Міністерство освіти і науки України
Луцький національний технічний університет
Факультет комп'ютерних та інформаційних технологій
Кафедра інженерії програмного забезпечення

КВАЛІФІКАЦІЙНА РОБОТА
ЗА СТУПЕНЕМ ВИЩОЇ ОСВІТИ «БАКАЛАВР»

ІНФОРМАЦІЙНА СИСТЕМА ОСОБИСТОГО КАБІНЕТУ СТУДЕНТА

INFORMATION SYSTEM OF THE STUDENT'S PERSONAL ACCOUNT

спеціальність 121 Інженерія програмного забезпечення
(шифр і назва спеціальності)

освітня програма «Інженерія програмного забезпечення»
(назва освітньої програми)

Виконав: здобувач вищої освіти
Групи ІПЗс-31
Климовець Дмитро Вадимович
_____
(підпис)

Керівник:
к.т.н., доцент
Повстяна Юлія Славомирівна
_____
(підпис)

Кваліфікаційну роботу
допущено до захисту
«8» 06 2023р.
Гарант освітньої програми:
к.т.н., доцент
Ліщина Наталія Миколаївна
_____
(підпис)

Луцьк – 2023 року

ЛУЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

Факультет *комп'ютерних та інформаційних технологій*
Кафедра *інженерії програмного забезпечення*
Ступінь вищої освіти: *бакалавр*
Галузь знань: *12 Інформаційні технології*
Спеціальність: *121 Інженерія програмного забезпечення*
Освітня програма: *«Інженерія програмного забезпечення»*

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____
«02» 02 2023 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧУ ВИЩОЇ ОСВІТИ

Климовцю Дмитру Вадимовичу
(прізвище, ім'я, по-батькові)

1. Тема кваліфікаційної роботи *Інформаційна система особистого кабінету студента*

Керівник роботи: *Повстяна Юлія Славомирівна*, к.т.н., доцент _____
затверджені наказом закладу вищої освіти від «23» грудня 2022 р. № 961-01-02
2. Строк подання здобувачем вищої освіти кваліфікаційної роботи «8» червня 2023 р.
3. Вихідні дані до роботи: *постановка завдання на кваліфікаційну роботу, мова програмування Python, фреймворк Django, система управління базами даних SQLite*

4. Зміст розрахунково-пояснювальної записки (перелік питань, що потрібно розробити):
аналіз предметної області, визначення вимог до програмного забезпечення, аналіз інструментів та технологій, проектування та розробка програмного забезпечення

5. Перелік графічного матеріалу:
рис. 28

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис	
		Завдання видав	Завдання прийняв
Аналіз предметної області	*Повстяна Ю.С.*		
Специфікація вимог до розробленої системи	*Повстяна Ю.С.*		
Розробка об'єкта проектування	*Повстяна Ю.С.*		
Нормоконтроль	*Повстяна Ю.С.*		
Гарант ОП	*Ліщина Н.М*		
Показник запозичень тексту	5,4 %		
Академічна доброчесність	*Ящук А.А.*		

7. Дата видачі завдання «2» лютого 2023 р.

КАЛЕНДАРНИЙ ПЛАН

No з/п	Назва етапів кваліфікаційної роботи бакалавра	Терміни виконання	Примітка
1	*Огляд літературних джерел по темі кваліфікаційної роботи бакалавра*	*05.02.2023 р.*	вик.
2	*Аналіз проблем розробки та впровадження об'єкту проектування*	*27.02.2023 р.*	вик.
3	*Обґрунтування вибору шляхів, технологій (алгоритмів) і засобів вирішення поставленого завдання*	*10.03.2023 р.*	вик.
4	*Розробка функціонально-структурної схеми роботи об'єкта проектування та проектування бази даних*	*17.04.2023 р.*	вик.
5	*Практична реалізація об'єкта проектування та розробка бази даних*	*18.05.2023 р.*	вик.
6	*Тестування та налагодження об'єкта проектування*	*01.06.2023 р.*	вик.
7	*Здача чистового варіанту кваліфікаційної роботи бакалавра на кафедру*	*08.06.2023 р.*	вик.

Здобувач вищої освіти ______ Климовець Д.В.
(підпис) (прізвище, ініціали)

Керівник кваліфікаційної роботи ______ Повстяна Ю.С.
(підпис) (прізвище, ініціали)

Міністерство освіти і науки України
Луцький національний технічний університет

Рецензія
на кваліфікаційну роботу

Здобувач вищої освіти:
Климовець Дмитро Вадимович

Тема:
«Інформаційна система особистого кабінету студента»

Коротка характеристика кваліфікаційної роботи:
Кваліфікаційна робота на тему «Інформаційна система особистого кабінету студента» представляє собою детальне дослідження та розробку інформаційної системи, яка спрямована на поліпшення навчального процесу та взаємодії між студентами та викладачами. Автор аналізує потреби студентів та використовує передові методи розробки, зокрема щодо архітектури системи та захисту даних. В результаті виконання кваліфікаційної роботи було створено інформаційну систему особистого кабінету студента.
Самостійні розробки і пропозиції автора:
Автор демонструє здатність до самостійної роботи та креативного мислення. Він розробив власну архітектуру системи, виходячи з аналізу потреб студентів. Також автор пропонує інноваційні рішення щодо захисту конфіденційної інформації та забезпечення безпеки даних студентів. Це свідчить про глибоке розуміння проблематики та здатність автора до створення практичних рішень.
Практичне значення роботи:
Кваліфікаційна робота має велике практичне значення для вищого навчального закладу та студентів. Інформаційна система особистого кабінету студента, розроблена автором, спрощує навчальний процес, забезпечує зручний доступ до особистої інформації та успішності. Вона дозволяє студентам більш ефективно організувати навчальний процес.
Недоліки:
Незважаючи на загально позитивну оцінку кваліфікаційної роботи, слід зазначити деякі недоліки. Перш за все, робота могла бути більш глибокою щодо аналізу потреб і вимог студентів до системи, а також у розробці функціональних можливостей. Також, автор міг би розглянути більше альтернативних рішень та провести їх порівняльний аналіз для досягнення більш оптимального результату.
Загальний висновок:
Кваліфікаційна робота на тему «Інформаційна система особистого кабінету студента» є добре виконаною роботою, яка вирішує актуальну проблему. Автор продемонстрував високий рівень професійної компетентності, знання передових методів розробки та здатність до самостійної роботи. Робота має велике практичне значення та може бути використана у вищих навчальних закладах для поліпшення процесу навчання студентів. З огляду на недоліки, рекомендується подальше вдосконалення роботи шляхом уточнення деяких аспектів та розширення дослідження. Загальною оцінкою роботи є "добре".

Рецензент: Ткачук Анатолій Анатолійович, к.т.н., доцент
(прізвище, ім'я, по-батькові, посада)

Рецензент кваліфікаційної роботи _____
(підпис)

«01» червня 2023 р.

АНОТАЦІЯ

Климовець Д.В. Інформаційна система особистого кабінету студента. Рукопис.

Кваліфікаційна робота бакалавра ОП «Інженерія програмного забезпечення». Луцький національний технічний університет. Луцьк, 2023.

Кваліфікаційна робота присвячена розробці інформаційної системи особистого кабінету студента.

Робота складається з трьох основних підрозділів, в яких проаналізовано актуальність сучасного стану проблем і поставлено завдання відповідно до теми роботи, проаналізовано інформацію про аналоги системи, розглянуто основні вимоги до інформаційної системи та функціональних можливостей. Для розробки системи було використано сучасні інструменти та технології, а саме: фреймворк Django та база даних SQLite. Також дипломний проект є вагомим внеском в розвиток інформаційних систем у навчальному процесі та надасть практичну користь застосування та взаємодію студентам та викладачам.

Ключові слова: *інформаційна система*, *особистий кабінет студента*, *Python*, *Django*.

ABSTRACT

Klymovets D.V. Information System of the Student's Personal Account. - Manuscript.

Bachelor thesis of EP «Software Engineering». Lutsk National Technical University. Lutsk, 2023.

Qualification work, dedicated to the development of the information system of the student's personal office.

The work consists of three main divisions, in which the relevance of the current state of problems is analyzed and tasks are set according to the topic of the work, information about system analogues is analyzed, and the main requirements for the information system and functional capabilities are clarified. Modern tools and technologies were used to develop the systems, namely: the Django framework and the SQLite database. Also, the diploma project is important beyond the development of information systems in the educational process and will provide practical application and interaction to students and teachers.

Keywords: *information system*, *student's personal account*, *Python*, *Django*.

ЗМІСТ

ВСТУП

Актуальність теми. На сьогоднішній день інформаційні технології відіграють дуже велику роль у житті людей. Актуальною проблемою у сучасному світі є автоматизація навчального процесу в закладах вищої освіти, що надає можливість студентам і викладачам більш ефективніше взаємодіяти між собою, також, слідкувати за навчальною успішністю та навчальним процесом в цілому.

Важливою особливістю автоматизації навчального процесу є інформаційна система особистого кабінету студента, яка надає можливість студентам за короткий час отримати відомості про навчальний процес і керувати ними. Інформаційна система включатиме в себе інформацію про особисті дані користувача, відомість про успішність користувачів з різних навчальних предметів, інформацію про навчальний процес та інші матеріали.

Метою дослідження кваліфікаційної роботи бакалавра є розробка інформаційної системи особистого кабінету студента, написаного на веб-фреймворку Django, що надасть можливість забезпечення ефективної взаємодії між студентами та викладачами, що натомість забезпечить швидкий і зручний доступ до особистої інформації користувачів та навчальний процес.

Об’єктом дослідження кваліфікаційної роботи бакалавра є інформаційна система особистого кабінету студента, що включає функціональні модулі для управління навчальним процесом, успішності студентів, даними користувачів та спілкування на курсах.

Предметом дослідження кваліфікаційної роботи бакалавра є реалізація функціоналу інформаційної системи особистого кабінету студента, що включає в себе: систему аутентифікації та авторизації, управління контентом та даними користувачів, перегляд успішності та доступних предметів, взаємодія між викладачами та студентами у чатах курсів. Розробка зручного інтерфейсу, що дозволить краще взаємодіяти студентам з системою та забезпечить зручність

використання. Забезпечення конфіденційності та безпеки користувацьких даних, при використанні різноманітних механізмів шифрування і захисту.

Однією з основних проблем в галузі автоматизації навчального процесу, які вирішує дана розробка, є недостатня автоматизація та інформатизація закладів вищою освіти, також недостатньо розвинений уніфікований підхід до навчального процесу, тому розробка інформаційної системи особистого кабінету відповідає актуальності сучасного стану проблеми і вимогам до навчального процесу.

РОЗДІЛ 1
АНАЛІЗ ІНФОРМАЦІЙНОЇ СИСТЕМИ ОСОБИСТОГО КАБІНЕТУ СТУДЕНТА І ПОСТАНОВКА ЗАВДАНЬ НА КВАЛІФІКАЦІЙНУ РОБОТУ

1.1 Аналіз сучасного стану проблеми

З постійним розвитком інформаційних технологій навчальні заклади переходять на електронні платформи, щоб збільшити взаємодію між студентами та викладачами. Такі зміни відбуваються в університетах, тому що в сучасному світі дистанційне навчання набирає популярності. Сучасні ідеї сприяють створенню постійних змін в сфері навчання, взаємодії студентів та викладачів, що спонукає університети створювати власні електронні кабінети та постійно їх покращувати.

На сьогоднішній день електронні кабінети є популярним сервісом з погляду навчання, оскільки вони є багатофункціональними. Університети все частіше переходять від електронних порталів міжнародного споживання до власних електронних кабінетів, які створені спеціально під конкретні навчальні заклади та синхронізовані з інформацією про студентів та їхню успішність. Електронні кабінети охоплюють великий спектр застосування під час навчання для покращення роботи, взаємодії студентів та викладачів, і розвитку власного бренду.

До основних можливостей електронних кабінетів входить наступний функціонал:

– перегляд особистої інформації та навчання;

– перегляд успішності студента;

– перегляд списку наявних курсів;

– запис на курси з детальною інформацією;

– листування в чаті курсу.

Для розробки інформаційної системи потрібно провести аналіз серед аналогів, щоб уникнути недоліків, яких допустили конкуренти. Для аналізу існуючих аналогів було обрано наступні електронні сервіси [1]:

– Google Classroom;

– Moodle;

– Електронний кабінет студента ЛНТУ;

– Source LMS;

Google Classroom – безкоштовний онлайн сервіс для навчальних закладів зі спрощеною системою поширення завдань. Метою даного сервісу є прискорення процесу поширення файлів серед учасників курсу (рис. 1.1, рис. 1.2).

Даний сервіс створений компанією «*Google*» та вміщує в собі наступні сервіси: Google Docs, Sheets, Slides, Forms, Drive і Calendar. Щоб стати учасником курсу потрібно отримати запрошення від адміністратора курсу або ввести приватний код курсу.

Переваги Google Classroom:

– в Classroom інтегровано багато сервісів Google для комфортної роботи в середині сервісу;

– файли студентів та викладачів знаходяться безпосередньо в Google Drive курсу;

– студенти та викладачі можуть додавати файли до завдання та залишати коментарі;

– можливість викладача публікувати оголошення для студентів;

– можливість архівування курсу в кінці року або семестру;

– робота у веб-сервісі та мобільному додатку;

– відсутність реклами та повна приватність від неї.

Недоліки Google Classroom:

– реєстрація лише через поштовий сервіс «Gmail»;

– обмежена кількість учасників навчального процесу;

– обмежене пам’ять сховища для зберігання файлів;

– потрібно навчати викладачів користуватися даним застосунком;

– функціонал системи не є повністю зрозумілим для користування;

– недостатня інформація про студентів та викладачів.

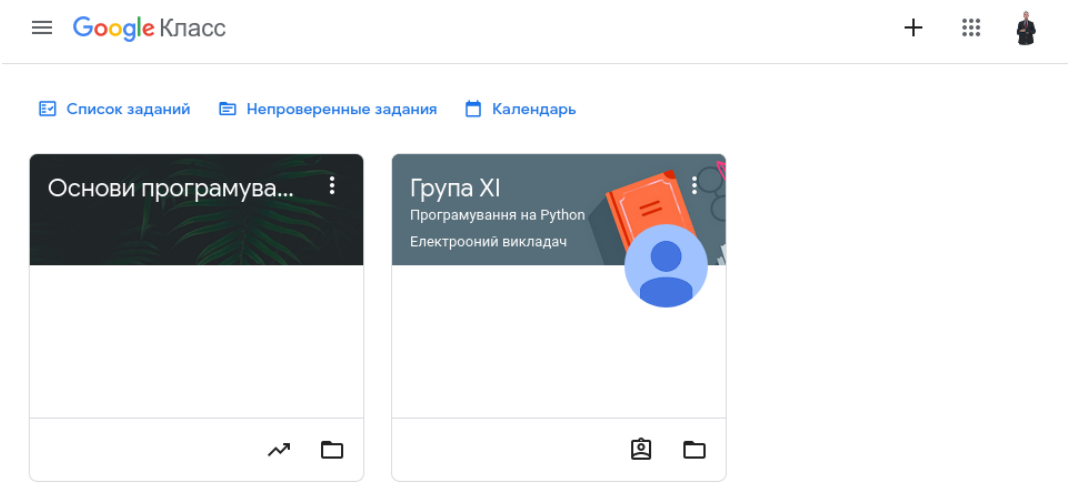

Рисунок 1.1 – Курси на Google Classroom

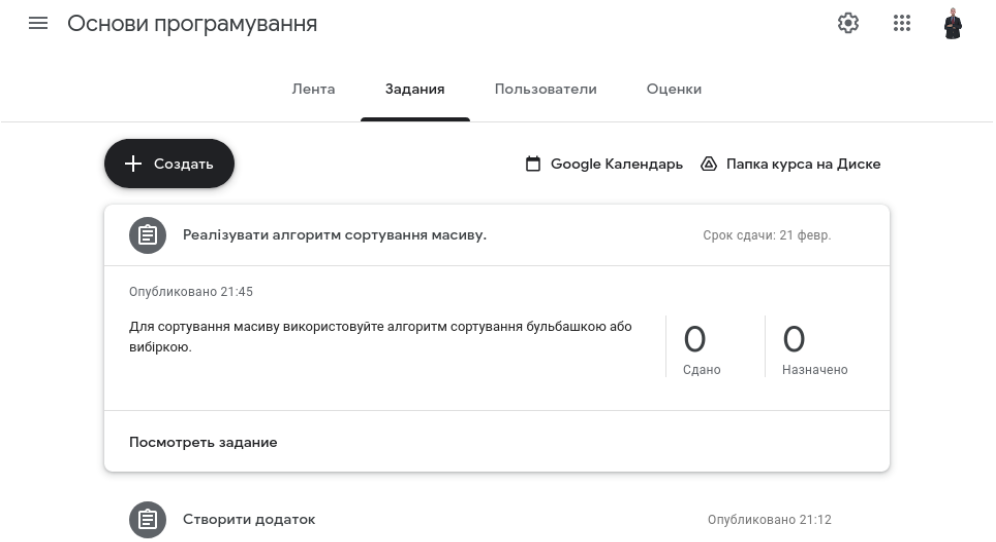

Рисунок 1.2 – Завдання на курсі

Moodle – навчальна платформа для створення персоналізованого навчального середовища [2]. Платформа є безкоштовною, з відкритим програмним забезпеченням, також орієнтована на створення взаємодії між студентами та викладачами (рис. 1.3, рис 1.4).

Переваги Moodle:

– поділ користувачів на типи, як: викладачі та студенти;

– запис, перегляд інформації та вмісту на курсі;

– розміщення навчального матеріалу різних форматів (для викладачів);

– можливість розміщення сторонніх скриптів та плагінів;

– зміна методів зарахування на курси та методів аутентифікації;

– система оцінювання та здачі навчальних робіт;

– проведення тестувань з налаштуванням часу та питань.

Недоліки Moodle:

– складний інтерфейс системи для функціонал для звичайних користувачів;

– складне адміністрування і налаштування системи.

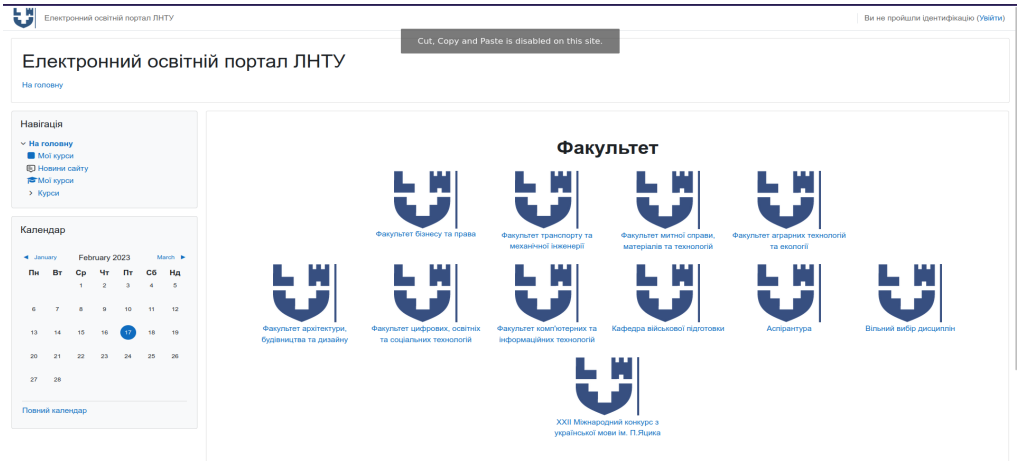

Рисунок 1.3 – Головна сторінка

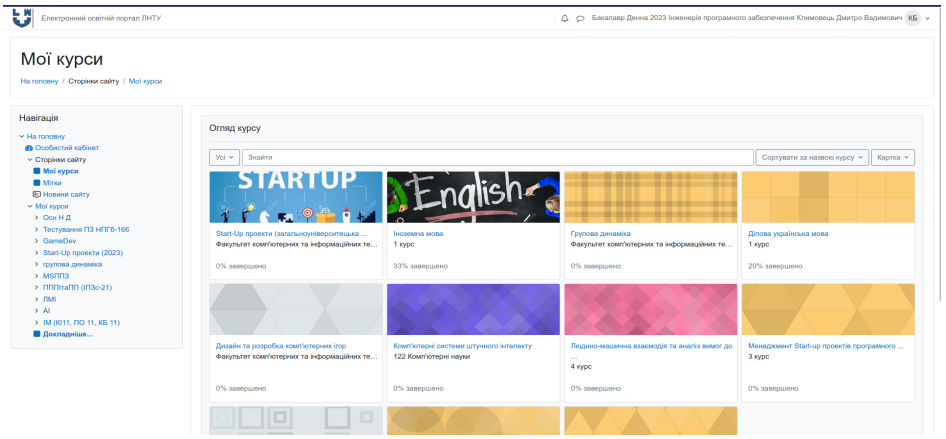

Рисунок 1.4 – Курсами в Moodle

Електронний кабінет студента ЛНТУ – веб-сервіс навчального закладу з особовими даними та успішністю студента. Даний сервіс призначений спеціально для університету (рис. 1.5, рис 1.6).

Переваги електронного кабінету студента ЛНТУ:

– перегляд особистої інформації про навчання;

– перегляд розкладу занять з інформацією про викладача;

– перегляд власної успішності за кожен предмет;

– запис на доступні предмети;

– проходження опитувань.

Недоліки електронного кабінету студента ЛНТУ:

– відсутня можливість змінювати логін/електронну пошту;

– неповна інформація про предмет в розділі «успішність»;

– прив’язаність під конкретний навчальний заклад;

– відсутність системи чатів та повідомлень.

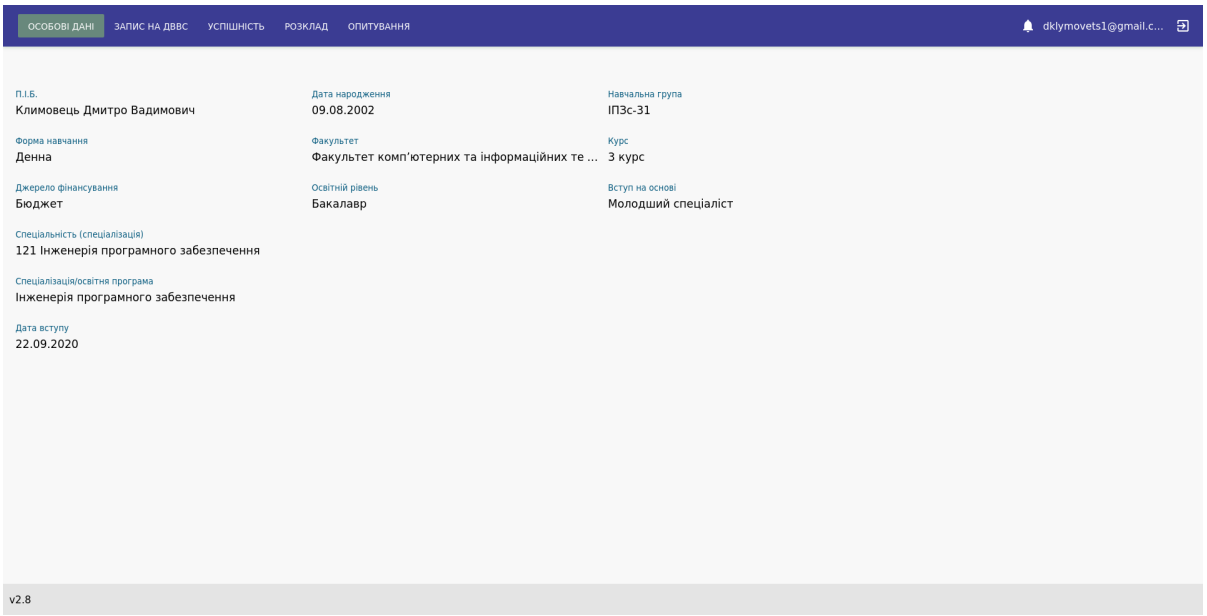

Рисунок 1.5 – Інформація про студента

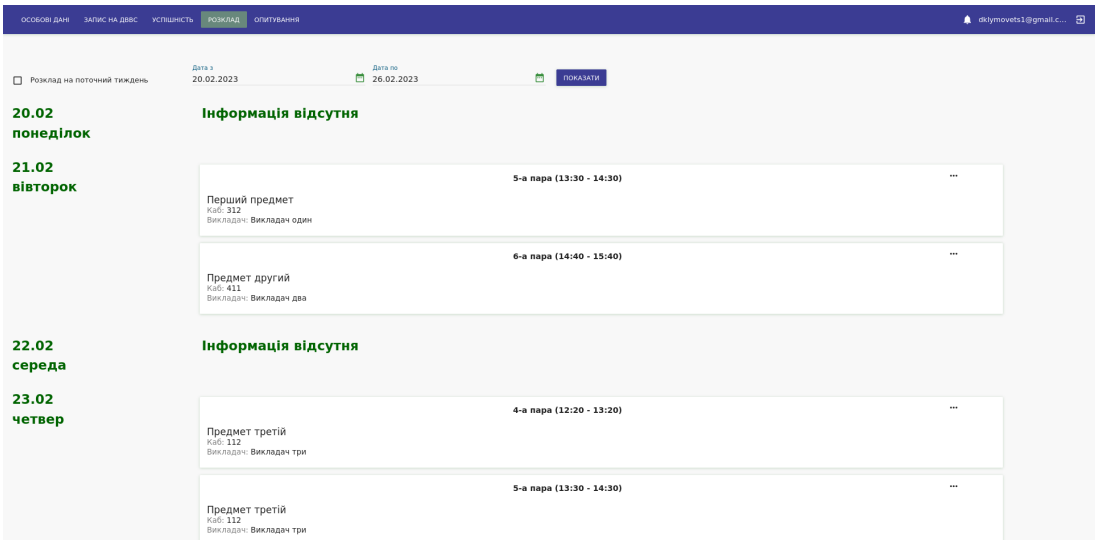

Рисунок 1.6 – Розклад занять

Source LMS – сервіс дистанційного навчання, що дозволяє управляти навчальними процесами онлайн [3]. Серед можливостей системи виділяють наступне: проведення дистанційних занять, запис лекції та демонстрація екрану, створення групових чатів. Також сервіс пропонує управляти ролями викладачів та студентів, переглядати структуру навчальних курсів (рис. 1.7, рис. 1.8).

Переваги Source LMS:

– курси доступні лише приєднаним користувачам;

– можливість прикріплювати файли до чатів;

– інтерактивна дошка, яка краще дозволяє наводити приклади;

– проведення онлайн тестувань з можливістю автоматичної перевірки;

– наявність групових та приватних чатів;

– простий та зручний інтерфейс системи;

– наявність конструктора курсів, що дозволяє швидко створювати курси.

Недоліки Source LMS:

– система доступна лише у платному варіанті;

– обмежений перегляд файлів в браузері без попереднього завантаження;

– відсутній календар для планування завдань та визначення термінів;

– відсутній поділ студентів на групи всередині курсу.

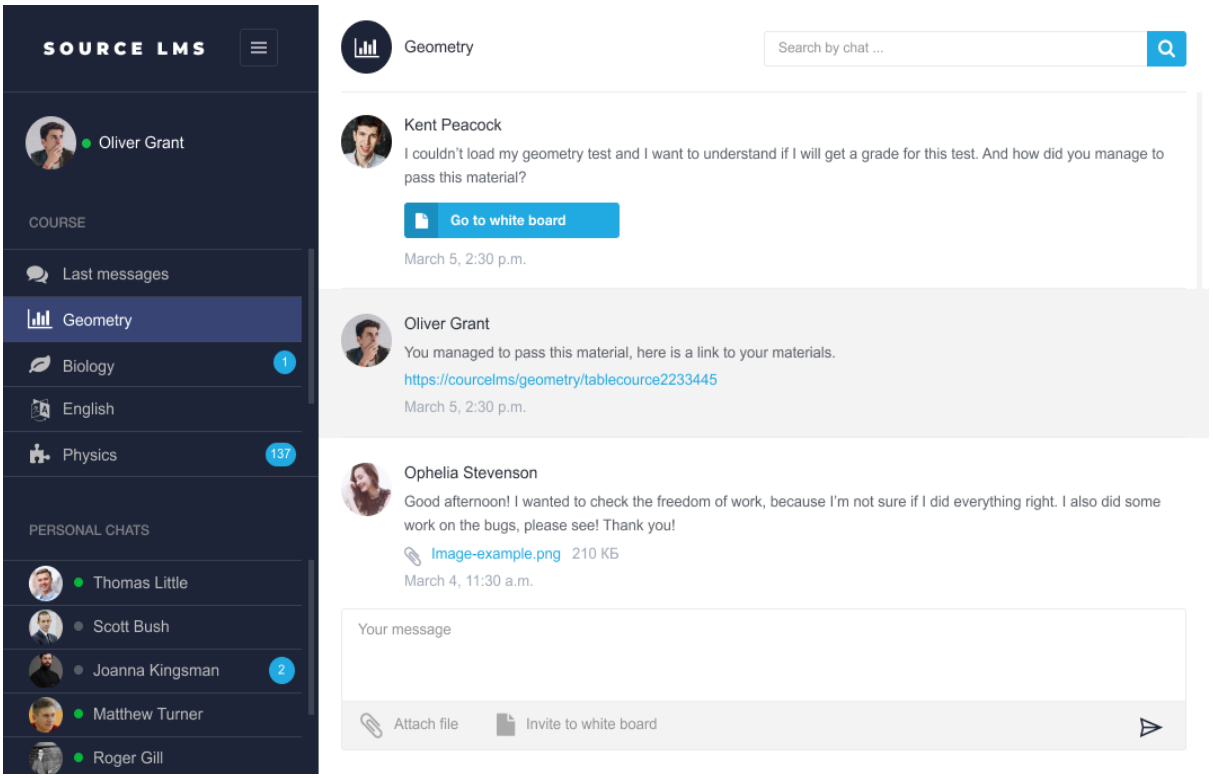

Рисунок 1.7 – Приватний чат предмету

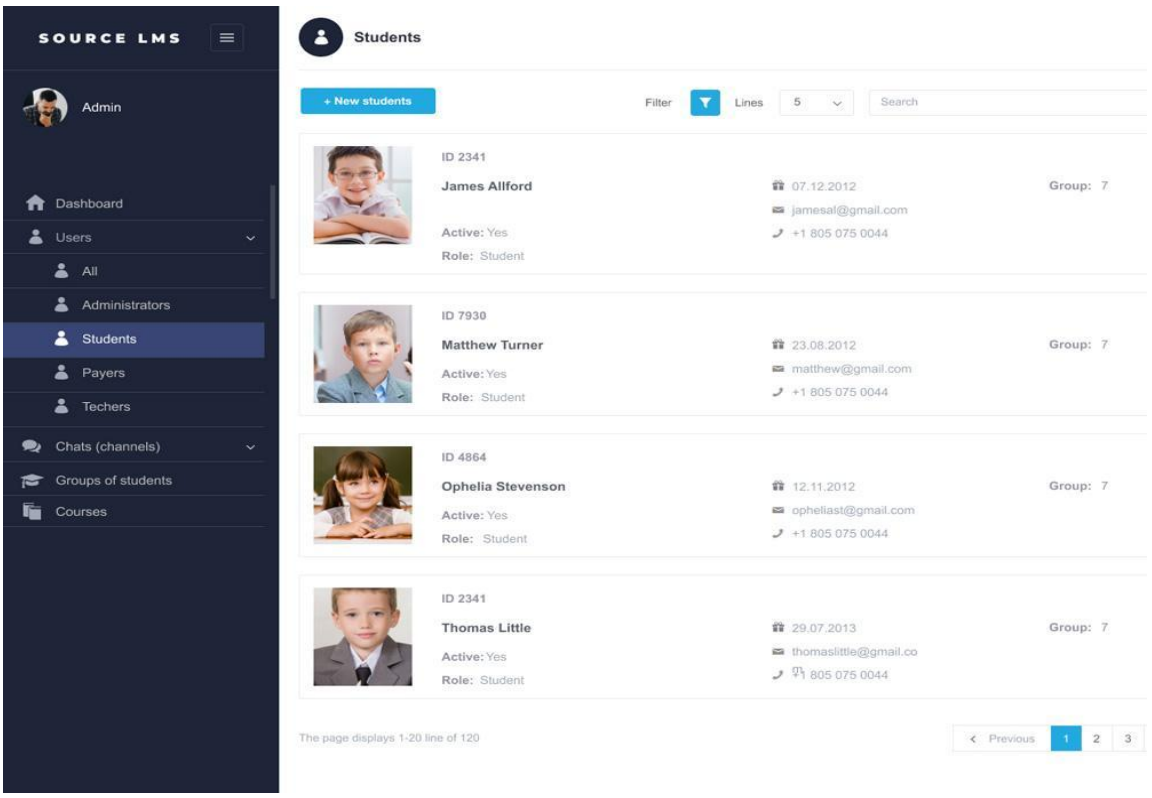

Рисунок 1.8 – Адміністративна панель

Вище описані аналоги інформаційної системи мають свої переваги, які слід взяти до уваги під час розробки проекту, та недоліки, яких слід уникати. Завдяки недолікам та деяким недостатнім функціоналом потрібно розробити нову інформаційну систему, яка охопить майже весь наявний функціонал, та розробить новий, якого немає у аналогів.

1.2 Постановка завдання на кваліфікаційну роботу бакалавра

Метою даної роботи є створення інформаційної системи особистого кабінету студента. Користувачі матимуть можливість переглядати особисту інформацію, власну успішність, інформацію про курси та навчальний процес. Система матиме зрозумілий та приємний україномовний інтерфейс.

Для досягнення кінцевої мети необхідно виконати наступний список завдань:

– проведення аналізу актуальності інформаційної системи;

– складання вимог до програмного забезпечення;

– створення прототипу інтерфейсу інформаційної системи;

– вибір технології та засобів реалізації для створення проекту;

– проектування системи у нотації UML;

– налаштування середовища розробки і технологій;

– проектування бази даних для проекту;

– написання коду для проекту:

  – створення системи аутентифікації та авторизації;

  – реалізація функціоналу системи, відповідно до завдання;

  – розробка бази даних.

Висновки до розділу 1

Проаналізувавши отриману інформацію про технології електронних кабінетів, ознайомлення з їхнім функціоналом та оглядом існуючих аналогів, можна зробити висновок, що основними недоліками більшості аналогів є:

– незрозумілий функціонал та складний інтерфейс системи;

– недостатня інформація про існуючі курси;

– відсутність можливості студенту самому записатися на курс.

Можна зробити висновок, що доцільно створити нову систему електронного кабінету, яка охопить достатню частину функціоналу аналогів та усуне їхні недоліки.

РОЗДІЛ 2
СПЕЦИФІКАЦІЯ ВИМОГ ДО РОЗРОБЛЮВАНОЇ СИСТЕМИ

2.1 Аналіз, визначення вимог до розроблюваного програмного забезпечення та проектування програмного забезпечення

Для виконання завдання з моделювання інформаційної системи потрібно вибрати конкретний тип моделі. Вибір моделі залежить від конкретної мети, яку прагне досягти кінцевий продукт. Для аналізу було обрано декілька типів моделей, а саме:

– модель класів – описує, які класи об’єктів містить система та зв’язки між ними;

– модель послідовності – описує послідовності подій, що відбуваються в системі при виконанні певної функції;

– модель компонентів – описує наявність компонентів в цій системі та взаємодію між компонентами;

– модель прецедентів – описує взаємодію користувачів з системою;

– модель станів – описує стани, в яких може перебувати система та її переходи між ними.

Для виконання завдання з моделювання системи було обрано модель послідовності, оскільки ця модель допоможе зрозуміти, як працюватимуть окремі функції інформаційної системи [4].

Важливим етапом проектування інформаційної системи є наведення специфікацій вимог до програмного забезпечення. Вимоги поділяються на: функціональні та нефункціональні [5].

Функціональні вимоги:

– інформаційна система повинна мати систему авторизації, автентифікації та реєстрації;

– система має забезпечувати користувачів з типом «Студент» можливістю запису на курси та перегляд власної успішності;

– система має забезпечувати користувачів з типом «Викладач» можливістю створення курсів та виставлення оцінок;

– система має надавати можливість вносити зміни до особистої інформації користувача;

– система повинна забезпечувати можливість розміщення оголошень для користувачів конкретного курсу.

Нефункціональні вимоги:

– система повинна залишатися доступною для використання цілодобово;

– система повинна мати захист від несанкційного доступу до інформації;

– система повинна надавати гнучку і надійну швидкість роботи;

– система повинна бути сумісною з різними операційними системами;

– система повинна бути простою і зрозумілою для користувачів.

Одним з найважливіших етапів проектування інформаційної системи є опис моделі елементів у нотації UML [6], які включають класи і діаграми послідовності (рис. 2.1) та прецедентів (рис. 2.2).

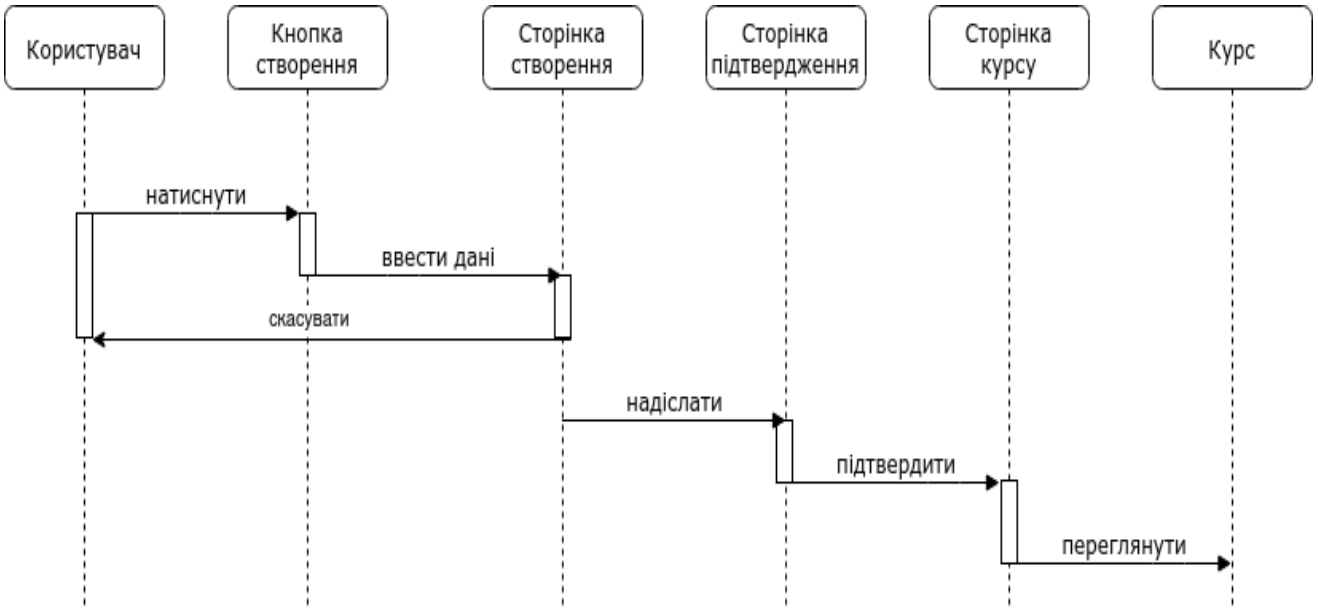

Рисунок 2.1 – Діаграма послідовності створення курсу

Рисунок 2.2 – Діаграма прецедентів викладача та студента

Для більш точного розуміння приводяться цілі та задачі інформаційної системи. Основними цілями інформаційної системи особистого кабінету студента є:

– забезпечення ефективного збору, зберігання і обробки даних;

– забезпечення конфіденційності і безпеки даних;

– забезпечення можливості взаємодії між студентами та викладачами через чати курсів;

– забезпечення можливості завантаження даних;

– забезпечення можливості переглядати відповідні дані користувачам.

Основні задачі, які вирішує інформаційна система:

– збір інформації про користувачів та реєстрація профілю;

– забезпечення можливості запису на курси;

– забезпечення можливості внутрішнього чату курсу для взаємодії між студентами і викладачами;

– забезпечення можливості адміністрування курсів;

– забезпечення безпеки і конфіденційності від несанкційного доступу до даних користувачів та системи;

– забезпечення доступу до перегляду інформації про курс та успішність;

– забезпечення можливості виставляти оцінки студентам.

Основні вимоги до продукту особистого кабінету студента включають в себе: функціональні вимоги, вимоги до продуктивності, вимоги до надійності і безпеки, вимоги до дизайну інтерфейсу.

Функціональні вимоги:

– можливість авторизації, автентифікації та реєстрації користувачів;

– можливість внесення змін до персональної інформації користувачів;

– можливість створення та редагування інформації про курси;

– можливість запису на курс та перегляду інформації про нього;

– можливість переглядати власну успішність;

– можливість виставляти оцінки студентам на власних курсах.

Вимоги до продуктивності:

– можливість швидкого завантаження сторінок з даними;

– мінімальний час відповіді на запит користувача;

– можливість надійного та безпечного з’єднання для передачі даних.

Вимоги до надійності і безпеки:

– забезпечення конфіденційності та цілісності даних;

– можливість відновлення інформації у разі їх втрати;

– наявність захисту від несанкційного доступу до даних користувачів та системи.

Вимоги до дизайну інтерфейсу:

– наявність графічних компонентів та інтерактивних функцій;

– забезпечення одноманітності компонентів на різних розмірах екранів;

– зручний і простий у використанні інтерфейс системи.

Для здійснення UX/UI-дизайну використовувався векторний онлайн редактор Figma, який дозволяє створювати прототипи дизайну інтерфейсу [7]. Під час розробки дизайну було визначено з яких блоків буде складатися сторінка з курсами (рис. 2.3):

– навігаційна панель, яка містить в собі інтерактивні кнопки;

– панель авторизації з іменем та типом користувача;

– повна, коротка назви курсу та опис;

– ім’я автора курсу [викладача];

– кількість годин та кредитів за проходження курсу;

– кнопка запису на курс, якщо користувач не записаний на нього.

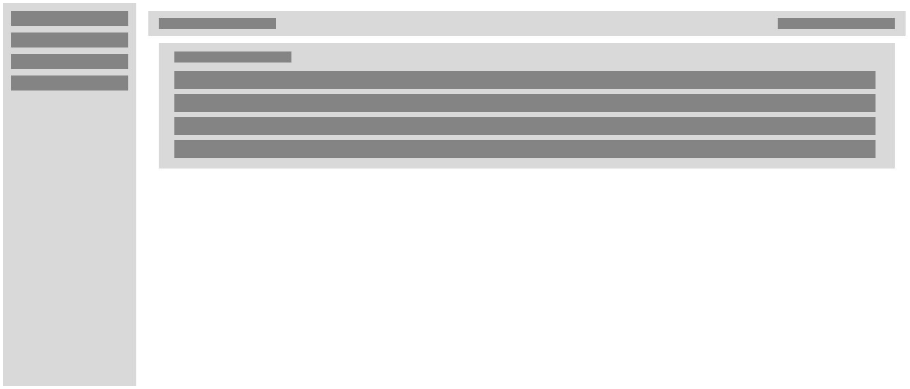

Рисунок 2.3 – UX-дизайн профілю користувача

Для створення прототипу сторінки з курсами потрібно вибрати палітру кольорів та визначитися з відступами між елементами, що дозволить краще сприймати інтерфейс [8]. Для самого проектування було створено фрейм розміром для «Full HD» екранів та безпосередньо розміщено елементи та текст з кольорами, відповідно до UX-дизайну. Роботу з редактором Figma зображено на рисунку 2.4.

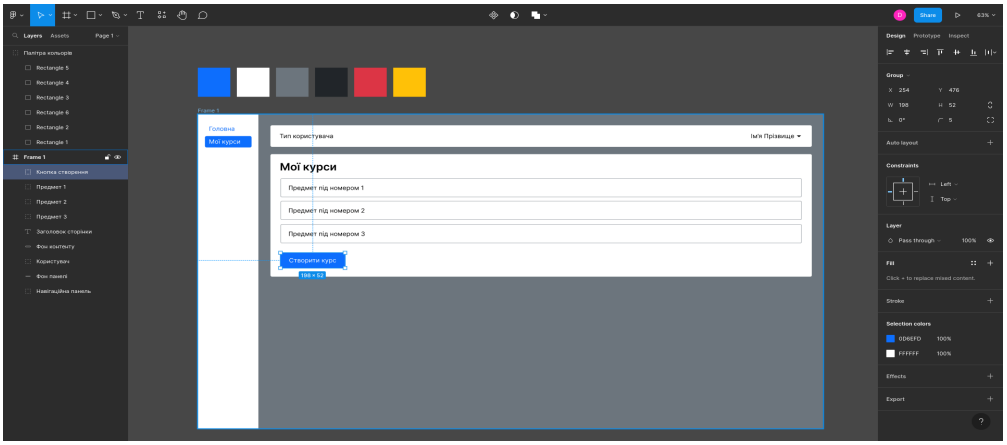

Рисунок 2.4 – UI-дизайн сторінки з курсами у редакторі Figma

Також було створено прототип сторінки з курсом, де розміщено елементи та таблицю зі студентами та їхніми балами за допомогою вбудованих конструкцій редактора Figma, що дозволило не створювати нові елементи, а використовувати попередньо розроблені, що прискорило процес проектування UI-дизайну. Під час проектування сторінки курсу було задіяно декілька основних елементів, які включають: фон контенту, текст з підписами, кнопки з кольорами, які відповідають діям, і таблиця, що містить текст з кнопками [8]. Приклад сторінки зображено на рисунку 2.5.

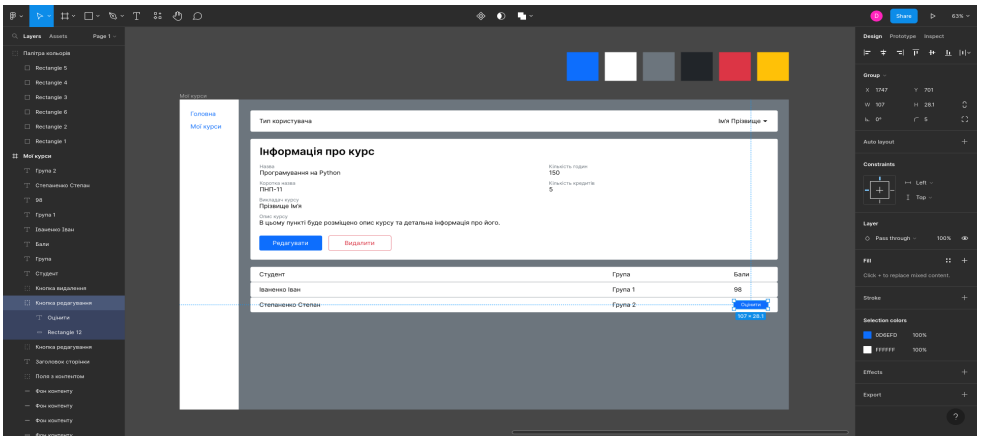

Рисунок 2.5 – UI-дизайн сторінки курсу у редакторі Figma

Завдяки використанню вбудованих конструкцій веб-редактору Figma вдалося прискорити процес проектування UI-дизайну системи в декілька разів [8]. Було використано компоненти, які є своєрідним блоком, який можна створити один раз, і використовувати в різних частинах дизайну, змінивши лише текст або колір копії. Вигляд компонентів зображено на рисунку 2.6.

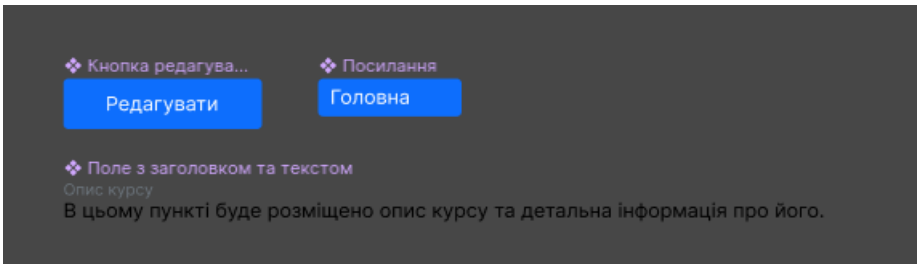

Рисунок 2.6 – Приклад UI-компонентів Figma

2.2 Вибір засобів, методів і алгоритмів вирішення поставленого завдання

Для обрання засобів та методів для розробки інформаційної системи було розглянуто такі технології, як: PHP, ASP.NET, Django і Java.

PHP – це скриптова мова програмування, яка була створена для генерації HTML-сторінок на стороні веб-серверу та є популярною для розробки веб-додатків [9]. Дана мова має наступні можливості:

– підтримує бази даних MySQL, SQLite, PostgreSQL, Oracle та багато інших;

– має велику кількість вбудованих функцій і модулів;

– дозволяє створювати динамічний контент, який може бути залежний від параметрів запиту.

Переваги PHP:

– безкоштовна і відкрита мова програмування;

– дозволяє швидко і ефективно створювати веб-додатки;

– підтримує сторонні плагіни для розширення функціональності;

– є легкою для вивчення новачкам.

Недоліки PHP:

– має проблеми з безпекою без використання сторонніх сервісів;

– проблеми з продуктивністю при великій кількості запитів;

– наявність недоліків у дизайні мови, що призводить до складнощів при програмування на високому рівні.

ASP.NET – це веб-фреймворк для написання динамічних веб-додатків, розроблений компанією Microsoft [9]. Ця технологія має наступні можливості:

– підтримка мов програмування C#, F#, Visual Basic;

– можливість масштабування та високої продуктивності;

– наявність вбудованих засобів безпеки та автентифікації;

– підтримка баз даних MySQL, SQL Server, Oracle;

– можливість використання технологій Web Forms, MVC та Web Pages.

Переваги ASP.NET:

– розробка у середовищі розробки з великою кількістю різноманітних інструментів;

– надійність і безпека у вбудованих засобів та технологій;

– наявність великої кількості бібліотек для розширення функціональності додатків;

– широка документація та підтримка від Microsoft.

Недоліки ASP.NET:

– пропрієтарний фреймворк, який може призвести до високих витрат на ліцензії;

– технологія використовує велику кількість пам’яті, що може призвести до великих затрат по ресурсам серверу;

– розгортання додатку може бути більш складним та вимагати велику кількість ресурсів сервера.

Python Django – це високорівневий фреймворк, призначений для написання веб-додатків на мові програмування Python [10], також має наступні можливості:

– має велику кількість вбудованих модулів і бібліотек;

– має вбудовану адміністративну панель;

– має вбудований механізм кешування, що підвищує продуктивність;

– надає підтримку і захист від CSRF атак;

– надає можливість створення REST API;

– підтримує бази даних MySQL, SQLite, PostgreSQL та інші;

– підтримує формати виведення даних: JSON, HTML, XML.

Переваги Django:

– наявність легкої та великої документації;

– можливість швидкого розгортання додатку на сервері;

– наявність вбудованої системи авторизації та автентифікації;

– можливість легкого розширення функціональності додатку;

– наявність вбудованого ORM, що дозволяє працювати з базою даних;

– має велику спільноту розробників.

Недоліки Django:

– недостатня гнучкість технології;

– недостатня швидкість додатку;

– відсутність підтримки асинхронності, що є проблемою для додатків, які потребують високу продуктивність;

– технологія може вимагати багато пам’яті при обробці великої кількості даних;

– може бути складним через налаштування додатку новачку.

Java – це об’єктно-орієнтована мова програмування, призначена для розробки веб-додатків, десктопних та мобільних додатків. Технологія має наступні можливості:

– можливість працювати на різних операційних системах;

– є об’єктно-орієнтованою мовою, що дозволяє створювати і підтримувати складні системи;

– наявність великої бібліотеки для швидкої розробки;

– наявність високого рівня безпеки.

Переваги Java:

– надійна мова програмування для створення якісних додатків;

– мова з високою продуктивністю;

– наявність великої спільноти розробників і стандартів;

– наявність розвиненої системи виключень і обробок помилок.

Недоліки Java:

– наявність доволі високого рівня абстракції, через що втрачається продуктивность;

– технологія вимагає багато ресурсів системи;

– у наявності велика кількість ключових слів і правил, що робить код важким;

– можливість проблем з сумісністю між версіями, що викликає помилки при роботі на різних платформах.

Для реалізації інформаційної системи особистого кабінету студента було обрано наступний стек технологій:

– Python Django;

– SQLite;

– Bootstrap 5.

Django – це безкоштовний фреймворк для розробки веб-додатків на мові програмування Python, який забезпечує високорівневий інтерфейс для роботи з базою даних [11]. Також підтримує модель MVC для проектування веб-додатків, містить вбудований ORM та систему аутентифікації і реєстрації.

Фреймворк має в наявності вбудовану адміністративну панель, яка дозволяє здійснювати управління в базі даних без використання додаткових плагінів. Також в наявності Django є широкий набір бібліотек та модулів, що дозволяє розширювати функціонал веб-додатків і прискорювати розробку [12].

Django використовує URL адреси, які визначають, який функціонал повинен виконуватись для кожного запиту. Також фреймворк використовує вбудований механізм кешування, який зменшує час відповіді на запити, особливо корисний при повторних запитах.

Фреймворк Django має великий спектр функціональних можливостей [13]:

– Django ORM – вбудована технологія, що дозволяє працювати та керувати базами даних MySQL, SQLite, PostgreSQL та інші [14];

– URL-адреси – що дозволяє визначати адреси та пов’язаний з ними функціонал, і дозволяє зберігати логіку додатків в окремих модулях;

– шаблони – фреймворк має зручний механізм роботи з шаблонами, що дозволяє створювати веб-сторінки та легко взаємодіяти між ними;

– автентифікація – має вбудовану систему автентифікації та авторизації, що робить додаток захищеним від несанкційного доступу;

– адміністративна панель – фреймворк надає вбудовану адміністративну панель, що дозволяє взаємодіяти з даними в базі даних без використання додаткових плагінів;

– кешування – вбудований механізм кешування, що дозволяє зберігати дані в пам’яті та зменшує час відповіді на запити;

– міжпроцесорна комунікація – дозволяє взаємодіяти з іншими додатками та сервісами одного проекту;

– розширення функціональності – має набір вбудованих бібліотек та модулів, що дозволяють розширювати функціональність додатків.

Методика вирішення задач використовуючи Django може включати декілька етапів, а саме:

– аналіз вимог – проводиться аналіз вимог до проекту, визначається його функціональні та нефункціональні вимоги, і бізнес-процеси;

– проектування архітектури – проводиться розробка архітектури проекту, структури бази даних та розробка моделі даних, і визначення функціональності проекту;

– розробка функціоналу – проводиться реалізація проекту:: розроблення моделей (Model), представлень (View), шаблонів (Template) та іншого функціоналу;

– тестування додатку – проводиться тестування системи, функціональності, безпеки та продуктивності;

– реліз та супровід – проводиться після виконання вищезазначених етапів і випускається в експлуатацію, після чого йде супровід з підтримкою, оновленням та розширенням функціоналу.

Також для створення багатофункціональних додатків можна використовувати різні набори інструментів, які надає фреймворк, такі як система маршрутизації, система кешування і система аутентифікації та авторизації. Окрім цього, фреймворк Django має велику спільноту розробників, яка надає доступ до

багатьох користувацьких модулів і бібліотек, що полегшують та пришвидшують розробку веб-додатку.

Для кращого розуміння проектованої системи можна розглянути алгоритм доступу до курсів (рис. 2.7), який обмежує несанкційний доступ до даних і повертає користувача на сторінку авторизації або реєстрації.

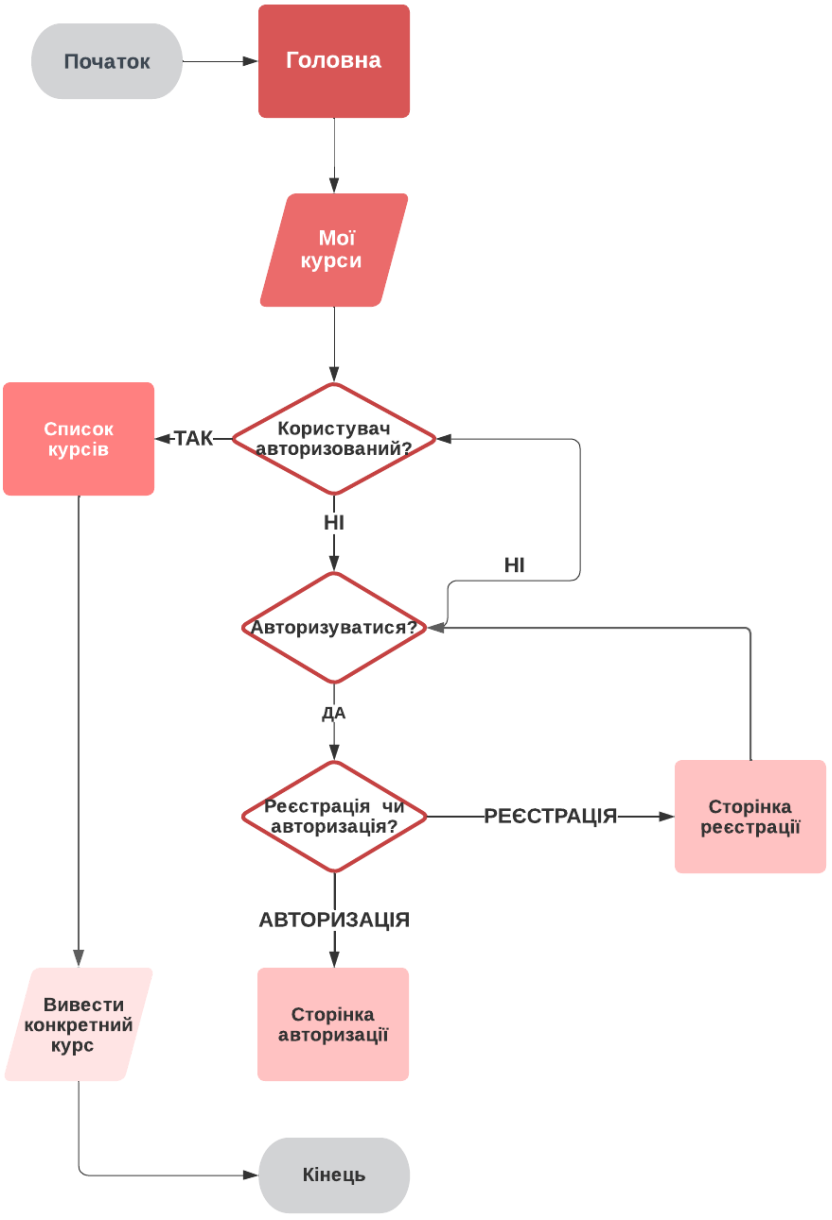

Рисунок 2.7 – Блок-схема алгоритму доступу до курсів

Висновки до розділу 2

У даному розділі було розглянуто конкретну модель для моделювання інформаційної системи та її елементів, специфікацію вимог до програмного забезпечення і функціональну складову системи у нотації UML.

Також, розглянуто існуючі інструменти розробки системи, які можуть бути використані для вирішення задач, та проаналізовано їх можливості, переваги і недоліки.

Отже, можна зробити висновок, що було обрано конкретний тип моделі для моделювання системи та конкретний інструмент і технології для вирішення поставлених задач.

РОЗДІЛ 3
РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ ОСОБИСТОГО КАБІНЕТУ СТУДЕНТА

3.1 Практична реалізація об’єкта проектування

Для створення додатку на Django фреймворку потрібно використовувати пакетний менеджер Python під назвою ‘*pip*’ та віртуальне середовище розробки ‘venv’ для коректної роботи усіх файлів та модулів.

Для початку потрібно створити віртуальне середовище, і лише після цього потрібно встановити Django фреймворк потрібної версії за допомогою команд, описаних в лістингу 3.1.

Лістинг 3.1 – Встановлення віртуального середовища та Django

```
python3 -m venv ‘cabinet’
pip install django
```

кінець лістингу 3.1

Після виконання вищеописаних команд потрібно створити проект з базовими файлами для функціонування системи в цілому за допомогою команди, описаної в лістингу 3.2.

Лістинг 3.2 – Створення проекту

```
django-admin startproject ‘cabinet’
```

кінець лістингу 3.2

Після виконання команди буде створено проект під назвою ‘cabinet’ та декількома файлами для повноцінного функціонування проекту, які складаються з: файлу налаштувань, головного файлу URL-адрес та іншими файлами для розгортання на сервері.

Для написання бізнес-логіки проекту потрібно створити додаток, який міститиме файли: представлення, моделей, форм, URL-адрес та

адміністративного управління. Для створення додатку використовується команда, описана в лістингу 3.3.

Лістинг 3.3 – Створення додатку

```
django-admin startapp ‘main’
```

кінець лістингу 3.3

Для написання бізнес-логіки додатку використовується файл ‘views.py’, де на початку потрібно написати просту функцію для відображення інформації на головній сторінці, описану в лістингу 3.4.

Лістинг 3.4 – Код функції головної сторінки

```
from django.shortcuts import render
def index(request):
    return render(request, ‘main/base.html’, {})
```

кінець лістингу 3.4

Наступним кроком є створення логіки профілю користувача, де відбувається безпосередній запит до бази даних по відповідного користувача, і перевіряється його тип, описаний в лістингу 3.5.

Лістинг 3.5 – Код запиту до бази даних по користувача

```
from django.shortcuts import render
from user.models import User
from .models import Teacher, Student
def profile_view(request):
    user = User.objects.get(id=request.user.id)
    profile = None
    try:
        if user.type == ‘ST’:
            profile = Student.objects.get(user=user.id)
```

```
            if user.type == ‘TR’:
                profile = Teacher.objects.get(user=user.id)
        except ObjectDoesNotExist:
            profile = None
    context = {‘user’: user,  ‘profile’: profile}
    return render(request, ‘main/profile-view.html’, context)
```

кінець лістингу 3.5

Одним з важливих елементів проекту є курси або ж предмети, які відіграють важливу роль в проекті. Нижче наведено приклад реалізації створення курсу для відповідного типу користувача, з введенням назви, опису, оцінок та балів, які можно отримати на курсі, описаний в лістингу 3.6.

Лістинг 3.6 – Код створення курсу

```
from .forms import CourseForm
def course_create(request):
    if request.user.type == ‘ST’:
        return redirect(‘/course/’)
    if request.method == ‘POST’:
        form = CourseForm(request.POST)
        if form.is_valid():
            course = Course.objects.create(
                name=form.cleaned_data[‘name’],
                short_name=form.cleaned_data[‘short_name’],
                description=form.cleaned_data[‘description’],
                teacher=request.user,
                hours=form.cleaned_data[‘hours’],
                points=form.cleaned_data[points’],
            )
            return redirect(f’/course/{course.id}/’)
```

```
        else:
            form = CourseForm()
        return redirect(request, ‘main/course/create.html’, {‘form’: form})
```

кінець лістингу 3.6

Також одним з важливих елементів проекту є опис адміністративної панелі, для прикладу, описується модель курсів. Для початку потрібно зареєструвати модель в файлі ‘admin.py’, після чого, створити класс з відповідною назвою, де відбувається присвоєння полів для списку відображення, списку фільтрів, відображення полів у запису та порядок сортування за полем, описаний в лістингу 3.7.

Лістинг 3.7 – Код адміністративної панелі курсів

```
from django.contrib import admin
from .models import Course
@admin.register(Course)
class CourseAdmin(admin.ModelAdmin):
    list_display = ('short_name', 'teacher', 'hours', 'points',)
    list_filter = ('short_name', 'teacher', 'hours', 'points',)
    fieldsets = (
        (None, {‘fields’: (‘name’, ‘short_name’,)}),
        (“Course information”, {‘fields’: (
            ‘teacher’, ‘description’, ‘hours’, ‘points’,
        )})
    )
    search_fields = (‘short_name’, ‘teacher’,)
    ordering = (‘short_name’,)
```

кінець лістингу 3.7

Головними над всіма процесами в додатку є суперкористувачі, які виконують роль адміністраторів та мають доступ до:

– додавання, редагування та видалення даних з бази даних;

– зміни типу користувачів та їх статусу;

– блокування даних та перегляд історії над ними.

Створення суперкористувача відбувається за допомогою команди в терміналі, з попередньо, визначеними полями та даними, які описані в окремому додатку, де користувачу надаються спеціальні права та можливості, а саме:

– користувачу присвоюється відмітка «Команда проекту»;

– користувачу присвоюється статус «Суперкористувач»;

– користувачу присвоюється статус «Активний»;

– права та доступ до управління базою даних.

Алгоритм створення суперкористувача описується в лістингу 3.8, де відбувається надання статусів та перевіряється, чи є вони активованими.

Лістинг 3.8 – Алгоритм створення суперкористувача

```
class UserManager(BaseUserManager):
    def create_superuser(self, email, password, **extra_fields):
        extra_fields.setdefault('is_staff', True)
        extra_fields.setdefault('is_superuser', True)
        extra_fields.setdefault('is_active', True)

        if extra_fields.get('is_staff') is not True:
            raise ValueError(_('Superuser must have is_staff=True'))
        if extra_fields.get('is_superuser') is not True:
            raise ValueError(_('Superuser must have is_superuser=True'))
        return self.create_user(email, password, **extra_fields)
```

кінець лістингу 3.8

3.2 Тестування та налагодження інформаційно-комп’ютерної системи

Для проведення тестування та налагодження системи потрібно скласти мінімальні системні вимоги для повноцінного функціонування додатку, які складаються з:

– операційна система: Windows 10 / Ubuntu 18.04 / MacOS 10;

– процесор: Intel Core i3 або вище;

– оперативна пам’ять: 4 Гб;

– версія Python: 3.8 або вище;

– версія Django: 4.1 або вище;

– версія Pillow: 9.4.2 або вище;

– версія бази даних SQLite: 3.0 або вище.

Отримавши перелік мінімальних системних вимог можна переходити до перевірки правильності встановлення веб-фреймворку Django та інших потрібних для функціонування модулів. Здійснити перевірку можна за допомогою команди в терміналі «pip freeze», що дасть нам наступний результат, описаної в лістингу 3.9.

Лістинг 3.9 – Алгоритм створення суперкористувача

```
asgiref==3.6.0
betterpath==0.2.2
click==8.1.3
python==3.11.1
Django==4.1.5
django-crispy-forms==1.14.0
l==0.11.0
sqlparse==0.4.3
vcversioner==2.16.0.0
zope.interface==5.5.2
```

кінець лістингу 3.9

Після отриманих результатів можна дійти висновку, що для повноцінної роботи веб-додатку на Django було встановлено всі потрібні бібліотеки та модулі, і перевірено їх працездатність.

Наступним кроком буде перевірка функціональності додатку, шляхом створення запитів до бази даних, за допомогою Django ORM. Щоб працювати з Django ORM потрібно у терміналі запустити команду «*python3 manage.py shell*».

Потрібно перевірити взаємодію між таблицями «Студенти» та «Курси» за допомогою команд, описаних в лістингу 3.10.

Лістинг 3.10 – Запит на отримання списку студентів

```
from main.models import Student, Course
first_course = Course.objects.get(id=1)
all_students = Student.objects.filter(course=first_course)
```

кінець лістингу 3.10

Після виконання вищеописаних команд було отримано список студентів, які записані на курс під номером «1». Провівши ще декілька тестів дійшов до висновку, що взаємодія між таблицями та працездатність повністю функціонує.

Також варто перевірити можливість створення нового курсу, та знайти невідповідності, які можуть виникнути в ході створення, описано в лістингу 3.11.

Лістинг 3.11 – Запит на створення курсу

```
from main.models import Teacher, Course
teacher = Teacher.objects.get(id=1)
new_course = Course.objects.create(
        name=”Новий предмет”, short_name=”НП-11”,
        description=”Інформація про предмет”,
        teacher=teacher.id, hours=150, points=5
)
```

кінець лістингу 3.11

Після виконання запиту до бази даних було створено новий курс під назвою «Новий предмет», де викладачем був користувач під номером «1». Тому, можна дійти висновку, що можливість створення нових записів до бази даних повністю функціонує.

Для тестування представлень та моделей використовувалося Unit-тестування, за допомогою якого, було протестовано окремі модулі і компоненти незалежно від іншого функціоналу системи [15].

Також для перевірки безпеки інформаційної системи було проведено тести на SQL-ін’єкції і CSRF-атаки, після яких не було виявлено жодних помилок.

3.3 Розробка бази даних

Розробка бази даних є однією з найважливіших частин інформаційної системи, що передбачає створення структури бази даних, яка містить таблиці, поля, типи даних та їхні відносини [16]. Для розробки бази даних потрібно спроектувати її, де буде описано таблиці з полями, зображено на рисунку 3.1.

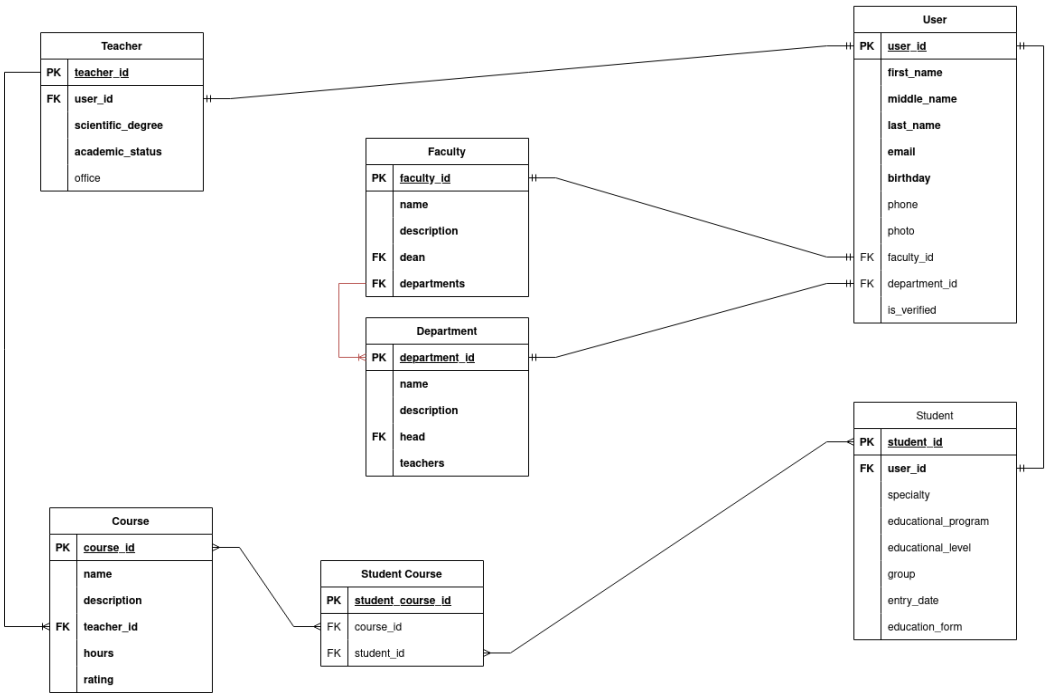

Рисунок 3.1 – Діаграма бази даних системи

Для створення бази даних та запитів до таблиць використовується Django ORM, яка містить запити у вигляді коду SQLite [17]. Для створення бази даних було використано команду, описану в лістингу 3.12.

Лістинг 3.12 – Команда для створення бази даних

```
sqlite3 cabinet_database.db
python3 manage.py dbshell
```

кінець лістингу 3.12

Для створення таблиці в базі даних з назвою «Курс» використовується запит для створення таблиць, який міститиме інформацію про назву, опис, викладача та інші дані, описані в лістингу 3.13.

Лістинг 3.13 – Запит на створення таблиці ‘Курс’

```
CREATE TABLE courses (
    id INTEGER PRIMARY KEY,
    name TEXT NOT NULL,
    description TEXT NOT NULL,
    teacher FOREIGN KEY,
    hours INTEGER,
    point INTEGER
);
```

кінець лістингу 3.13

Також, для повноцінного функціонування таблиці потрібно заповнити її відповідною інформацією, зробити це можна за допомогою запиту на додавання даних до таблиці, описаного в лістингу 3.14.

Лістинг 3.14 – Запит на додавання даних до таблиці «Курс»

```
INSERT INTO courses (id, name, description, teacher, hours, point)
VALUES
    (1, ‘Архітектура ПЗ’, ‘Відсутнє’, 2, 150, 5),
```

```
(2, ‘Розробка ігор’, ‘Розробка ефектів та анімації’, 3, 150, 4),
(3, ‘Архітектура комп’ютера’, ‘Інформація’, 1, 180, 5),
(4, ‘Веб-дизайн’, ‘Робота з редактором’, 4, 180, 5),
(5, ‘Бази даних’, ‘Розробка баз даних’, 5, 150, 3);
```

кінець лістингу 3.14

Для викладача потрібним функціоналом є відображення курсів, які йому належать, для цього потрібно створити відповідний запит, який міститиме фільтр з полем «teacher_id», описаний в лістингу 3.15. В кінцевому результаті буде отримано список курсів, які належать викладачу, вказаному в атрибуті «teacher_id».

Лістинг 3.15 – Запит на вибірку курсів

```
SELECT name, description
FROM courses
WHERE teacher_id=2;
```

кінець лістингу 3.15

Важливим елементом функціонування додатку є відображення об’єднання таблиць «Факультет» та «Кафедра», де буде отримано список кафедр, які належать відповідним факультетам, за допомогою запиту з методом об’єднання таблиць, описаного в лістингу 3.16.

Лістинг 3.16 – Запит на об’єднання таблиць

```
SELECT faculties.name, departments.name
FROM faculties
JOIN departments
ON faculties.id = departments.faculty_id;
```

кінець лістингу 3.16

3.4 Захист інформаційно-комп’ютерної системи

Важливою частиною інформаційної системи є робота механізмів, які відповідають за авторизацію, шифрування передачі даних, також захист від різного виду атак та хешування паролів [18].

Основними механізмами додатку є автентифікація та авторизація, що виконують ідентифікацію користувача, відповідно до даних в базі даних. Для більш точного розуміння цих механізмів потрібно розглянути їх поняття.

Автентифікація – це процес перевірки користувача на наявність його даних в базі даних, тобто чи є він дійсно тим, за кого себе видає. Зазвичай, такий процес виконується за допомогою логіну, яким може бути електронна пошта, та паролю, тоді здійснюється перевірка правильності даних у базі даних, і повертається успішний результат, якщо дані в базі збігаються. Принцип роботи механізму автентифікації користувача зображено на рисунку 3.2.

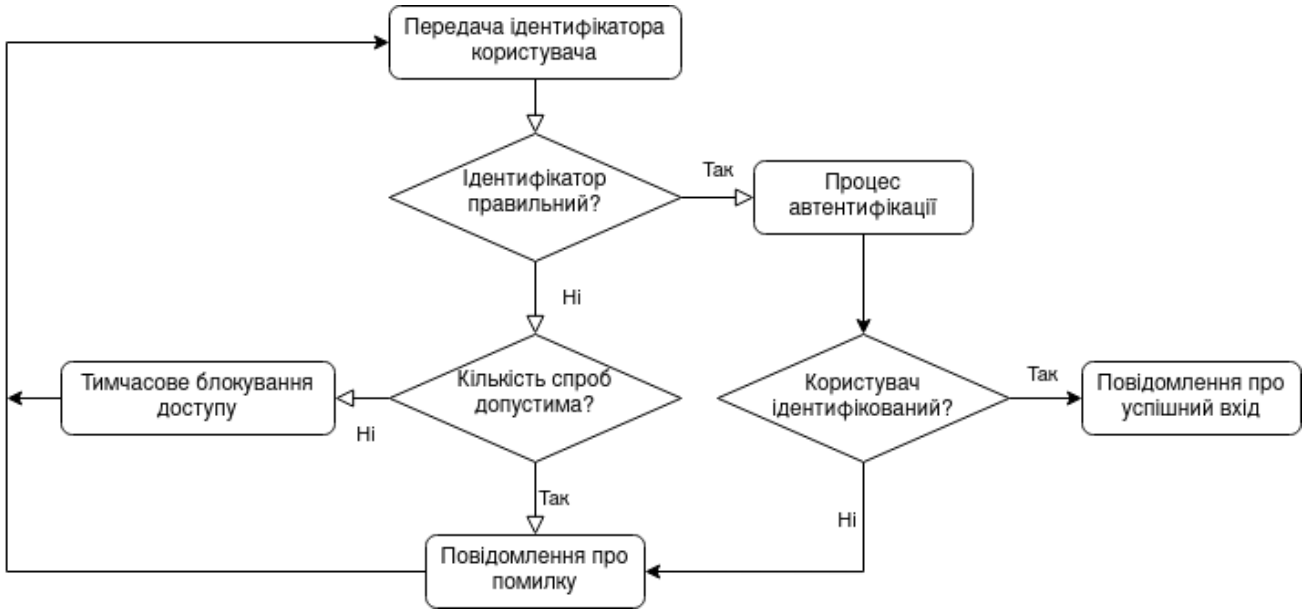

Рисунок 3.2 – Схема роботи механізму аутентифікації

Авторизація – це процес перевірки автентифікованого користувача на наявність прав доступу до певного контенту та даних в додатку. У даній

інформаційній системі такий процес здійснюється за допомогою прав доступу, які поділяють користувачів на декілька типів. Також, типи користувачів налаштовані на отримання доступу до даних лише по відповідних правах, які їм належать, а якщо він не має таких прав, то не зможе отримати до них доступ.

При налаштуванні додатку, фреймворк Django може використовувати систему сесій для механізму аутентифікації та авторизації [19], що дозволить створити сесію для користувача і тимчасово зберігати дані на сервері, які пов’язані з цим користувачем.

Для автентифікації користувача в систему було створено окремий додаток для реєстрації та автентифікації, який містить форму для введення даних та модулі, по яких здійснюється безпосередній вхід в систему, описано в лістингу 3.17.

Лістинг 3.17 – Форма аутентифікації та реєстрації

```
from django.contrib.auth import login, authenticate
from django.contrib.auth.forms import UserCreationForm
from user.models import User
class RegisterForm(UserCreationForm):
    class Meta:
    model = User
    fields = ('last_name', 'first_name', 'middle_name', 'email', 'password1', 'password2',)
```

кінець лістингу 3.17

Завдяки методу «authenticate» з використанням форми реєстрації можна створити користувача, якщо надіслана форма пройшла валідацію, та дані відповідають правилам механізмів.

Ще одним важливим механізмом є шифрування передачі даних, яке відповідає за безпеку системи. Для забезпечення безпеки Django використовує протокол HTTPS, який використовує сертифікати SSL або покращену версію

TSL, що допомагає шифрувати дані, які передаються між сервером та браузером. Також цей протокол забезпечує конфіденційність і цілісну передачу даних, що запобігає їх перехопленню.

Сертифікати SSL та TSL використовуються для встановлення безпечного з'єднання, при якому сервер і клієнт обмінюються ключами шифрування, які використовуються для шифрування та розшифрування даних, зображеними на рисунку 3.3. Для забезпечення безпеки шифрування даних потрібно провести налаштування серверу з відповідним сертифікатом, що може бути зроблено з використанням вбудованого серверу Django або іншого серверу, такого як Nginx чи Apache.

Рисунок 3.3 – Алгоритм шифрування передачі даних
за допомогою ключів шифрування

Крім цього, фреймворк Django дозволяє налаштувати систему для безпечної роботи з кукі, які дозволяють зберігати дані авторизації користувача та шифрувати їх від підробки.

Важливим механізмом Django-додатків є механізм хешування, що призначений для безпечного зберігання паролів у базі даних. Для точного розуміння даного механізму потрібно розглянути його поняття та особливість роботи.

Хешування паролів – це процес перетворення паролів в хеш-фукнцію, яка унеможливлює перетворити пароль в початковий стан [20]. Особливість роботи механізму хешування полягає у хешуванні паролю при створенні облікового запису та зберіганні його в базі даних, тоді, при наступній автентифікації,

введений користувачем пароль хешується і порівнюється з хешем, що зберігається в базі даних. Якщо хеші співпадають, то пароль правильний і користувачу надається доступ до даних.

Ще однією особливістю хешування є використання солі [20]. Сіль – це випадкова послідовність символів, яка додається до пароля перед хешуванням. Перед процесом хешування, відповідна функція приймає два параметри, а саме: байтовий рядок паролю та сіль. Використання солі забезпечує надійність та безпеку хеш-функції, тому, що після хешування два однакові паролі будуть мати різний хеш через використання різних солей. Для кращого розуміння структури захешованого паролю можна розглянути рисунок 3.4, де описується структура, яка містить алгоритми, сіль та хеш паролю.

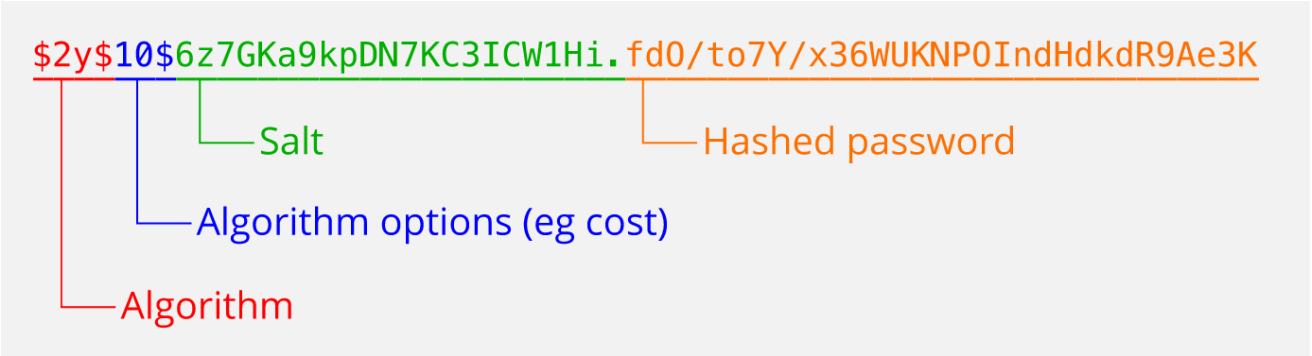

Рисунок 3.4 – Захешований пароль з використанням солі

Для хешування паролю в Django використовуються спеціальні вбудовані бібліотеки, які хешують пароль та додають сіль, щоби зробити хеш унікальним. Модулі, що використовуються, описані в лістингу 3.18.

Лістинг 3.18 – Модулі хешування паролю в Django

```
PASSWORD_HASHERS = [
    'django.contrib.auth.hashers.PBKDF2PasswordHasher',
    'django.contrib.auth.hashers.PBKDF2SHA1PasswordHasher',
]
```

кінець лістингу 3.18

Механізм хешування паролю за допомогою вбудованих бібліотек Django працює наступним чином: стандартний алгоритм PBKDF2 використовує декілька ітерацій хешування, щоб уповільнити та ускладнити атаку зловмисників на хешовані паролі, описаний в лістингу 3.19. Також у даного механізму є можливість перевизначення кількості ітерацій, що дозволить уповільнювати атаки, коли обчислювальні потужності серверів будуть зростати.

Лістинг 3.19 – Механізм хешування паролю PBKDF2

```
class PBKDF2PasswordHasher(BasePasswordHasher):
    algorithm = "pbkdf2_sha256"
    iterations = 390000
    digest = hashlib.sha256
    def encode(self, password, salt, iterations=None):
        self._check_encode_args(password, salt)
        iterations = iterations or self.iterations
        hash = pbkdf2(password, salt, iterations, digest=self.digest)
        hash = base64.b64encode(hash).decode("ascii").strip()
        return “%s$%d$%s$%s" % (self.algorithm, iterations, salt, hash)
```

кінець лістингу 3.19

Важливим етапом в захисті інформаційної системи є захист від DDoS-атак. Django має декілька вбудованих методів, які допомагають захистити систему від DDoS-атак.

Найефективнішим методом захисту є захист на рівні серверу, який полягає в налаштуванні сервера на обмеження кількості запитів на інший сервер, можливість встановити ліміт запитів та синхронізувати час.

Ще одним методом захисту є використання CDN, тобто, розподілених серверів, які можуть зберігати копії статичного вмісту додатку, а саме: зображень, CSS і JavaScript, які розміщуються ближче до користувача, що допомагає скоротити час завантаження сторінок та зменшити навантаження на сервер.

Також, серед методів захисту є використання сторонніх бібліотек, таких як: Django Ratelimit або Django Defender, які допомагають обмежити кількість запитів на сервер, встановити ліміти на основні IP-адреси чи користувачів, також запобігають атакам у вигляді SQL-ін’єкцій та переповнення буферу. Ще однією особливістю цих бібліотек є автоматичне блокування IP-адрес, які порушують встановлені правила.

Додатковим методом захисту є використання Captcha, що дозволяє протистояти автоматичним запитам, оскільки генерується на стороні сервера, користувачу доведеться ввести правильні символи для успішного виконання запиту.

3.5 Огляд роботи інформаційної системи

Для розуміння інформаційної системи зі сторони візуалізації потрібно розглянути практичне застосування системи з оглядом її функціональності. Для початку потрібно перейти в додаток за допомогою веб-браузера та увійти в систему під логіном, який є адресою електронної пошти та паролем, зображено на рисунку 3.5.

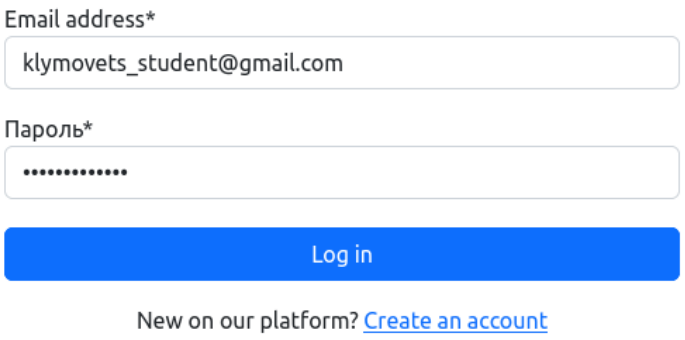

Рисунок 3.5 – Вигляд системи автентифікації у додатку

Після успішної автентифікації користувача направляє на головну сторінку, де можна прочитати коротку довідку про систему, зображено на рисунку 3.6.

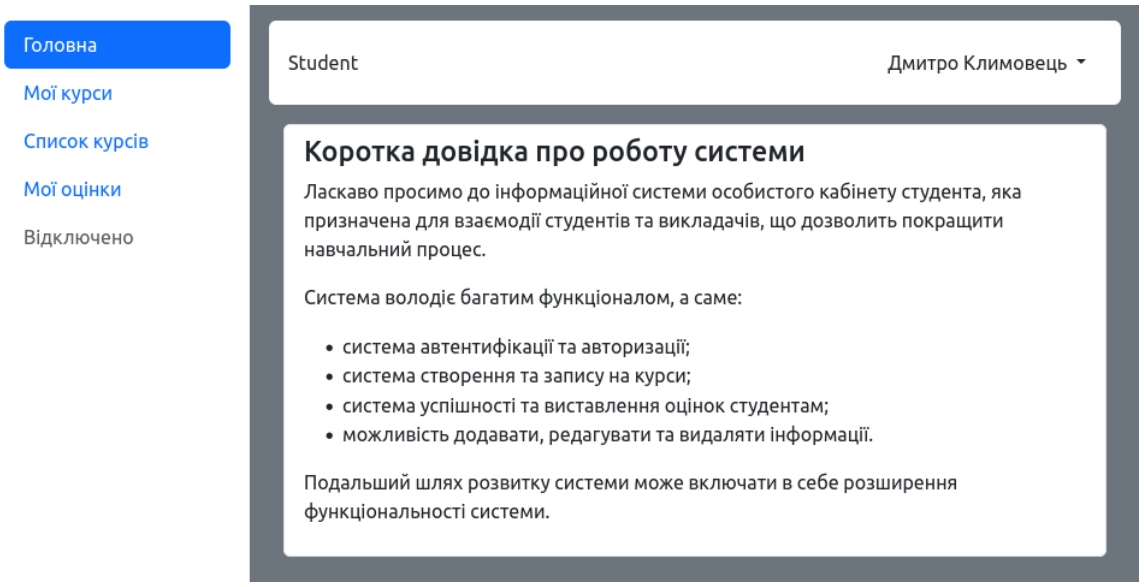

Рисунок 3.6 – Вигляд головної сторінки додатку

Наступною сторінкою буде пункт «Мої курси», де відображається список з усіма курсами на які записаний користувач з типом студент. Натиснувши на потрібний курс можна переглянути інформації про нього, а саме: назву, опис, викладач, що веде даний курс, та іншу інформацію, зображено на рисунку 3.7.

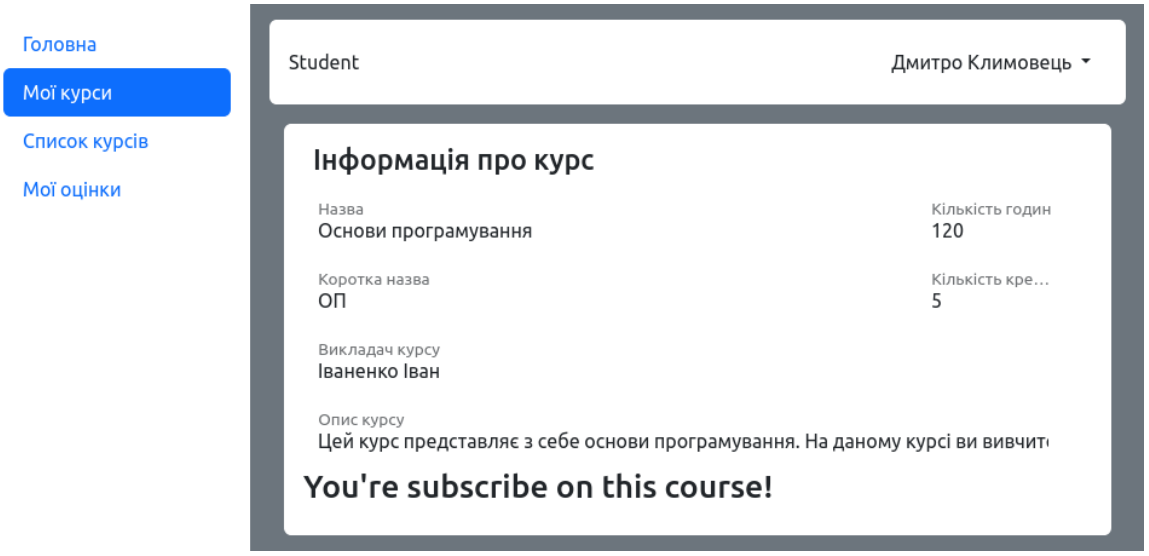

Рисунок 3.7 – Вигляд сторінки з курсами

Одним з найцікавіших сторінок додатку є пункт «Мої оцінки», де студент може переглянути власну успішність з усіх предметів, на які він записаний.

Сторінка складається з таблиці, в якій вказано предмет та оцінку оцінку студента, яку виставив викладач, або підпис «Оцінка відсутня», якщо оцінку ще не поставили, зображено на рисунку 3.8.

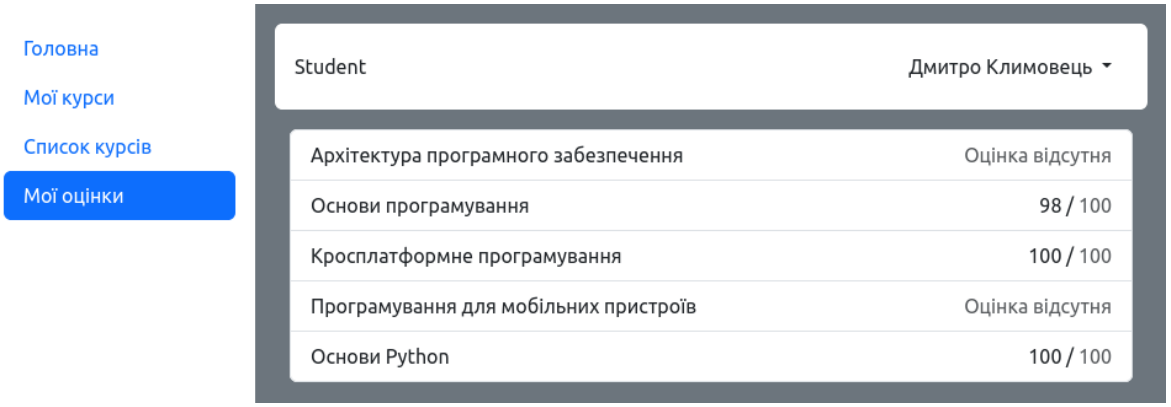

Рисунок 3.8 – Вигляд сторінки з успішність студента

Ще однією важливою сторінкою системи є пункт «Профіль», перейшовши на яку можна переглянути особисту інформацію про користувача з типом студент та інформацію про навчання, а саме: спеціальність, навчальна програма, освітній рівень, група, дата вступу. Також є можливість зміни всіх даних за допомогою відповідних форм, натиснувши кнопку «Редагувати». У нового користувача профіль відсутній, він сам повинен його заповнити, натиснувши кнопку «Створити», після чого його буде переведено на сторінку заповнення профілю. Вигляд сторінки профілю зображено на рисунку 3.9.

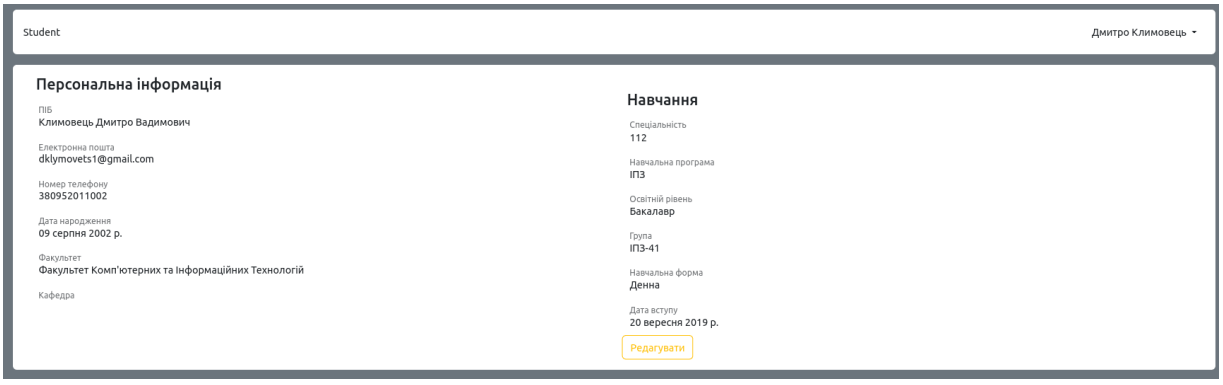

Рисунок 3.9 – Вигляд сторінки профілю студента

Автентифікувавшись як користувач з типом «Викладач» та перейшовши до сторінки «Мої курси» можна переглянути всі курси, які належать викладачу. На даній сторінці, у викладача є можливість створити новий курс, описавши його. Сторінка зі списком курсів, створених викладачем зображено на рисунку 3.10.

Рисунок 3.10 – Вигляд сторінки з курсами викладача

Перейшовши до одного з курсів можна переглянути інформацію про курс, редагувати її, або повністю видалити курс. Також на сторінці присутня таблиця зі студентами курсу, їхніми групами і оцінками, зображено на рисунку 3.11. Якщо оцінка відсутня, у полі з оцінкою викладач може оцінити студента за допомогою кнопки «Оцінити», зображено на рисунку 3.12.

Рисунок 3.11 – Вигляд сторінки з курсом та оцінками

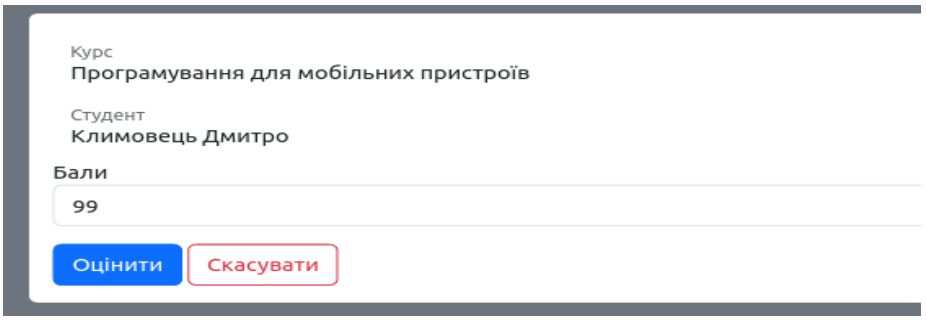

Рисунок 3.12 – Форма з оцінюванням студента

Також у викладача є можливість заповнити власний профіль, описавши вчительську інформацію, а саме: науковий ступінь, академічний рівень та кабінет, в якому перебуває його кафедра, зображено на рисунку 3.13.

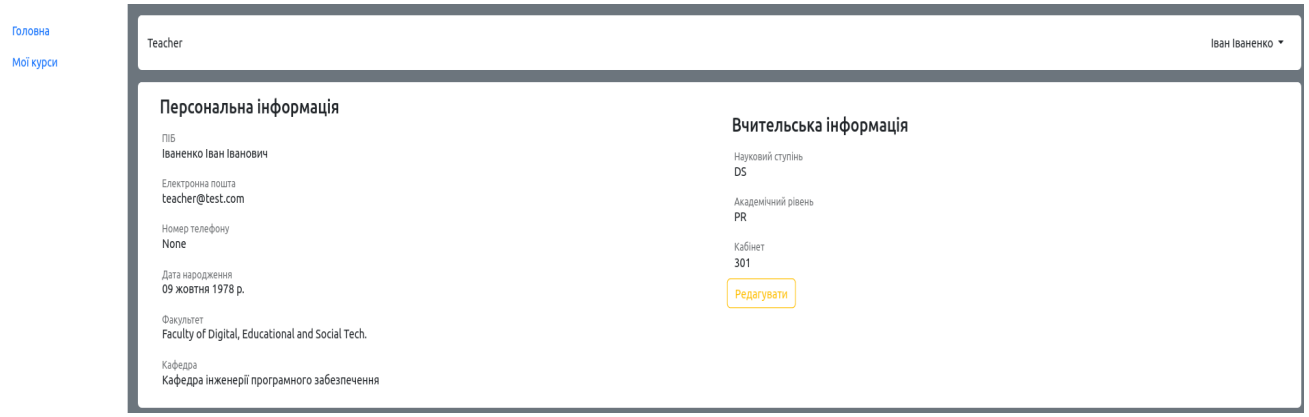

Рисунок 3.13 – Вигляд сторінки профілю викладача

Після закінчення роботи у системі користувач може успішно завершити сесію, вийшовши з системи за допомогою кнопки «Вийти».

Висновки до розділу 3

У даному розділі було проведено ряд практичних робіт, від реалізації додатку до інформаційного захисту системи. Було описано практичну реалізацію об’єкта проектування, а саме: етапи створення додатку, вигляд представлень з

використанням різних алгоритмів реалізації бізнес-логіки та описано типи користувачів з їхніми можливостями та правами.

Важливим етапом розділу було проведення тестування та налагодження системи, де було описано мінімальні системні вимоги для повноцінного функціонування додатку, також, описано тестування моделей додатку за допомогою різноманітних запитів до бази даних.

Також, було спроектовано схему бази даних, після чого, було реалізовано на практиці. В цьому пункті, також, описано роботу запитів на створення та додавання даних до бази даних.

На кінцевому етапі даного розділу було розглянуто захист інформаційно-комп’ютерної системи, де було описано різноманітні методи та механізми захисту даних, авторизації та автентифікації, і хешування паролів, які було розглянуто в функціональних схемах.

ВИСНОВКИ

Під час виконання кваліфікаційної роботи було розроблено інформаційну систему особистого кабінету студента, яка надає достатній спектр функціональності, пов’язаної з навчальним процесом.

Було розглянуто аналоги розроблюваної інформаційної системи, такі як: Google Classroom, Moodle, електронний кабінет ЛНТУ та Source LMS. Також було описано їхні можливості, переваги та недоліки, щоб охопити різні аспекти.

Відповідно до поставленого завдання, було складено вимоги до програмного забезпечення, які включають: функціональні вимоги, нефункціональні вимоги, вимоги до продуктивності системи, вимоги до надійності та безпеки, і вимоги до дизайну інтерфейсу, що дозволило створити повноцінну інформаційну систему для роботи з навчальним процесом.

Для швидкого та зручного отримання необхідної інформації про навчання було створено зручний та лаконічний прототип інтерфейсу користувача за допомогою векторного редактора Figma. Завдяки великому функціоналу вдалося створити сторінки з усієї системи, відповідно до принципів веб-дизайну.

Провівши аналіз обраних засобів та технологій на їхні переваги і недоліки, було обрано фреймворк Django, що має великий вбудований інструментарій, який дозволив швидко і ефективно створити інформаційну систему з достатньою функціональністю та відповідним рівнем безпеки.

Також було проведено проектування інформаційної системи, а саме: створено блок-схему доступу до курсів, щоб візуалізувати послідовність етапів розробки алгоритму. Також було змодельовано діаграму послідовності для створення курсу та діаграму прецедентів для викладача і студента.

Було налаштовано середовище розробки та встановлено відповідні технології.

В ході розробки було створено схему бази даних, сформовано її структуру та розміщено відповідні відносини між таблицями і полями.

Також було написано код для усіх функціональних систем та механізмів, а саме: створено систему аутентифікації і авторизації, реалізовано функціональні блоки, відповідно до завдання та розроблено базу даних з усіма таблицями і полями, відповідно до схеми.

Отриманий результат можна використовувати в закладах вищої освіти, особливо в управлінні навчальним процесом, успішності та роботою з даними студентів.

Подальший шлях розвитку системи може включати в себе розширення функціональності системи, також, підвищення рівня безпеки та покращення продуктивності, що дозволить скоротити час на опрацювання запитів.

Отже, всі завдання поставлені на виконання даної роботи було успішно виконано, також, було розроблено інформаційну систему відповідно до поставлених вимог, який повністю готовий до практичного застосування.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Дистанційне навчання з використанням Google Classroom, інструментів Microsoft і LMS-систем: Source LMS, Moodle тощо. URL: https://evergreens.com.ua/ua/articles/lms-comparison.html. (дата звернення: 06.02.2023).

2. Організація дистанційного навчання в Moodle. URL: https://osvita.ua/vnz/high_school/72285/. (дата звернення: 09.02.2023).

3. Source LMS і Moodle: порівняння можливостей. URL: https://evergreens.com.ua/ua/articles/source-lms-vs-moodle.html. (дата звернення: 13.02.2023).

4. UML для бізнес-моделювання: для чого потрібні діаграми процесів. URL: https://evergreens.com.ua/ua/articles/uml-diagrams.html. (дата звернення: 28.02.2023).

5. Що таке функціональні вимоги: приклади, визначення, повний посібник. URL: https://visuresolutions.com/uk/blog/functional-requirements/. (дата звернення: 19.02.2023).

6. Мельничук Ю. Є. Архітектура та проектування програмного забезпечення / Ю. Є. Мельничук. – Луцьк: Луцький НТУ, 2019. – 98 с. (дата звернення: 17.04.2023).

7. Особливості web-design. URL: https://outsourcing.team/uk/blog/design /osoblivosti-web-design/. (дата звернення: 04.03.2023).

8. Прототипування в Figma: Work smarter, not harder. URL: https://medium.com/design-pub/прототипирование-в-figma-work-smarter-not-harder-3ced8a0cf69. (дата звернення: 09.03.2023).

9. Самі популярні мови програмування бекенду. URL: https://habr.com/companies/skillbox/articles/534684/. (дата звернення: 18.03.2023).

10. Шершун О. О. Переваги та недоліки застосування Django для створення веб-додатків / О. О. Шершун. – Одеса: ОНАХТ, 2020. (дата звернення: 02.04.2023).

11. Django. URL: https://uk.wikipedia.org/wiki/Django. (дата звернення: 07.04.2023).

12. Варламов О. Розробка веб-додатків з використанням Django. URL: https://wezom.com.ua/ua/blog/razrabotka-veb-prilozhenij-s-ispolzovaniem-python-i-django. (дата звернення: 27.03.2023).

13. Django. Documentation. URL: https://docs.djangoproject.com/en/4.2/. (дата звернення: 12.04.2023).

14. Django ORM Cookbook. URL: https://django.fun/en/docs/django-orm-cookbook/2.0/. (дата звернення: 04.04.2023).

15. Тестування веб-проєктів: основні етапи та поради. URL: https://qalight.ua/baza-znaniy/testuvannya-veb-proektiv-osnovni-etapi-ta-poradi. (дата звернення: 25.04.2023).

16. Організація баз даних та знань. Проєктування баз даних / [О. В. Тарасов, В. В. Федько, М. Ю. Лосєв та ін.]. – Харків: Видавництво ХНЕУ, 2011. – 200 с. (дата звернення: 03.05.2023).

17. SQLite. URL: https://uk.wikipedia.org/wiki/SQLite. (дата звернення: 21.04.2023).

18. Безпека веб-додатків Django. URL: https://developer.mozilla.org/docs/Learn/Server-side/Django/web_application_security. (дата звернення: 12.05.2023).

19. Автентифікація. URL: https://uk.wikipedia.org/wiki/Автентифікація. (дата звернення: 16.05.2023).

20. Хеш-функція. URL: https://uk.wikipedia.org/wiki/Хеш-функція. (дата звернення: 18.05.2023).