAndView.setViewName("fill-criteries");
239. return modelAndView;
240. }

241. @RequestMapping(value = "/add-course", method = RequestMethod.POST)
242. @ResponseStatus(HttpStatus.OK)
243. public void submitCourse(HttpServletResponse response, HttpSession
244. session,
245. @RequestParam("criteries") List<String>
246. submittedCriteries) throws IOException {
247. AddCourseData particularFilledCourse = (AddCourseData)
248. session.getAttribute("addedCourseData");
249. particularFilledCourse.setSubmiitedCriteria(submittedCriteries);

250. Integer submittedCourseId =
251. userService.submitCourse(particularFilledCourse);

252. session.setAttribute("addedCourseData", null);
253. response.sendRedirect(submittedCourseId == null ? "/dlas/student" :
254. "/dlas/student?submittedCourseId=" + submittedCourseId);
255. }
256. }

257. public class ManagerServiceImpl implements ManagerService {

258. @Autowired
259. private CoursesDAO coursesDAO;

260. @Autowired
261. private ProgramParser programParser;

262. @Override
```

```
263. public boolean uploadProgram(MultipartFile file) {
264. List<Course> parsedCourses = programParser.parseProgram(file);

265. if (parsedCourses.isEmpty()) {
266. return false;
267. }

268. return coursesDAO.uploadCourses(parsedCourses);
269. }

270. @Override
271. public List<Course> getCoursesForGroup(String groupName) {
272. return coursesDAO.getCoursesForGroup(groupName);
273. }

274. @Override
275. public List<SubmittedCourse> getAcceptedCourses() {
276. return coursesDAO.getAcceptedCourses();
277. }
278. }

279. public class TeacherServiceImpl implements TeacherService {

280. @Autowired
281. private TeacherDAO teacherDAO;

282. @Override
283. public String getTeacherEmailForCourse(Integer courseId) {
284. return teacherDAO.getTeacherEmailForCourse(courseId);
285. }

286. @Override
287. public List<SubmittedCourse> getNotAcceptedCourses(String teacherEmail) {
288. return teacherDAO.getNotAcceptedCoursesForTeacher(teacherEmail);
289. }
```

```
290.  @Override
291.  public void acceptCourse(Integer submittedCourseId) {
292.  teacherDAO.setCourseStatus(submittedCourseId,
293.  SubmittedCourseStatus.ACCEPTED);
294.  }

295.  @Override
296.  public void rejectCourse(Integer submittedCourseId) {
297.  teacherDAO.setCourseStatus(submittedCourseId,
298.  SubmittedCourseStatus.REJECTED);
299.  }

300.  @Override
301.  public List<Course> getCoursesWithoutCriteria(String teacherEmail) {
302.  return teacherDAO.getCoursesWithoutCriteria(teacherEmail);
303.  }

304.  @Override
305.  public void uploadCriteries(Integer courseId, List<String> criteries) {
306.  teacherDAO.uploadCriteries(courseId, criteries);
307.  }

308.  @Override
309.  public List<String> getCourseCriteria(Integer courseId) {
310.  return teacherDAO.getCourseCriteria(courseId);
311.  }
312.  }

313.  @Service
314.  public class UserServiceImpl implements UserService {

315.  private static final double AUTO_ACCEPT_CRITERIA = 0.8;

316.  @Autowired
```

```
317.  private UserDAO userDAO;

318.  @Autowired
319.  private TeacherService teacherService;

320.  @Autowired
321.  private CoursesDAO coursesDAO;

322.  @Override

323.  public User getUser(String username) {
324.  return userDAO.getUser(username);
325.  }
326.  @Override
327.  public String getGroupName(String email) {
328.  return userDAO.getUserGroupName(email);
329.  }

330.  @Override
331.  public List<SubmittedCourse> getSubmittedCourses(String studentEmail) {
332.  return userDAO.getSubmittedCoursesForStudent(studentEmail);
333.  }

334.  @Override
335.  public SubmittedCourse getSubmittedCourseById(Integer courseId) {
336.  return coursesDAO.getSubmittedCourseById(courseId);
337.  }

338.  @Override
339.  public Integer submitCourse(AddCourseData particularCourseData) {
340.  String teacherEmail =
341.  teacherService.getTeacherEmailForCourse(particularCourseData.getCourseId());
342.  Course course =
343.  coursesDAO.getCourseById(particularCourseData.getCourseId());
344.  if (StringUtils.isEmpty(teacherEmail) || Objects.isNull(course)) {
```

```
345.  return null;
346.  }

347.  SubmittedCourse submittedCourse = new SubmittedCourse();
348.  submittedCourse.setCourseId(particularCourseData.getCourseId());
349.  submittedCourse.setStudentEmail(particularCourseData.getStudentEmail());
350.  submittedCourse.setLecturesHours(particularCourseData.getLecturesHours());
351.  submittedCourse.setPracticalHours(particularCourseData.getPracticalHours());
352.  submittedCourse.setCourseUrl(particularCourseData.getCourseUrl());
353.  submittedCourse.setCertificateUrl(particularCourseData.getCertificateUrl());
354.  submittedCourse.setGraduate(particularCourseData.getGraduate());
355.  submittedCourse.setTeacherEmail(teacherEmail);

356.  int submittedCurseHours = particularCourseData.getLecturesHours() +
357.  particularCourseData.getPracticalHours();
358.  boolean isHourAccepted = course.getHours() * AUTO_ACCEPT_CRITERIA <=
359.  submittedCurseHours;

360.  List<String> courseCriteria =
361.  teacherService.getCourseCriteria(particularCourseData.getCourseId());
362.  List<String> notAcceptedCriteria = courseCriteria.stream()
363.  .filter(i ->
364.  !particularCourseData.getSubmiitedCriteria().contains(i))
365.  .collect(Collectors.toList());

366.  boolean isCriteriaAccepted = notAcceptedCriteria.size() <=
367.  courseCriteria.size() * AUTO_ACCEPT_CRITERIA;
368.  submittedCourse.setCourseStatus(isHourAccepted && isCriteriaAccepted
369.  ? SubmittedCourseStatus.ACCEPTED
370.  : SubmittedCourseStatus.SUBMITTED);
371.  submittedCourse.setAcceptedCriteries(courseCriteria.size() -
372.  notAcceptedCriteria.size());

373.  return coursesDAO.uploadSubmittedCourse(submittedCourse);
374.  }
```

```
375.  @Override
376.  public Course getCourseById(Integer courseId) {
377.  return coursesDAO.getCourseById(courseId);
378.  }

379.  @Override
380.  public String getUserNameByEmail(String email) {
381.  User user = userDAO.getUser(email);
382.  return user.getFirstName() + StringUtils.SPACE + user.getLastName();
383.  }
384.  }

385.  @Service
386.  public class UserDetailsServiceImpl implements UserDetailsService {

387.  @Autowired
388.  private UserService userService;

389.  @Override
390.  public UserDetails loadUserByUsername(String s) throws
391.  UsernameNotFoundException {
392.  User user;
393.  try {
394.  user = userService.getUser(s);
395.  } catch (Exception e) {
396.  throw new IllegalStateException(e.getMessage());
397.  }
398.  if(user == null) {
399.  throw new UsernameNotFoundException("No user for such username");
400.  }
401.  Set<GrantedAuthority> roles = new HashSet<>();
402.  roles.add(new SimpleGrantedAuthority(user.getRole().name()));
403.  return new
404.  org.springframework.security.core.userdetails.User(user.getEmail(),
```

```
405.  user.getPassword(), roles);
406.  }
407.  }

408.  public class ProgramParser {

409.  private static final Logger LOG = Logger.getLogger(ProgramParser.class);

410.  public List<Course> parseProgram(MultipartFile file) {
411.  Workbook workbook = getWorkbookFromFile(file);
412.  if (Objects.isNull(workbook)) {
413.  return Collections.emptyList();
414.  }
415.  FormulaEvaluator evaluator =
416.  workbook.getCreationHelper().createFormulaEvaluator();

417.  List<Course> parsedCourses = new ArrayList<>();

418.  Sheet sheet = workbook.getSheetAt(0);
419.  for (Row row: sheet) {
420.  String firstCellValue = row.getCell(0).getStringCellValue();
421.  if (firstCellValue.isEmpty()) {
422.  continue;
423.  }

424.  if ("ВСЬОГО:".equals(firstCellValue)) {
425.  break;
426.  }

427.  parsedCourses.add(new Course(row.getCell(1).getStringCellValue(),
428.  getHours(row, evaluator),
429.  row.getCell(32).getStringCellValue()));
430.  }
431.  return parsedCourses;
432.  }
```

```
433.  private int getHours(Row row, FormulaEvaluator evaluator) {
434.  Cell cell;
435.  if (CellType.FORMULA.equals(row.getCell(3).getCellTypeEnum())) {
436.  cell = row.getCell(3);
437.  } else {
438.  cell = row.getCell(17);
439.  }

440.  return CellType.FORMULA.equals(cell.getCellTypeEnum()) ?
441.  new
442.  Double(evaluator.evaluate(cell).getNumberValue()).intValue() : 0;
443.  }

444.  private Workbook getWorkbookFromFile(MultipartFile file) {
445.  try {
446.  if
447.  ("xlsx".equals(FilenameUtils.getExtension(file.getOriginalFilename()))) {
448.  return new XSSFWorkbook(file.getInputStream());
449.  }
450.  return new HSSFWorkbook(file.getInputStream());
451.  } catch (IOException e) {
452.  LOG.error("Can't open file: " + file.getOriginalFilename());
453.  }
454.  return null;
455.  }
456.  }

457.  public class BaseDAO {

458.  @Autowired
459.  private SessionFactory sessionFactory;

460.  private static final Logger LOG =
461.  Logger.getLogger(CoursesDAOHibernateImpl.class);
```

```
462.  protected BaseDAO() {
463.  }

464.  private Session currentSession;

465.  private Transaction currentTransaction;

466.  public Session openCurrentSession() {

467.  currentSession = sessionFactory.openSession();
468.  return currentSession;
469.  }

470.  public Session openCurrentSessionwithTransaction() {
471.  currentSession = sessionFactory.openSession();
472.  currentTransaction = currentSession.beginTransaction();
473.  return currentSession;
474.  }

475.  public void closeCurrentSession() {
476.  currentSession.close();
477.  }

478.  public void closeCurrentSessionwithTransaction() {
479.  if (!currentTransaction.wasCommitted()) {
480.  currentTransaction.commit();
481.  }
482.  currentSession.close();
483.  }

484.  public Session getCurrentSession() {
485.  return currentSession;
486.  }
```

```
487.  public Transaction getCurrentTransaction() {
488.  return currentTransaction;
489.  }
490.  }

491.  public class TeacherDAOHibernateImpl extends BaseDAO implements
      TeacherDAO {
492.  private int batchSize;
493.  public TeacherDAOHibernateImpl(int batchSize) {
494.  this.batchSize = batchSize;
495.  }

496.  private static final Logger LOG =
497.  Logger.getLogger(TeacherDAOHibernateImpl.class);

498.  @Override
499.  public String getTeacherEmailForCourse(Integer courseId) {
500.  try {
501.  Session session = openCurrentSession();
502.  Query q = session.createSQLQuery("select teacher_email from
503.  teacher_course where course_id = ?")
504.  .addScalar("teacher_email", new StringType());
505.  q.setInteger(0, courseId);
506.  return q.uniqueResult().toString();
507.  } catch (HibernateException e) {
508.  LOG.error("Could not get teacher email for course " + courseId,
509.  e);
510.  } finally {
511.  closeCurrentSession();
512.  }
513.  return StringUtils.EMPTY;
514.  }

515.  @Override
516.  public List<SubmittedCourse> getNotAcceptedCoursesForTeacher(String
```

```
517.  teacherEmail) {
518.  try {
519.  Session session = openCurrentSession();
520.  Query q = session.createQuery("from SubmittedCourse where
521.  teacherEmail = :teacherEmail and courseStatus = :courseStatus");
522.  q.setParameter("teacherEmail", teacherEmail);
523.  q.setParameter("courseStatus", SubmittedCourseStatus.SUBMITTED);
524.  return q.list();
525.  } catch (HibernateException e) {
526.  LOG.error("Could not get courses", e);
527.  return Collections.emptyList();
528.  } finally {
529.  closeCurrentSession();
530.  }
531.  }

532.  @Override
533.  public void setCourseStatus(Integer submittedCourseId,
534.  SubmittedCourseStatus courseStatus) {
535.  try {
536.  Session session = openCurrentSession();
537.  SubmittedCourse course = (SubmittedCourse)
538.  session.get(SubmittedCourse.class, submittedCourseId);
539.  course.setCourseStatus(courseStatus);
540.  session.update(course);
541.  session.flush();
542.  } catch (HibernateException e) {
543.  LOG.error("Could not get courses", e);
544.  } finally {
545.  closeCurrentSession();
546.  }
547.  }

548.  @Override
549.  public List<Course> getCoursesWithoutCriteria(String teacherEmail) {
```

```
try {
Session session = openCurrentSession();
return session.createSQLQuery("select * from course INNER JOIN "
+
"teacher_course ON course.id=teacher_course.course_id
where " +
"teacher_course.teacher_email = :teacherEmail AND
course.criteria_added = 0")
.addEntity(Course.class)
.setParameter("teacherEmail", teacherEmail).list();
} catch (HibernateException e) {
LOG.error("Could not get courses", e);
return Collections.emptyList();
} finally {
closeCurrentSession();
}
}

@Override
public void uploadCriteries(Integer courseId, List<String> criteries) {
try {
Session session = openCurrentSessionwithTransaction();
for (int i = 0; i < criteries.size(); i++) {
session.createSQLQuery("INSERT INTO courses_criteria
(course_id, criteria) VALUES (:courseId, :criteria)")
.setParameter("courseId", courseId)
.setParameter("criteria", criteries.get(i))
.executeUpdate();

if (i % batchSize == 0) {
session.flush();
session.clear();
}
}
```

```
session.createSQLQuery("update course set course.criteria_added =
1 where course.id = :courseId")
.setParameter("courseId", courseId)
.executeUpdate();

} catch (HibernateException e) {
getCurrentTransaction().rollback();
LOG.error("Could not upload courses", e);
} finally {
closeCurrentSessionwithTransaction();
}
}

@Override
public List<String> getCourseCriteria(Integer courseId) {
try {
Session session = openCurrentSession();
return session.createSQLQuery("select criteria from
courses_criteria where course_id = :courseId")
.setParameter("courseId", courseId).list();
} catch (HibernateException e) {
LOG.error("Could not get courses", e);
return Collections.emptyList();
} finally {
closeCurrentSession();
}
}
}

@Repository
public class UserDAOHibernateImpl extends BaseDAO implements UserDAO {
private static final Logger LOG =
Logger.getLogger(UserDAOHibernateImpl.class);

@Override
```

```
613.  public User getUser(String email) {
614.  try {
615.  Session session = openCurrentSession();
616.  Query q = session.createQuery("from User where email = :email");
617.  q.setString("email", email);
618.  return (User) q.uniqueResult();
619.  } catch (HibernateException e) {
620.  LOG.error("Could not get user " + email, e);
621.  } finally {
622.  closeCurrentSession();
623.  }
624.  return null;
625.  }

626.  @Override
627.  public String getUserGroupName(String email) {
628.  try {
629.  Session session = openCurrentSession();
630.  Query q = session.createSQLQuery("select group_name from groups
631.  where student_email = ?")
632.  .addScalar("group_name", new StringType());
633.  q.setString(0, email);
634.  return q.uniqueResult().toString();
635.  } catch (HibernateException e) {
636.  LOG.error("Could not get group name for user " + email, e);
637.  } finally {
638.  closeCurrentSession();
639.  }
640.  return StringUtils.EMPTY;
641.  }

642.  @Override
643.  public List<SubmittedCourse> getSubmittedCoursesForStudent(String
644.  studentEmail) {
645.  try {
```

```
646. Session session = openCurrentSession();
647. Query q = session.createQuery("from SubmittedCourse where
648. studentEmail = :studentEmail");
649. q.setParameter("studentEmail", studentEmail);
650. return q.list();
651. } catch (HibernateException e) {
652. LOG.error("Could not get courses", e);
653. return Collections.emptyList();
654. } finally {
655. closeCurrentSession();
656. }
657. }
658. }

659. public class CoursesDAOHibernateImpl extends BaseDAO implements
     CoursesDAO {
660. private int batchSize;
661. public CoursesDAOHibernateImpl(int batchSize) {
662. this.batchSize = batchSize;
663. }

664. private static final Logger LOG =
665. Logger.getLogger(CoursesDAOHibernateImpl.class);

666. @Override
667. public boolean uploadCourses(List<Course> courses) {
668. try {
669. Session session = openCurrentSessionwithTransaction();

670. for (int i = 0; i < courses.size(); i++) {
671. session.save(courses.get(i));
672. if (i % batchSize == 0) {
673. session.flush();
674. session.clear();
675. }
```

```
676.  }
677.  return true;
678.  } catch (HibernateException e) {
679.  getCurrentTransaction().rollback();
680.  LOG.error("Could not upload courses", e);
681.  return false;
682.  } finally {
683.  closeCurrentSessionwithTransaction();
684.  }
685.  }

686.  @Override
687.  public List<Course> getCoursesForGroup(String groupName) {
688.  try {
689.  Session session = openCurrentSession();
690.  return session.createSQLQuery("SELECT * FROM course where
691.  group_name = :groupName AND criteria_added = 1")
692.  .addEntity(Course.class)
693.  .setParameter("groupName", groupName)
694.  .list();
695.  } catch (HibernateException e) {
696.  LOG.error("Could not get courses", e);
697.  return Collections.emptyList();
698.  } finally {
699.  closeCurrentSession();
700.  }
701.  }
702.  @Override
703.  public Integer uploadSubmittedCourse(SubmittedCourse submittedCourse) {
704.  try {
705.  return (Integer) openCurrentSession().save(submittedCourse);
706.  } catch (HibernateException e) {
707.  LOG.error("Could not get courses", e);
708.  return null;
709.  } finally {
```

```
710.  closeCurrentSession();
711.  }
712.  }
713.  @Override
714.  public Course getCourseById(Integer courseId) {
715.  try {
716.  Session session = openCurrentSession();
717.  return (Course) session.get(Course.class, courseId);
718.  } catch (HibernateException e) {
719.  LOG.error("Could not get course by id " + courseId, e);
720.  return null;
721.  } finally {
722.  closeCurrentSession();
723.  }
724.  }

725.  @Override
726.  public SubmittedCourse getSubmittedCourseById(Integer courseId) {
727.  try {
728.  Session session = openCurrentSession();
729.  return (SubmittedCourse) session.get(SubmittedCourse.class,
730.  courseId);
731.  } catch (HibernateException e) {
732.  LOG.error("Could not get course by id " + courseId, e);
733.  return null;
734.  } finally {
735.  closeCurrentSession();
736.  }
737.  }

738.  @Override
739.  public List<SubmittedCourse> getAcceptedCourses() {
740.  try {
741.  Session session = openCurrentSession();
742.  Query q = session.createQuery("from SubmittedCourse where
```

```
743.  courseStatus = :courseStatus");
744.  q.setParameter("courseStatus", SubmittedCourseStatus.ACCEPTED);
745.  return q.list();
746.  } catch (HibernateException e) {
747.  LOG.error("Could not get accepted courses", e);
748.  return null;
749.  } finally {
750.  closeCurrentSession();
751.  }
752.  }
753.  }
```

ДОДАТОК Б

Комп'ютерна презентація

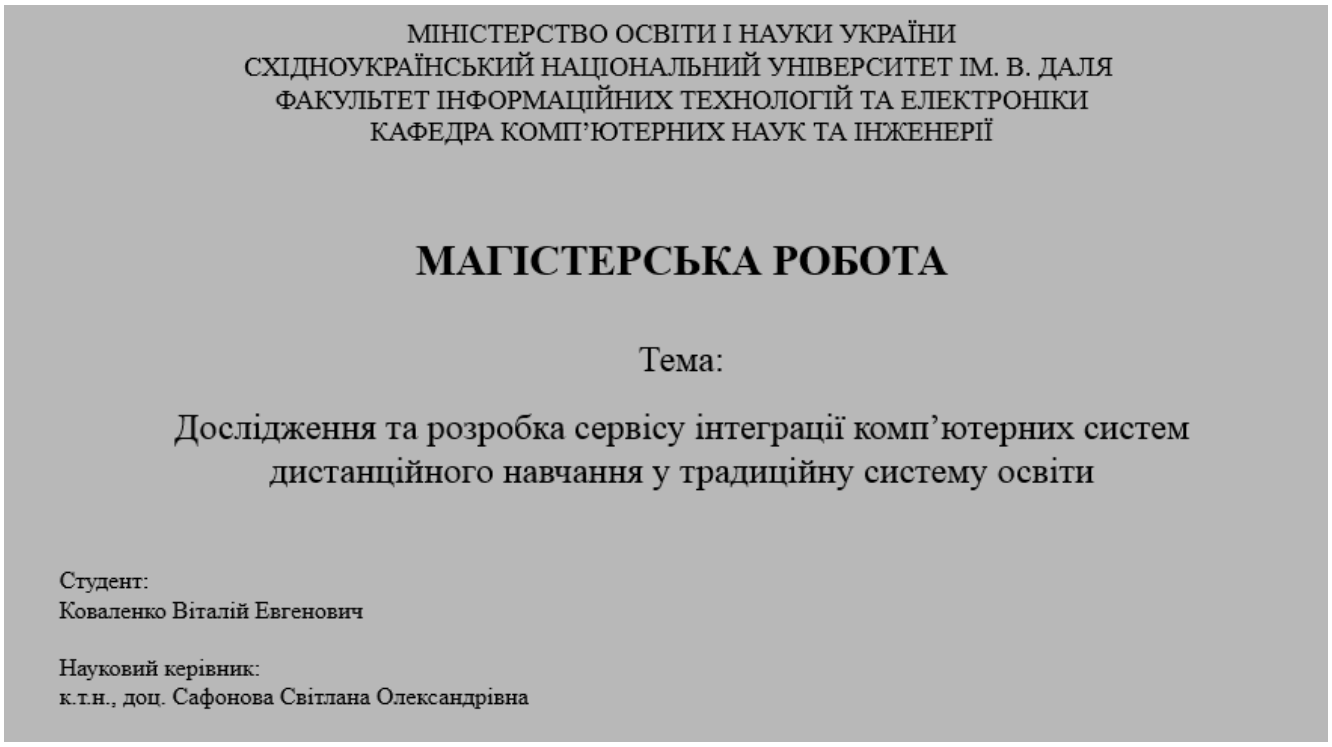


Рисунок Б.1- Слайд №1

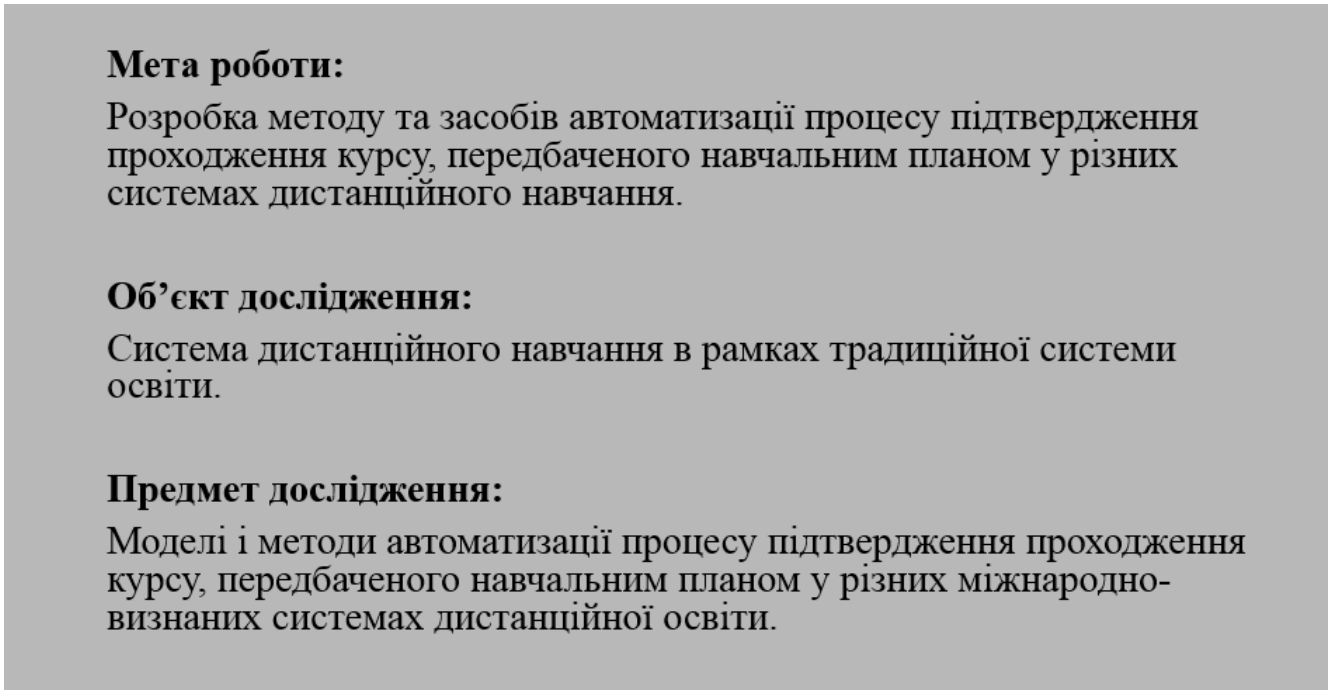


Рисунок Б.2- Слайд №2

Рисунок Б.3- Слайд №3

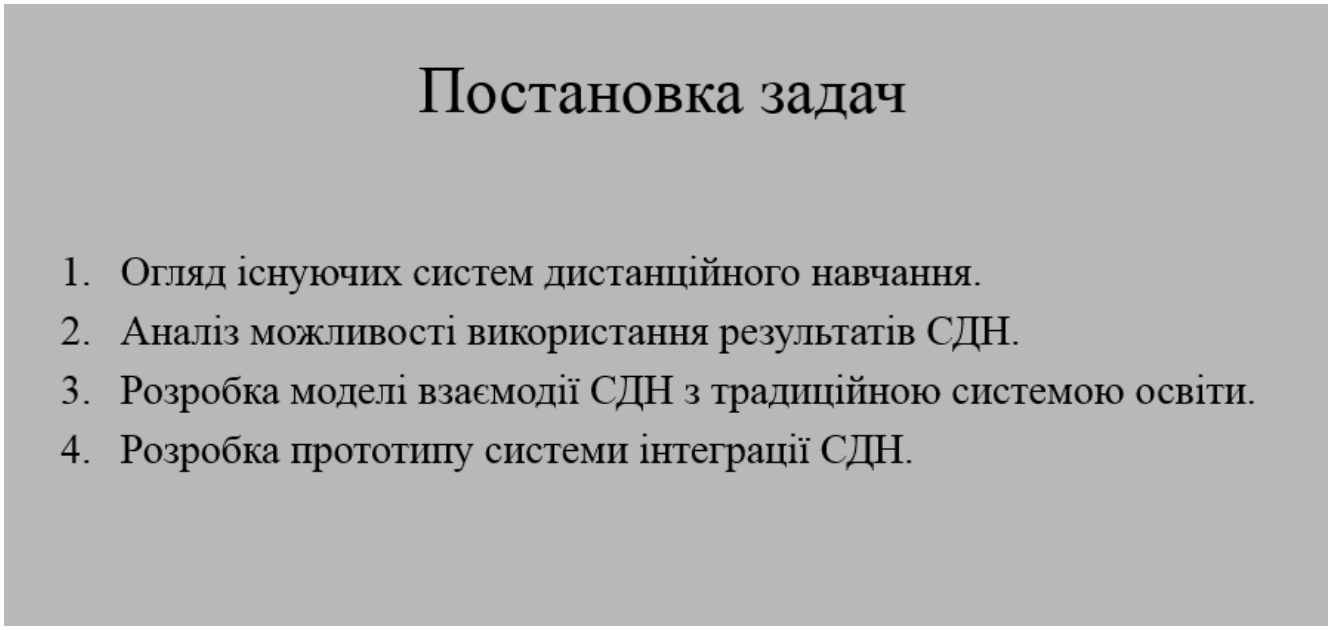


Рисунок Б.4- Слайд №4

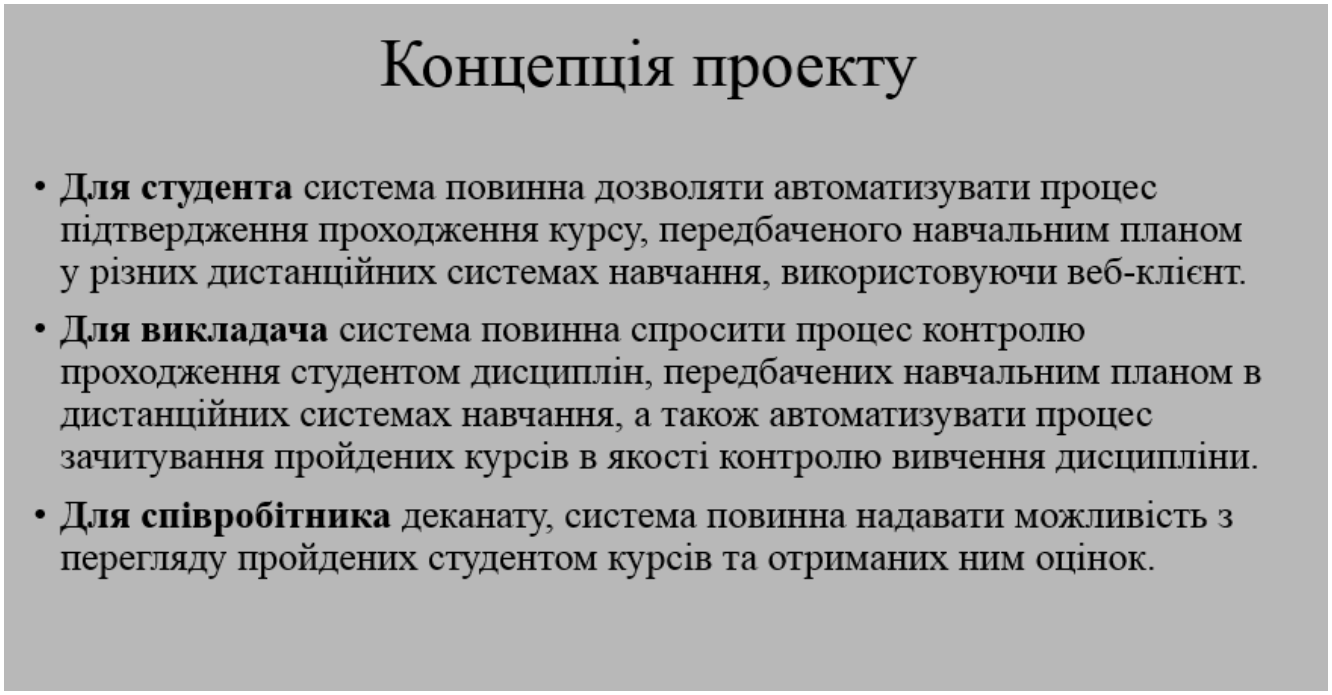


Рисунок Б.5- Слайд №5

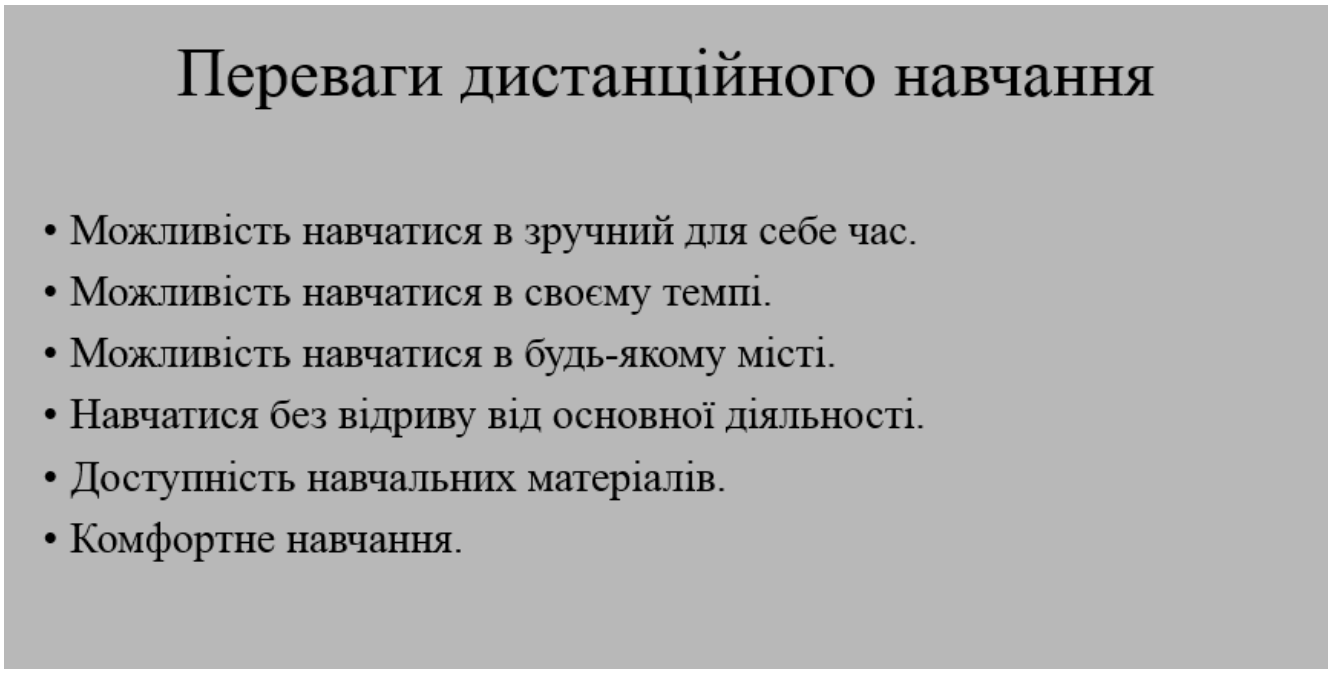


Рисунок Б.6- Слайд №6

Рисунок Б.7- Слайд №7

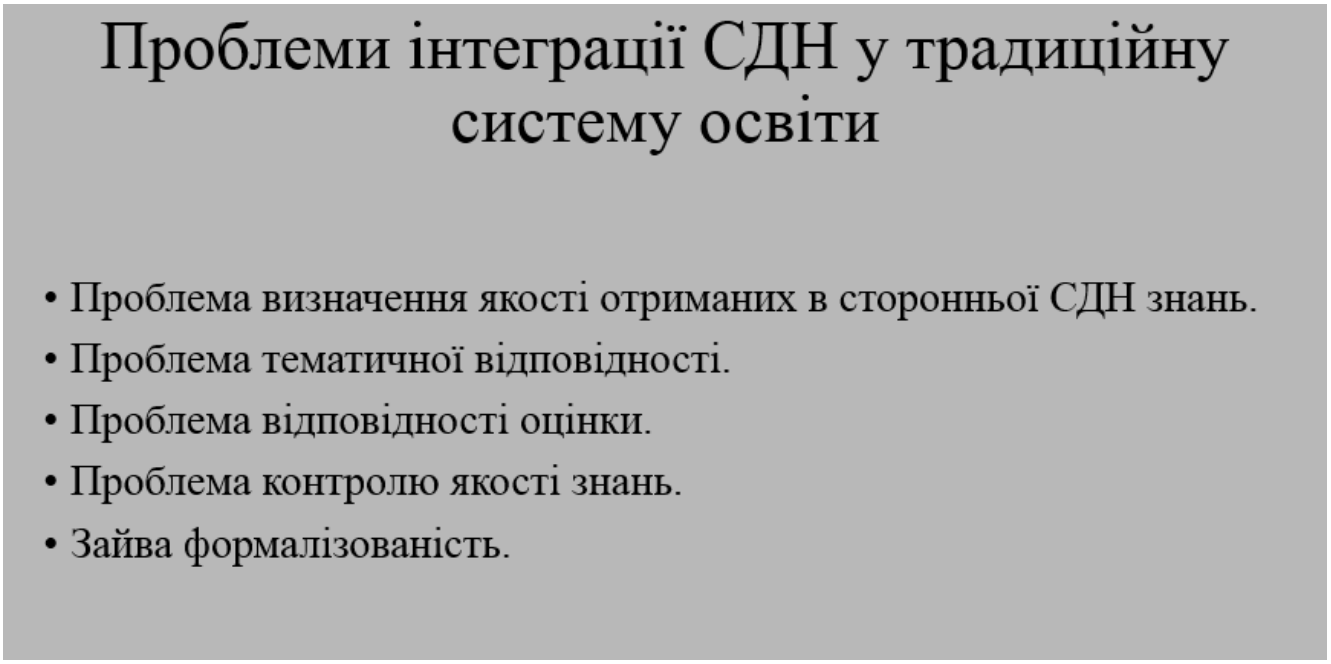


Рисунок Б.8- Слайд №8

Вимоги до курсу

Для того щоб було можливо зарахувати пройдений студентом курс в одній із СДН, необхідної умовою є те, що цей курс має відповідати певному набору критеріїв, а саме:

- назва курсу має відповідати дисципліні;
- об'єм курсу має відповідати об'єму дисципліни;
- структура курсу має відповідати структурі дисципліни;
- тематична наповненість курсу – тематика курсу має в певній мірі співпадати с тематичним наповненням дисципліни;
- має застосовувати компетенційний підхід.

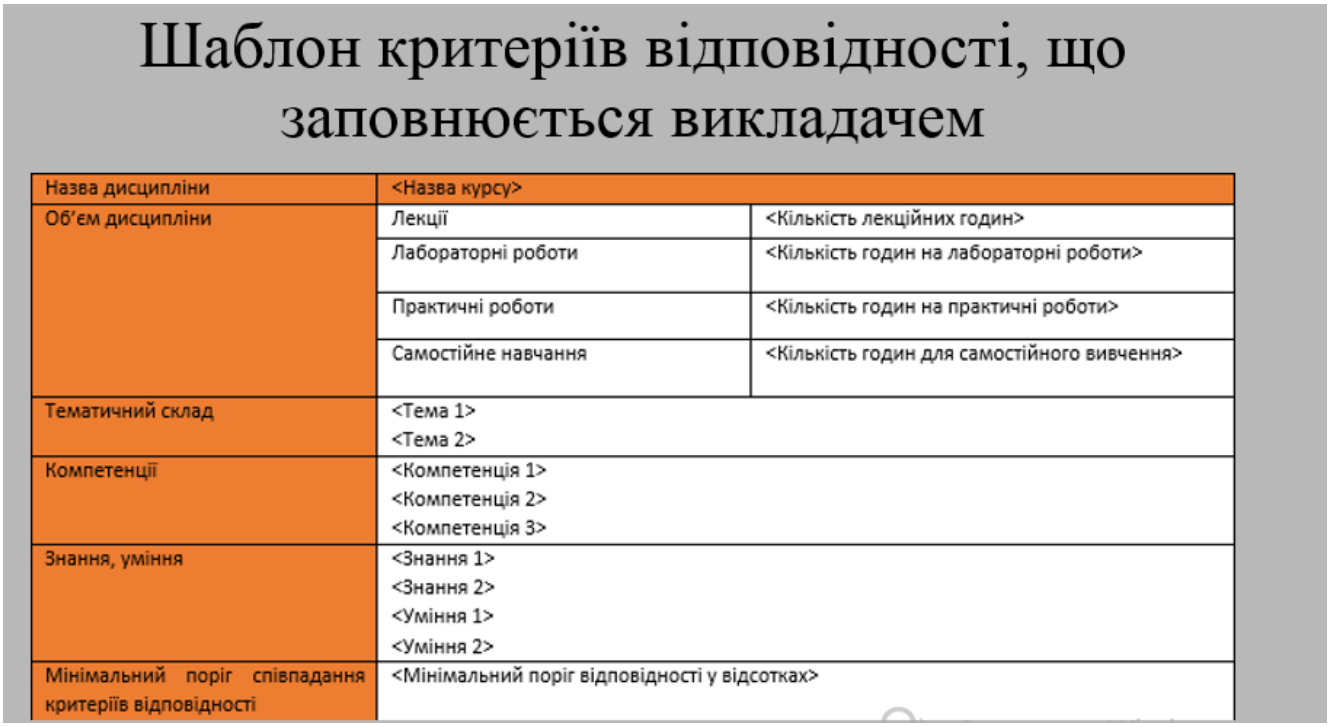
Рисунок Б.9- Слайд №9

Варіанти вирішення проблеми відповідності компетенцій

Для вирішення проблеми об'єктивності визначення відповідності здобутих студентом компетенцій є кілька підходів.

1. Студент відповідає на питання про компетенції у форматі тестування.
2. Студент відповідає на питання у форматі «Чи освоїли ви компетенцію 1?» з відповідями «Так» або «Ні».
3. Третій підхід – це розробка системи з методами машинного навчання, штучного інтелекту та засобів кросмовного аналізу.

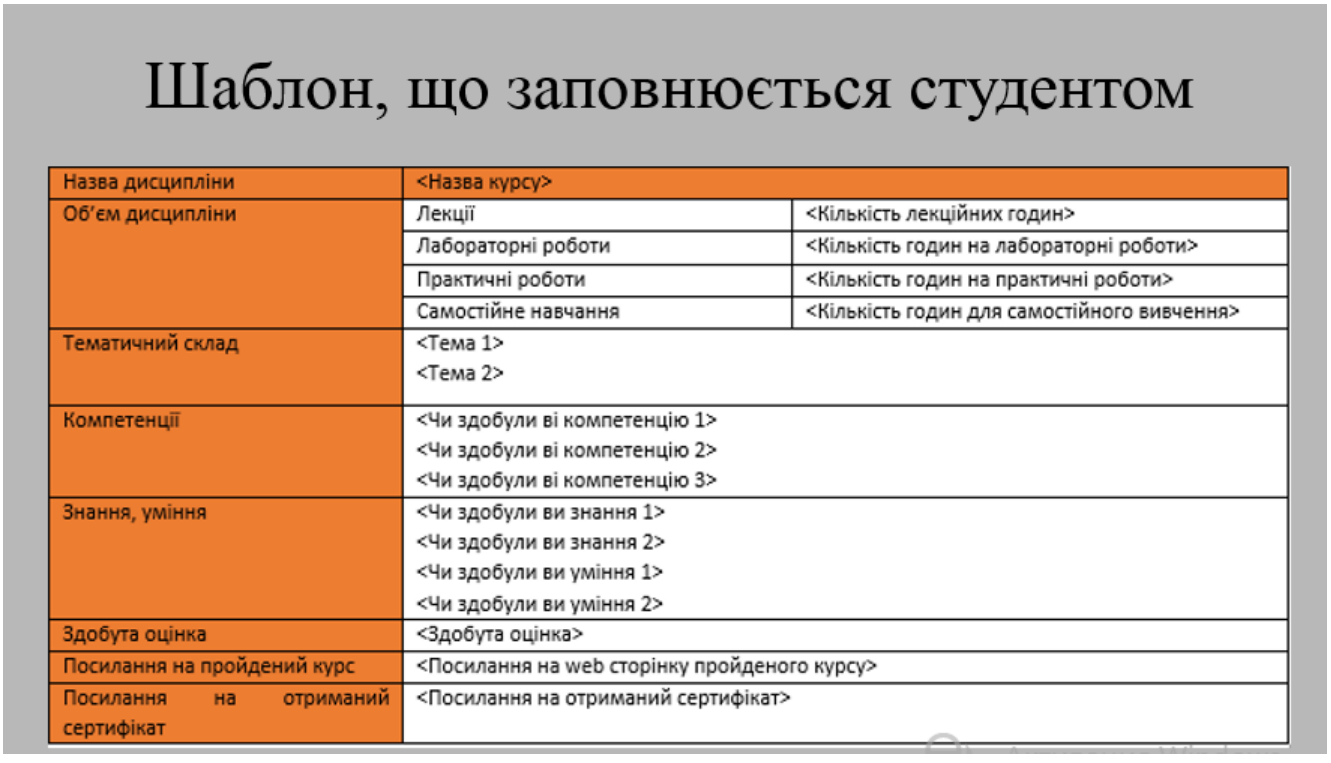
Рисунок Б.10- Слайд №10



Шаблон критеріїв відповідності, що заповнюється викладачем

Назва дисципліни	<Назва курсу>	
Об'єм дисципліни	Лекції	<Кількість лекційних годин>
	Лабораторні роботи	<Кількість годин на лабораторні роботи>
	Практичні роботи	<Кількість годин на практичні роботи>
	Самостійне навчання	<Кількість годин для самостійного вивчення>
Тематичний склад	<Тема 1> <Тема 2>	
Компетенції	<Компетенція 1> <Компетенція 2> <Компетенція 3>	
Знання, уміння	<Знання 1> <Знання 2> <Уміння 1> <Уміння 2>	
Мінімальний поріг співпадання критеріїв відповідності	<Мінімальний поріг відповідності у відсотках>	

Рисунок Б.11- Слайд №11



Шаблон, що заповнюється студентом

Назва дисципліни	<Назва курсу>	
Об'єм дисципліни	Лекції	<Кількість лекційних годин>
	Лабораторні роботи	<Кількість годин на лабораторні роботи>
	Практичні роботи	<Кількість годин на практичні роботи>
	Самостійне навчання	<Кількість годин для самостійного вивчення>
Тематичний склад	<Тема 1> <Тема 2>	
Компетенції	<Чи здобули ві компетенцію 1> <Чи здобули ві компетенцію 2> <Чи здобули ві компетенцію 3>	
Знання, уміння	<Чи здобули ви знання 1> <Чи здобули ви знання 2> <Чи здобули ви уміння 1> <Чи здобули ви уміння 2>	
Здобута оцінка	<Здобута оцінка>	
Посилання на пройдений курс	<Посилання на web сторінку пройденого курсу>	
Посилання на отриманий сертифікат	<Посилання на отриманий сертифікат>	

Рисунок Б.12- Слайд №12

Рисунок Б.13- Слайд №13

Рисунок Б.14- Слайд №14

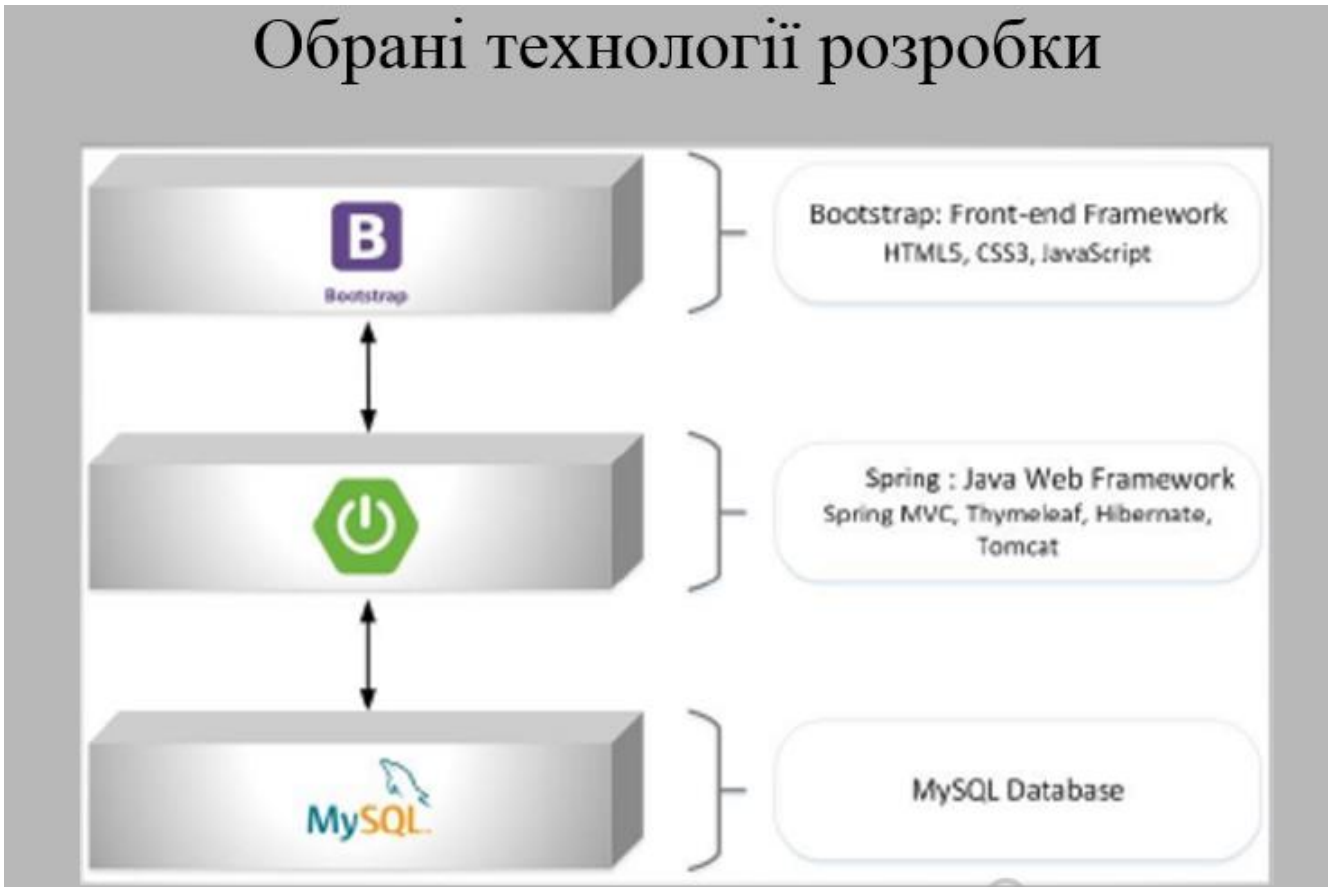


Рисунок Б.15- Слайд №15

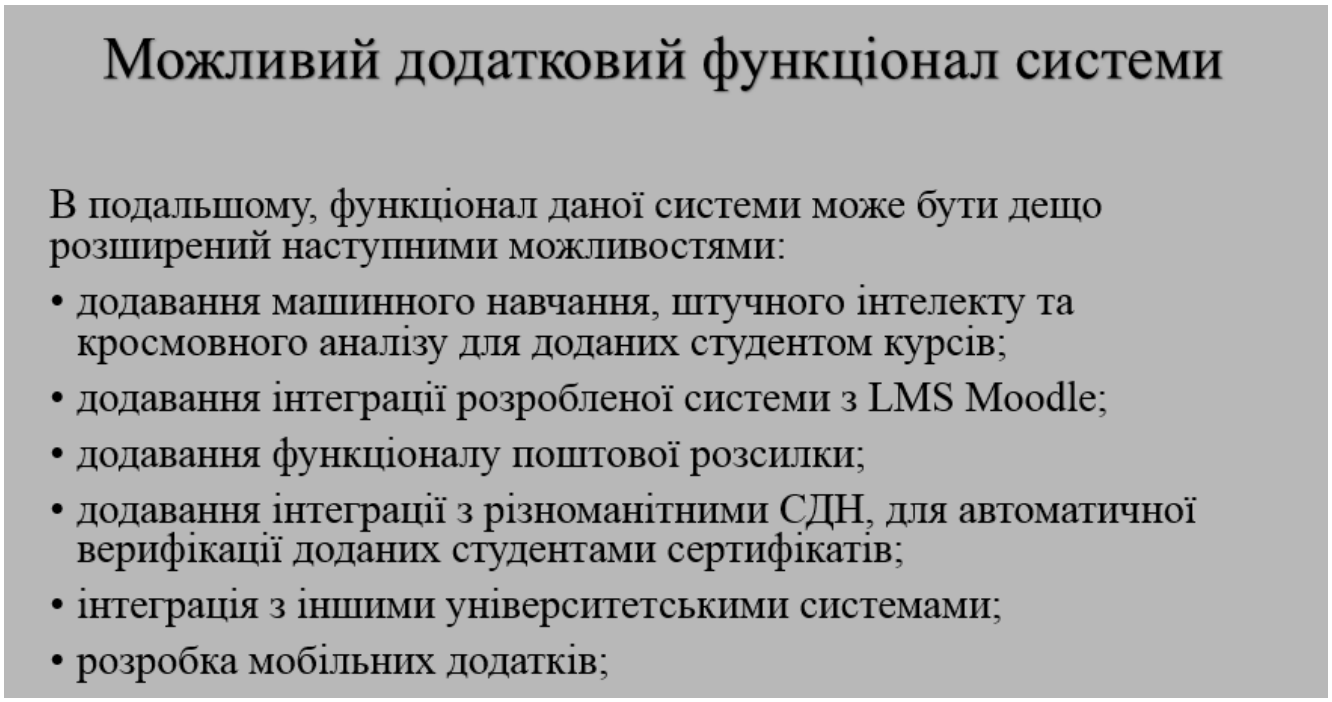


Рисунок Б.16- Слайд №16

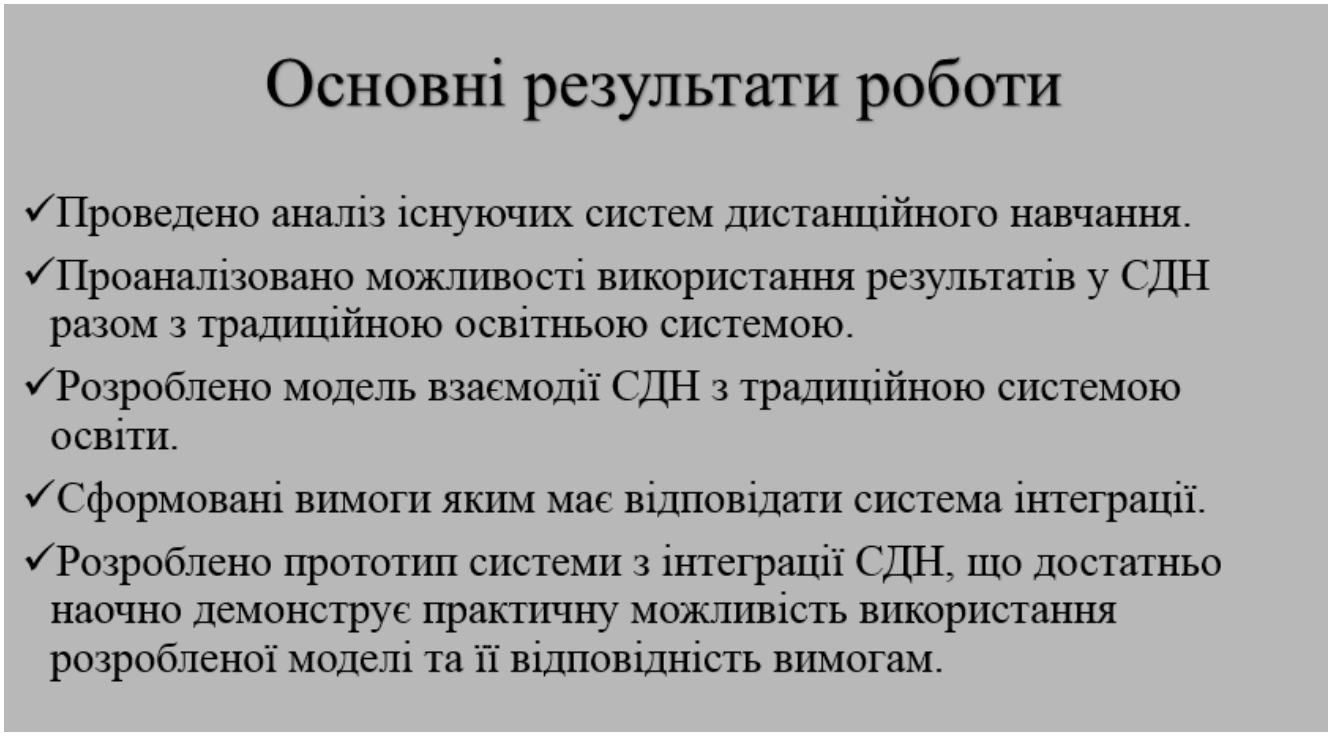


Рисунок Б.17- Слайд №17

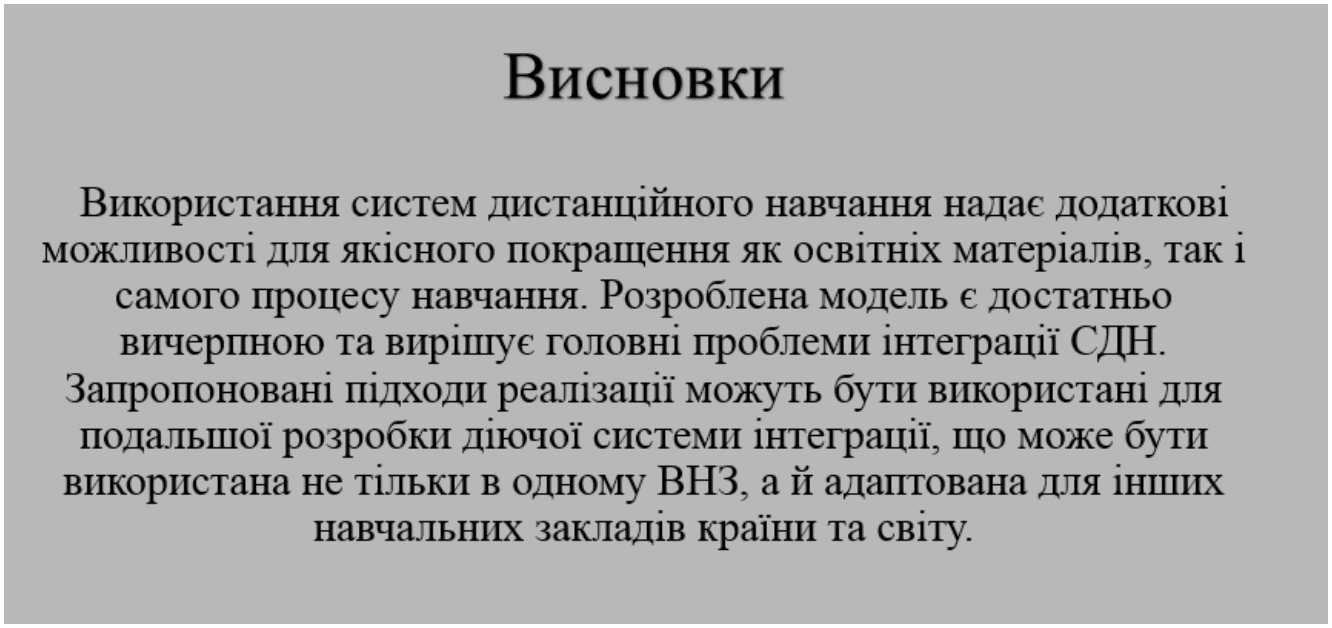


Рисунок Б.18- Слайд №18