

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Інститут інформатики та радіoeлектроніки,

Факультет комп'ютерних наук і технологій

(повне найменування інституту, назва факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

бакалавр

(ступінь вищої освіти)

на тему ПРОГРАМНА РЕАЛІЗАЦІЯ АВТОМАТИЗОВАНОЇ

СИСТЕМИ МОНІТОРИНГУ ДЛЯ ДЕКАНАТУ

SOFTWARE IMPLEMENTATION OF THE AUTOMATED

MONITORING SYSTEM FOR THE DEAN'S OFFICE

Виконав: студент(ка) 4 курсу, групи КНТ-218сп

Спеціальності 122 Комп'ютерні науки

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Комп'ютерні науки

Ржанніков К.В.

(прізвище та ініціали)

Керівник Гофман Є.О.

(прізвище та ініціали)

Рецензент Скрупський С.Ю.

(прізвище та ініціали)

(назва освітньої програми (спеціалізації))

“ ” 2021 року

Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	Гофман Є.О., доцент		
Нормоконтролер	Липовець М.В., асистент		

7. Дата видачі завдання “1” лютого 2021 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи	1 тиждень	Завдання, ТЗ
2	Аналіз та дослідження предметної області	2-5 тижні	Розділи 1, 2
3	Розробка архітектури програми	6 тиждень	Розділ 2
4	Розробка програми	7-9 тижні	Розділ 3
5	Тестування та експериментальне дослідження програми	10 тиждень	Розділ 4
6	Оформлення пояснювальної записки та документів до неї. Нормоконтроль та рецензування	11 тиждень	Додатки
7	Захист роботи	12 тиждень	

Студент(ка)

(підпис)

Ржанніков К.В.
(прізвище та ініціали)

Керівник проєкту (роботи)

(підпис)

Гофман Є.О.
(прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 87 с., 43 рис., 4 табл., 4 дод., 17 джерел.

ВИЩА ОСВІТА, МОНІТОРИНГ, ДОКУМЕНТООБІГ, ДІДЖИТАЛІЗАЦІЯ, АВТОМАТИЗАЦІЯ, РОЗРОБКА, ПРОЄКТУВАННЯ, СЕРВЛЕТ, ВЕБ-РОЗРОБКА.

Об'єкт дослідження – процес автоматизації моніторингу успішності здобувачів освіти.

Предмет дослідження – методи автоматизації моніторингу успішності здобувачів освіти.

Метою роботи є програмна реалізація автоматизованої системи моніторингу для деканату.

Матеріали, методи та технічні засоби: об'єктно-орієнтоване програмування, мова програмування Java, технологія Java сервлетів, система керування базою даних MySQL.

Результати. Розроблено автоматизовану систему моніторингу успішності здобувачів вищої освіти для деканату.

Практична цінність роботи полягає в автоматизації процесів моніторингу успішності здобувачів вищої освіти працівниками деканату та розробці такої системи у вигляді кросплатформного веб-застосунку.

Висновки. Розроблено програмне забезпечення, що реалізує автоматизований процес моніторингу успішності здобувачів вищої освіти працівниками деканату.

Галузь використання – працівники деканатів та інших структурних підрозділів закладів вищої освіти.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 87 pages, 43 figures, 4 tables, 4 appendixes, 17 sources.

HIGHER EDUCATION, MONITORING, DOCUMENT MANAGEMENT, DIGITALIZATION, AUTOMATION, DEVELOPMENT, DESIGN, SERVLET, WEB DEVELOPMENT.

The object of research is the process of automating monitoring the success of educational applicants.

The subject of the research is methods of automating monitoring the success of educational applicants.

The aim of the work is to programmatically implement an automated monitoring system for the dean's office.

Materials, methods, and technical tools: Object-Oriented Programming, Java programming language, Java Servlet technology, MySQL database management system.

Results. An automated system for monitoring the success of higher education applicants for the dean's office has been developed.

The practical value of the work lies in automating the processes of monitoring the success of higher education applicants by employees of the dean's office and developing such a system in the form of a cross-platform web application.

Conclusions. Software has been developed that implements an automated process for monitoring the success of higher education applicants by employees of the dean's office.

Field of usage: employees of Deans ' offices and other structural divisions of higher educational institutions.

ЗМІСТ

	С.
Перелік скорочень та умовних познач.....	8
Вступ.....	9
1 Аналіз предметної області	10
1.1 Системи електронного документообігу	10
1.1.1 Ключові можливості систем електронного документообігу	10
1.2 Дослідження існуючих аналогів	12
1.2.1 Пакет програм від «Політек-Софт».....	12
1.2.2 Система «FossDoc»	13
1.2.3 Система управління навчанням Moodle	15
1.2.4 Порівняння існуючих аналогів	16
1.3 Постановка завдання	18
1.3.1 Функціональність системи	18
1.3.2 Вимоги до інтерфейсу.....	19
1.3.3 Не функціональні вимоги до продукту.....	20
1.3.4 Оцінка надійності, продуктивності і підтримки	20
1.4 Висновки за розділом 1	21
2 Розробка архітектури програми	22
2.1 Вибір мови програмування.....	22
2.1.1 Вибір середовища розробки.....	25
2.2 Загальна структура додатку.....	29
2.3 Висновки за розділом 2	30
3 Основні рішення щодо реалізації компонентів системи	32
3.1 Проектування системи на основі предметної області	32
3.2 Розробка архітектури застосунку.....	40
3.3 Висновки за розділом 3	42
4 Експлуатація, тестування та експериментальне дослідження програми.....	43
4.1 Опис застосування	43
4.2 Умови виконання програми.....	43

4.3 Інструкція по експлуатації програми	44
4.3.1 Звернення до програми.....	44
4.3.2 Вхідні й вихідні дані	44
4.4 Методика та результати тестування	44
Висновки	53
Перелік джерел посилання	55
Додаток А Технічне завдання	57
Додаток Б Опис програми	64
Додаток В Текст програми	69
Додаток Д Слайди презентації.....	80

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

GUI – графічний інтерфейс користувача;

MVC – Model-view-controller;

UML – Unified Modeling Language;

БД – база даних;

ЗВО – заклад вищої освіти;

ОС – операційна система

ПЗ – програмне забезпечення;

ПК – персональний комп’ютер;

СКБД – система керування базою даних;

ТЗ – технічне завдання.

ВСТУП

В даний час комп'ютерна техніка широко застосовується у всіх областях людської діяльності. Існує багато інформаційних систем, які автоматизують роботу у різних підприємствах. В закладах вищої освіти (ЗВО) також потрібна автоматизація робочих місць, а саме деканатів університету. Адже найбільш тяжкою за трудомісткістю є робота з документацією та великим обсягом інформації. Для автоматизації їх роботи розробляються програмне забезпечення (ПЗ) та веб-застосунки, які містять у собі інформацію та усі можливі функції для роботи з нею.

Веб-застосунок – клієнт-серверний застосунок, в якому клієнтом виступає браузер або інша клієнтська програма, а сервером – веб-сервер. Логіка веб-застосунків розподілена між сервером і клієнтом, зберігання даних здійснюється, переважно, на сервері, обмін інформацією відбувається по мережі. Розробка та впровадження веб-застосунку надає можливість керувати різного виду даними.

Тож метою дипломної кваліфікаційної роботи бакалавра є допомога ЗВО у підвищенні їх ефективності шляхом забезпечення їх сучасним інструментом управління освітнім процесом. Актуальністю програми вважається необхідність в розробці системи автоматизованого формування документів для деканату, на підставі даних, які вводяться різними користувачами, їх редагуванні та видаленні. Розроблювана система дозволить спростити роботу з документацією деканату, зокрема із заповненням бланків атестацій і дозволить автоматизувати створення звітів про роботу кураторів навчальних груп.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Системи електронного документообігу

Електронний документообіг – це життєвий цикл електронних документів в організації, починаючи від їх отримання (введення, електронна пошта і т.п.), проходження в підрозділах зі зміною стану (доведений до відома, узгоджений, підписаний, в роботі, закритий і т. п.) і закінчуючи списанням в архів. Часто електронний документообіг позначається терміном *workflow*, який характеризує рух документів як потік робіт, що виконуються в рамках того чи іншого бізнес-процесу. Система електронного документообігу – це програмне забезпечення, головними завданнями якого є організація та підтримка життєвого циклу електронних документів [1]-[2].

Система електронного документообігу – це рішення на той чи іншій платформі, призначене для створення електронного архіву документів, організації корпоративного документообігу і автоматизації бізнес-процесів на підприємствах, в установах і організаціях будь-якого роду діяльності. Програма дозволяє вирішити велику кількість завдань, реалізація яких покладена на відповідні модулі. Система може бути легко переналаштована з урахуванням специфіки роботи кожного конкретного підприємства [2].

1.1.1 Ключові можливості систем електронного документообігу

Представимо ключові можливості систем електронного документообігу у вигляд певного розподілу за сферами. Так слід розпочати із персоніфікації графічного інтерфейсу користувача (GUI) і функціоналу [3]:

- системи дозволяють систематизувати документи за розділами, папками, каталогами, підпапками і т.д.;
- переміщувати папки і підпапки, щоб організувати порядок на користувацькому робочому столі як більше подобається;

- системи надають розумні повідомлення та підказки, які не дадуть помилитися або заплутатися;
- можливість відключати невикористовуваний функціонал (наприклад, можна відключити папку "на візування" і відповідно її функціонал);
- брендиувати застосунки та вихідні документи.

Системи електронного документообігу надають віртуальне комфортне робоче місце [4]:

- підтримка електронного підпису;
- глобальна синхронізація: так почати роботу можна на одному пристрої, а продовжити на іншому;
- підтримка offline-режиму, коли можна працювати з документами і завданнями навіть без Інтернету (зміни за документами будуть передані в систему при відновленні доступу до мережі);
- сповіщення про нові документи (push-повідомлення).

Також подібні системи надають розширені можливості з формування доручень та звітів [3]-[5]:

- створення багатопунктових доручень – можливо створити і відправити разом кілька доручень;
- перегляд доручень і звітів завдяки дереву доручень – в ієрархічній структурі;
- створення ініціативних доручень;
- створення та редагування звітів.

Також результати роботи в системах електронного документообігу можуть бути узгоджені та підписані [3]-[5]:

- перегляд дерева узгодження;
- візування та підписання проєкту документа;
- формування зауважень: голосових, текстових і графічних.

Тож як бачимо розробка систем, які б автоматизували внутрішні процеси державних установ, в тому числі ЗВО має ряд переваг, що значно покращують та спрощують роботу персоналу [3], [5].

1.2 Дослідження існуючих аналогів

На сьогоднішній день на просторах Інтернету існує достатньо багато систем, що схожі за функціоналом та загалом надають ПЗ для автоматизації та діджиталізації внутрішніх процесів закладів освіти. Проте слід зазначити, що більшість з них досить різняться за основним процесом та фокусом систем [3], [6].

Так певна кількість зосереджена на процесах автоматизації та діджиталізації саме освітнього процесу. Це відноситься до систем, на кшталт: Moodle або GSuit [3], [6].

Інші ж системи направлені саме на цифровий документообіг структурних підрозділів ЗВО та мало пов'язані саме із процесами навчання та викладання [3], [6].

Під час аналізу існуючих аналогів зосередимося саме на представниках другого класу.

1.2.1 Пакет програм від «Політек-Софт»

Компанія «Політек-Софт» пропонує цілий програмний комплекс «Деканат». Загалом, ПЗ розраховане на використання в навчальній частині ЗВО. А серед вирішуваних задач: створення, редагування та підрахунки навчальних планів (за кількостями годин, кредитів, тощо), отримання звітних документів за результатами контрольних заходів та формування множини навантажень для професорсько-викладацького складу [7].

Сама програма постачається у вигляді веб-застосунку, що розгортається на сервері. Що надає ряд переваг [7]:

- не має необхідності встановлення додаткового ПЗ на інших комп'ютерах мережі ЗВО; вищого навчального закладу додатково нічого встановлювати не потрібно;
- відсутня необхідність чіткої прив'язки працівника до конкретного робочого місця – працівник може працювати з ПЗ з будь-якого комп'ютера мережі ЗВО;
- ПЗ надає можливість віддаленої роботи без складностей із віддаленості навчальних корпусів;
- посилання на ПЗ можуть бути розміщені на офіційних медіа-ресурсах ЗВО, що спрощує доступ та гарантує відкритість освітнього та навчального процесу, що важливо в рамках 9-ого критерію Національного агентства.

Тож загалом це суттєво спрощує впровадження та підтримку системи [7].

На рисунку 1.1 наведено приклад вікна ПЗ від «Політек-Софт».

Держ. код	Осв. ступінь (СНП)	Форма навч.	СНП	Перепідготовка?	Деталізована назва навч. плану	Точна назва напрямку (спеціальності)
014	Бакалавр	Денна	Так		6.014 Середня освіта (фізика). Спеціальність: інформатика (Д.2016/2017 н.р.)	Середня освіта (фізика). Спеціальність: інформатика
014	Бакалавр	Денна	Так		6.014 Середня освіта (фізика). Спеціальність: інформатика (Д.2017/2018 н.р.)	Середня освіта (фізика). Спеціальність: інформатика
014	Бакалавр	Денна	Так		6.014 Середня освіта (математика). Спеціальність: економіка (Д.2016/2017 н.р.)	Середня освіта (математика). Спеціальність: економіка
014	Бакалавр	Денна	Так		6.014 Середня освіта (математика). Спеціальність: економіка (Д.2017/2018 н.р.)	Середня освіта (математика). Спеціальність: економіка

Рисунок 1.1 – Приклад вікна ПЗ від «Політек-Софт»

1.2.2 Система «FossDoc»

Слід зазначити, що «FossDoc» – це система для загальної діджиталізації та автоматизації документообігу організацій. Система

електронного документообігу FossDoc дозволяє вирішувати всі завдання сучасного діловодства. Базовими поняттями FossDoc є картка документа, доручення і папки документів. За допомогою картки можна переглядати і редагувати будь-який документ: вводити значення полів, розміщувати посилання на інші документи і додавати іншу інформацію, наприклад, вкладені файли з відсканованим паперовим оригіналом документа, приєднувати листування по електронній пошті [8].

Часто у великих організаціях вирішенням завдань контролю виконання документів займається спеціальний підрозділ – відділ контролю. В системі функції співробітників відділу контролю автоматизовані: існує відповідна роль користувача, а в картці документа є спеціальне поле «Контроль» – де відповідальні співробітники ставлять контрольні позначки [8].

На рисунку 1.2 наведено приклад вікна системи FossDoc.

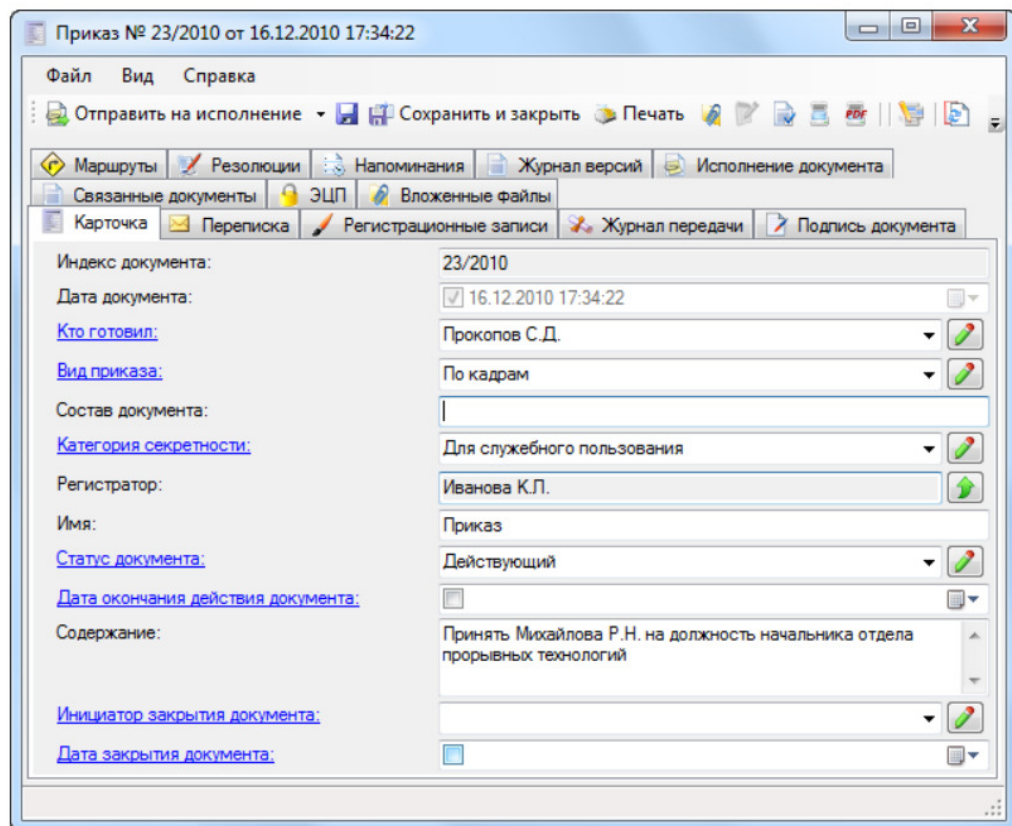


Рисунок 1.2 – Приклад вікна системи FossDoc

У FossDoc автоматизовано створення різних типів звітів: виконання документів, виконання контрольних доручень, звіти по вхідній кореспонденції, за зверненнями громадян і т.п. [8].

1.2.3 Система управління навчанням Moodle

Moodle – це комплексна система навчанням або віртуальне навчальне середовище. Являє собою вільний веб-застосунок, що надає можливість створювати сайти для онлайн-навчання [9].

Платформа надає простір для спільної роботи вчителів і студентів. У Moodle доступні різні можливості для відстеження успішності учнів, а також є підтримка масової реєстрації з безпечною аутентифікацією [9].

На рисунку 1.3 наведено приклад сторінки системи Moodle.

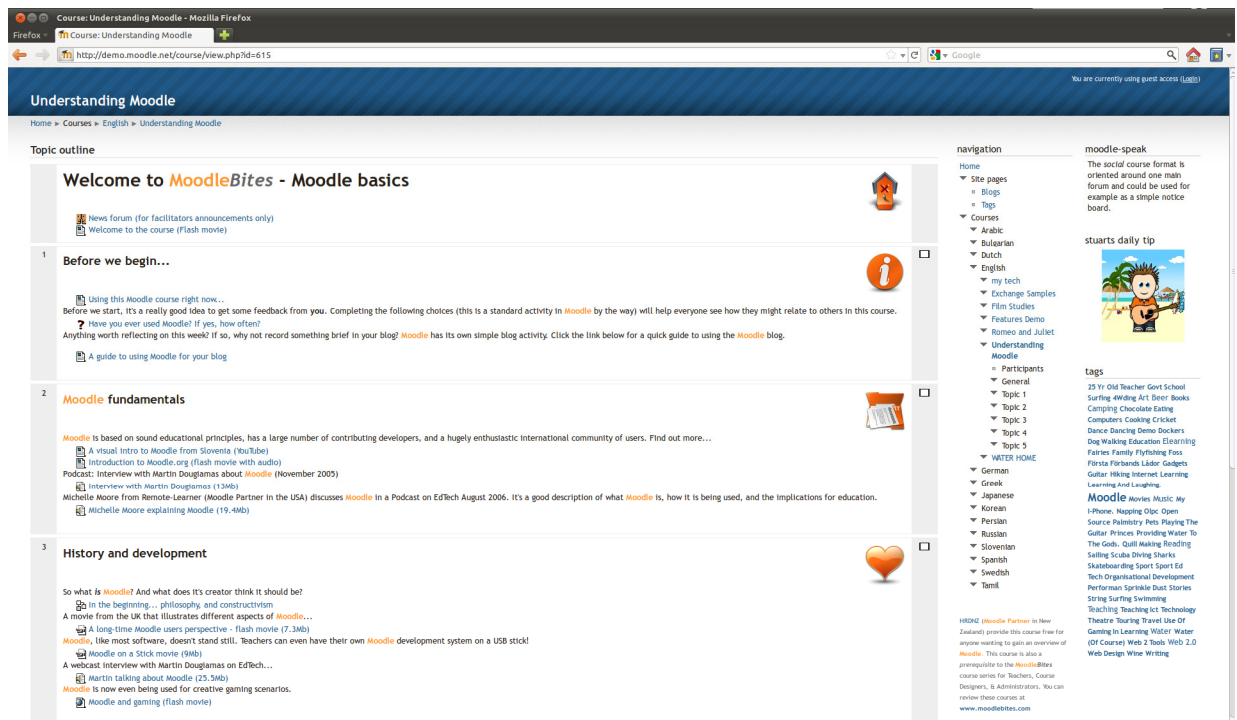


Рисунок 1.3 – Приклад сторінки системи Moodle

Система має гнучкий інтерфейс з можливістю конфігурування макетів і дизайну окремих сторінок. Платформу можна інтегрувати з великою кількістю програмного забезпечення, включаючи інструменти для

спілкування, спільної роботи, управління документами та інші програми для підвищення продуктивності. Moodle має відкритий вихідний код [9].

Також важливою складовою є наявність звітної інформації за результатами контрольних заходів, що може бути організовано передана навчальним відділам [9].

1.2.4 Порівняння існуючих аналогів

Проведемо порівняння за певними критеріями, хід якого представимо у вигляді таблиці (табл. 1.1). У таблиці будуть порівнюватися попередні аналоги із соціальною мережею, що розробляється – Focus. Серед критеріїв виділимо наступні:

- спеціалізованність під специфіку ЗВО – чи є у системи можливості бути адаптованою саме під роботу із ЗВО та особливі потреби навчальних підрозділів;
- форма застосунку – у якому вигляді функціонує за стосунок (веб, десктоп);
- складність підтримки – чи вимагає підтримка втручання виключно в серверну частину або потрібне налаштування і серверної, і клієнтської частини;
- рівень підтримки процесів документообігу – чи надає система можливість автоматично генерувати та моніторити звітну документацію.

З результатів порівняння можна зробити наступні висновки. По-перше, більшість аналогів або слабо пов'язані із специфікою ЗВО або із специфікою саме навчальних підрозділів ЗВО. Також, не усі аналоги підтримують саме документообіг. По-друге, аналоги в формі веб-застосунків більш легкі в підтримці саме завдяки тому, що має підтримуватися тільки серверна частина. Також такі застосунки надають можливість віддаленої роботи з будь-якого робочого місця.

Таблиця 1.1 – Порівняння існуючих аналогів

Критерій порівняння	Політек-Софт	FossDoc	Moodle	Система Деканат
Спеціалізованість під специфіку ЗВО	Високий рівень	Низький рівень	Високий рівень	Високий рівень
Форма застосунок	Веб-застосунок	Десктопний застосунок	Веб-застосунок	Високий рівень
Складність підтримки	Низький рівень	Високий рівень	Низький рівень	Низький рівень
Рівень підтримки процесів документообігу	Високий рівень	Високий рівень	Низький рівень	Високий рівень

Тож актуальною залишається задача розробки та підтримки застосунку автоматизації моніторингу для навчальних відділів, в тому числі деканатів. Для додаткової аргументації актуальності розглянемо таблицю 1.2.

Таблиця 1.2 – Особливості розробки та використання систем автоматизованого моніторингу в ЗВО

Застосунок	Недоліки	Переваги
Використання незалежної розробки	– надмірна комерціалізація; – складність налаштування під особливості ЗВО; – низька ергономічність дизайну	– якісна документація; – сервісна підтримка; – широкий набір функціональних можливостей
Власна розробка ЗВО	– нижчий рівень стійкості; – залежність від супроводу розробників; – неякісна документованість	– більша відповідність особливостям конкретного ЗВО; – більш гнучке та оперативне оновлення на основі власних нормативних документів; – зручність для використання

Як видно більш вигідніше вкластися в розробку власного застосунку для конкретного ЗВО, адже така система надасть повноцінний інтерфейс для взаємодії працівників. Така система буде більш зв'язана із внутрішньою нормативною базою ЗВО та буде адаптована під особливості конкретного ЗВО.

1.3 Постановка завдання

За результатами аналізу проблемної області та дослідження існуючих аналогів для вирішення проблемних питань, сформулюємо постановку задачі.

Метою є розробка нової автоматизованої системи моніторингу для деканату. Серед функціональних вимог визначимо:

- реєстрація користувачів;
- вивід інформації про категорії даних;
- перегляд інформації;
- пошук інформації;
- додавання інформації;
- видалення інформації;
- оформлення звіту з обраної інформації;
- вивід створених або збережених звітів на друк.

Система буде розроблена у вигляді веб-застосунку, що гарантує кросплатформність.

Розглянемо вимоги до розроблюваної системи за їх типом [10]-[11].

1.3.1 Функціональність системи

Функціональність системи має забезпечуватися за допомогою:

- наявність та редагування списку студентів, що вчаться на факультеті;

- наявність та редагування інформації про форму навчання;
- можливість розрахунку оцінок студентів за навчальний період;
- можливість розрахунку середнього балу студентів згідно з оцінками;
- наявність та оновлення списку студентів, що отримують стипендію;
- наявність та редагування списку студентів, що отримують освіту екстерном;
- наявність та оновлювання списку іноземних студентів;
- наявність та редагування списку груп факультету та загальна їх кількість;
- оновлення списку студентів, що навчаються на пільговій основі;
- редагування розклад занять для студентів;
- редагування розклад занять для викладачів факультету;
- можливість видачі довідок;
- система повинна здійснювати збір інформації про студентів та викладачів;
- можливість виконувати пошук потрібної інформації у системі.

1.3.2 Вимоги до інтерфейсу

Для забезпечення ергономічності, зручності та зрозумілості інтерфейс розроблюваного веб-додатку повинен:

- інтерфейс повинен бути простим та зрозумілим у використанні;
- у забарвленні системи повинні бути кольори неясних, пастельних відтінків;
- пароль відображається зірочками.

1.3.3 Не функціональні вимоги до продукту

Мінімальні системні вимоги та програмне забезпечення , що необхідні для функціонування серверної частини мобільного застосунку:

- операційна система (ОС) – будь-яка з різновидів операційної системи Unix (включаючи Mac OS X) або Windows;
- веб-сервер – HTTP-сервер Apache для Unix і Windows, сервер Microsoft Internet Information або сервер Netscape Enterprise;
- інтерпретатор PHP5;
- MySQL-сервер із правами створення та вилучення користувачів бази даних (БД) (або база SQLite з аналогічними правами).

Для експлуатації клієнтської частини необхідні такі програмно-технічні засоби:

- персональний комп'ютер (ПК);
- процесор із тактовою частотою від 1,5 GHz;
- обсяг оперативної пам'яті – не менше 3GB;
- підключення до мережі Internet;
- встановлений веб-оглядач останньої версії.

1.3.4 Оцінка надійності, продуктивності і підтримки

Для забезпечення достатньої надійності та продуктивності система має відповідати певним вимогам:

- система повинна працювати постійно без збою;
- займатися підтримкою системи повинен адміністратор та модератор;
- кожен користувач системи повинен мати ідентифікований логін та пароль;
- бажано щоб програма працювала без додаткового налаштування;

- дана система безкоштовна в межах компанії.

1.4 Висновки за розділом 1

Виконано огляд проблеми розробки та підтримки автоматизованої системи моніторингу для деканату. Виділено основні шляхи для напрямки для розроблення нового ПЗ у вигляді веб-застосунку.

Розробка та підтримка автоматизованої системи «Система Деканат» має надати ПЗ, що забезпечує можливості:

- реєстрація користувачів;
- вивід інформації про категорії даних;
- перегляд інформації;
- пошук інформації;
- оновлення інформації;
- оформлення звіту з обраної інформації;
- вивід створених або збережених звітів на друк.

Система повинна бути кросплатформною. Система також повинна бути простою й зрозумілою користувачу.

2 РОЗРОБКА АРХІТЕКТУРИ ПРОГРАМИ

У попередньому розділі було проведено аналіз та дослідження існуючих аналогів автоматизованих систем моніторингу для ЗВО. Виходячи з проведеного аналізу, було сформульовано основні можливості, якими повинна оперувати система, що проєктується.

Що стосується технічних та програмних можливостей, то система повинна забезпечувати можливість моніторингу та ознайомлення із даними про здобувачів вищої освіти.

Даний розділ містить опис питань, які стосуються безпосередньо структури системи, що розробляється, вибору середовища проєктування, а також визначаються основні та специфічні архітектурні рішення, які має система.

2.1 Вибір мови програмування

Java – строго типізована об'єктно-орієнтована мова програмування загального призначення, розроблена компанією Sun Microsystems (в подальшому придбаною компанією Oracle). Розробка ведеться спільнотою, організованим через Java Community Process; мова і основні реалізують його технології поширюються за ліцензією GPL [12]-[13].

Програми Java зазвичай транслюються в спеціальний байт-код, тому вони можуть працювати на будь-якій комп'ютерній архітектурі, для якої існує реалізація віртуальної Java-машини. Дата офіційного випуску – 23 травня 1995 року. Стабільно займає високі місця в рейтингах популярності мов програмування [12]-[13].

Однією з найприємніших особливостей Java є її багатогранна природа. Звичайно, створення традиційних десктопних і навіть мобільних застосунків – це основа [12]-[13]. Але можливо зайти і на територію розробки веб-застосунків на Java. В комплекті з мовою йде повноцінний Servlet API, який

дозволяє вам створювати надійні веб-застосунків без особливого клопоту [12]-[13].

Сервлети, особливий тип Java-програм, що виконується в межах веб-контейнера (також званий контейнером сервлетів, як, наприклад, Tomcat і Jelly), які дозволяють обробляти запити клієнтів і відповіді сервера просто і ефективно. Сервлети створюються і знищуються їх контейнерами, а не розробником, і діють як проміжний рівень між клієнтами (як правило, веб-браузерами) та іншими застосунками, запущеними на сервері (наприклад, базами даних) [12]-[13].

Сервлет крім усього іншого може приймати дані від клієнта, як правило через GET і POST-запити, працювати з cookie і параметрами сеансу. А ще вона обробляє дані через додаткові рівні застосунків і відправляє вихідні дані клієнту як в текстовому, так і в бінарному форматах (HTML, XML, PDF, JPG, GIF і т.д.), у багатьох випадках використовуючи Java Server Pages файли [12]-[13].

Слід також відзначити основні особливості використання технології сервлетів, що робить їх популярними під час веб-розробки [12]-[13].

Ефективність. Не треба породжувати нового процесу. У стандартній ситуації все вже в голові і відразу починає виконуватися. Зрозуміло, що тут сервлети порівнюють з CGI_скриптами. В принципі, навіть інтегрований в браузер Perl, модуль mod_perl в Apache, наприклад, не повинен теоретично забезпечувати такої продуктивності, яку обіцяють сер влети [12]-[13].

Єдність середовища розробки програмного забезпечення. Вивчив Java і пиши собі код на здоров'я. Не треба вчити інші мови програмування, наприклад, Perl, VBScript або C. Правда, сама ця мова, м'яко кажучи, не дуже проста (мається на увазі Java) [12]-[13].

Потужність. Java – мова багата. Написати на ній можна багато чого. Правда, Perl в цьому сенсі не гірше. У всякому разі, там, де не треба розписувати віконечка і кнопочки екранних інтерфейсів. А програмування на стороні сервера – це якраз такий випадок. Perl і Java-це основні мови

розробки веб-застосунків. З точки зору складності вони приблизно однакові. Правда, кожна по своєму [12]-[13].

Інтерпретованість. Інтерпретовані програми за великим рахунком всі переносимі. Був би на відповідній платформі потрібний інтерпретатор. Java краще тільки в тому плані, що для нього централізовано підтримується єдина специфікація мови [12]-[13].

Безкоштовність. 80% веб-застосунків працює на безкоштовному Apache, це норма [12]-[13].

Для аргументації вибору Java з використанням технології сервлетів проведемо порівняння, результати порівняння представимо в табличному вигляді – таблиця 2.1. У якості критеріїв виділемо:

- швидкість роботи застосунку – швидкість роботи розробленого застосунку;
- спільнота та підтримка – кількість матеріалів для вивчення технології та періодичність оновлення технології;
- захист даних – вбудовані механізми захисту даних;
- багато поточність – вбудовані механізми, що забезпечують організацію потоків для обробки множини запитів;
- інтерпретованість застосунку – переносимість застосунку між платформами.

Таблиця 2.1 – Порівняння використання фреймворку та CMS під завдання замовника

	Java (сервлети)	C# (.Net)	Python	PHP
Швидкість роботи застосунку	++	+	±	+
Спільнота та підтримка	+	+	+	++
Захист даних	++	++	+	+
Багатопоточність	++	±	±	±
Інтерпретованість застосунку	+	±	+	+

Виходячи з порівняння розробка з використанням Java та технології сервлетів буде більш актуальною та надасть більше переваг вже розробленому застосунку для подальшого використання та можливих модифікацій.

2.1.1 Вибір середовища розробки

Програмне забезпечення JetBrains IntelliJ IDEA – це провідне середовище швидкої розробки на мові Java. IntelliJ IDEA являє собою високотехнологічний комплекс тісно інтегрованих інструментів програмування, що включає інтелектуальний редактор вихідних текстів з розвинутими засобами автоматизації, потужні інструменти рефакторінгу коду, вбудовану підтримку технологій J2EE, механізми інтеграції із середовищем тестування Ant/JUnit і системами керування версіями, унікальний інструмент оптимізації та перевірки коду Code Inspection, а також інноваційний візуальний конструктор графічних інтерфейсів [14].

Унікальні можливості JetBrains IntelliJ IDEA позбавляють програміста від вантажу рутинної роботи, допомагають своєчасно усунути помилки і підвищити якість коду, піднімаючи продуктивність розробника на нову висоту [14].

Нещодавно було представлено оновлену версію IntelliJ IDEA, що має цілий ряд змін. Нова платформа для розробки та командної взаємодії Space тепер інтегрована з IntelliJ IDEA. Після того як користувач увійшов в свою організацію в JetBrains Space, він зможе переглядати і клонувати репозиторії проєктів, рев'ювати код своїх колег і писати скрипти автоматизації Space.

З'явилася підтримка WSL 2 для Java-проєктів [14].

Сервіс для спільної розробки і парного програмування Code With Me тепер доступний «з коробки», а крім того, в ньому з'явилася підтримка відео- та аудіо-дзвінків [14].

Нова функція Run Targets дозволяє запускати, тестувати, профілювати і налагоджувати додатки в Docker-контейнерах і на віддалених комп'ютерах [14].

В IDE доданий вбудований браузер для прев'ю HTML-файлів. Прев'ю оновлюється в реальному часі при редагуванні HTML-коду або пов'язаних файлів CSS і JavaScript. Зовнішні залежності можна включати в область пошуку і виключати з неї [14].

Було розставлено лейбли на елементах інтерфейсу і полях редактора, і тепер вони коректно обробляються програмами читання з екрану. IDE автоматично включить висококонтрастну тему при першому запуску, якщо в Windows обраний режим високої контрастності. Працюючи в Windows, тепер можна відкривати недавні проєкти, клікнувши правою кнопкою миші по значку IntelliJ IDEA на панелі завдань або в меню Пуск [14].

У цьому релізі представлені оновлення для роботи з пул-реквестами: прискорився процес їх створення, з'явилася можливість швидко відкривати diff в редакторі і додалася підтримка шаблонів пул-реквестів. З'явилася підтримка шаблонів Git-коммитов, і в них можна вносити будь-яку інформацію, яка повинна бути в повідомленні комміта. Тепер можна просто вибрати профіль інспекцій коду перед комітом. За допомогою нової дії поруч з Show diff в діалозі Compare with branch можна отримати файл з іншої гілки. Нова дія Save to Shelf дозволяє скопіювати зміни на вкладку Shelf, зберігши їх при цьому в локальній історії.

Perforce-плагін тепер повністю сумісний з IntelliJ IDEA Community Edition і поставляється разом з IDE. Завдяки новим налаштуванням типографіки можна регулювати товщину звичайних і жирних шрифтів в меню Preferences / Settings | Editor / Fonts. Коли в редакторі відкрито кілька вкладок в режимі Split, будь-яку з них можна розгорнути подвійним кліком.

IDE підтримує вирази JSONPath з синтаксисом Goessner і Jayway. Тепер через меню Edit / Find можна викликати нову дію Evaluate JSONPath Expression для написання і тестування jsonpath-запитів.

IntelliJ IDEA підтримує формат JSON Lines і розпізнає файли з розширеннями .jsonl, .jslines, .ldjson і .ndjson. В IntelliJ IDEA з'явилася базова підтримка Java 16, що вийшла в березні цього року.

Було додано нові інспекції, в тому числі і для аналізу потоку даних. З'явилися нові налаштування, завдяки яким методи Builder, що викликаються по ланцюжку, стало простіше читати [14].

Було додано інтерфейс для зміни прав доступу при редагуванні об'єктів [14].

Додані шаблони Live Templates, за допомогою яких можна генерувати прості оператори прямо з Провідника баз даних. Навігація спростилася: дія go to declaration, викликана для об'єкта в SQL, тепер відкриває DDL, а не провідник бази даних [14].

Було покращено сортування даних [14].

З'явилася можливість копіювати, вирізати і вставляти джерела даних за допомогою всім відомих поєднань клавіш Ctrl/Cmd+C/V/X. Тепер ви можете редагувати дані в колекціях MongoDB. Також доступний попередній перегляд запитів [14].

У таблиці 2.2 наведено порівняння CMS за критеріями. Серед характеристик для порівняння виділімо:

- робота із сервлетами – вбудовані механізми роботи із технологію сервлетів;
- покращені механізми роботи із системами керування БД (СКБД) – чи наявні додаткові механізми роботи із СКБД;
- механізми роботи із Git – чи наявні вбудовані механізми підтримки систем управління версіями та колективної розробки Git;

- кроссфлатформність – чи є кроссфлатформні версії середовища розробки для майбутньої підтримки та модифікацій застосунку;
- плагіни та розширення – наявність та кількість плагінів та розширень для середовища, що спростить розробку;
- вартість використання – чи наявні безкоштовні версії середовища або схеми безкоштовного використання.

Таблиця 2.2 – Порівняння CMS

	IntelliJ IDEA	Eclipse	NetBeans
Робота із сервлетами	++	±	+
Покращені механізми роботи із СКБД	++	±	+
Механізми роботи із Git	++	++	++
Кроссфлатформність	++	++	–
Плагіни та розширення	+	++	±
Вартість використання	±	++	++

Як видно із порівняння усі три середовища розробки є лідерами галузі роботи із Java. Сильною стороною Eclipse завжди була його модульність – це середовище можна повністю адаптувати під власні потреби розробника за рахунок використання майже безкінечних плагінів, розширень та надбудов. Проте іноді це є негативною стороною – адже зустрічаються конфлікти версій та ризики несумісності. NetBeans – більш збалансоване середовище розробки, проте загальною проблемою цього варіанту завжди є певний конфлікт із найбільш актуальною версією Java. Більш того, NetBeans не має офіційних версій для інших ОС окрім Windows, що може стати проблемою при подальшій підтримці розроблених проєктів. IntelliJ IDEA – дійсно підтримує свою роль лідера і найголовнішою перевагою є досить розширений апарат розробника «із коробки», що гарантує сумісність та безконфліктність використання усіх механізмів. Проте, IntelliJ IDEA не має безкоштовної версії (окрім тестової), але надає механізми вільного

використання для здобувачів освіти. В решті решт, саме IntelliJ IDEA є найбільш оптимальним вибором з проміж усіх, що видно із таблиці порівняння.

2.2 Загальна структура додатку

Термін «клієнт-серверна архітектура» – збірне поняття, що складається з двох взаємодоповнюючих компонентів: сервера і, власне, клієнта [12]-[13], [15].

Клієнт – локальний комп'ютер на стороні віртуального користувача, який виконує відправку запиту до сервера для можливості надання даних або виконання певної групи системних дій [12]-[13], [15].

Сервер – дуже потужний комп'ютер або спеціальне системне обладнання, яке призначається для вирішення певного кола завдань по процесу виконання програмних кодів. Він виконує роботи сервісного обслуговування за клієнтськими запитами, надає користувачам доступ до певних системних ресурсів, зберігає дані або БД [12]-[13], [15].

Особливості такої моделі полягають в тому, що користувач відправляє певний запит на сервер, де той системно обробляється і кінцевий результат відсилається клієнту. У можливості сервера входить одночасне обслуговування відразу декількох клієнтів [12]-[13], [15].

Якщо одночасно надходить більше одного запиту, то такі запити встановлюються в певну чергу і сервером виконуються по черзі. Часом запити можуть мати свої власні пріоритети. Частина запитів з більш високими пріоритетами будуть постійно виконуватися в першочерговому порядку [12]-[13], [15].

Параметри, які можуть реалізуватися на стороні сервера:

- зберігання, захист і доступ до даних;
- робота з вступниками клієнтськими запитами;
- процес відправки відповіді клієнту.

Параметри, які можуть реалізуватися на стороні клієнта:

- майданчик з надання користувацького графічного інтерфейсу;
- формулювання запиту до сервера і його подальша відправка;
- отримання підсумків запиту і відправка додаткової групи команд (запити на додавання, оновлення інформації, видалення групи даних).

Архітектура системи клієнт-сервер формулює принципи віртуального спілкування між локальними комп'ютерами, а всі правила і принципи взаємодії знаходяться всередині протоколу.

Мережевий протокол – це особливий набір правил, на підставі якого виконується точна взаємодія між комп'ютерами всередині віртуальної мережі.

Застосунок «Система Деканат» буде мати клієнт-серверну архітектуру, що представлена на рисунку 2.1.

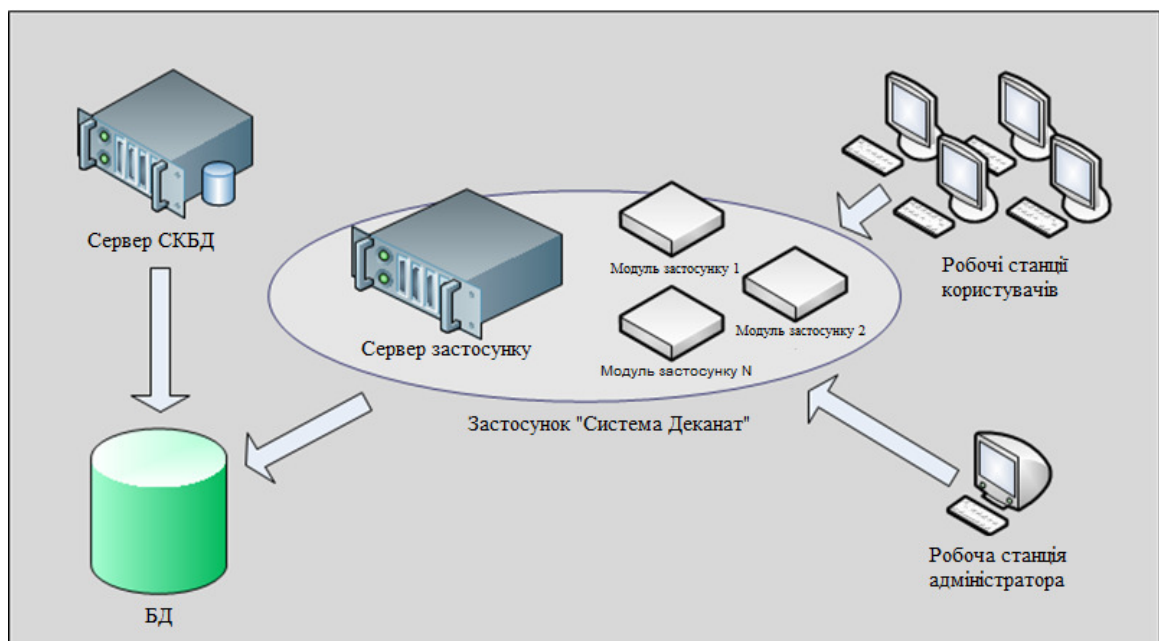


Рисунок 2.1 – Архітектура застосунку «Система Деканат»

2.3 Висновки за розділом 2

Таким чином, архітектура програмного засобу дозволяє вирішувати задачі, що були поставлені перед системою, пов'язані із електронною

комерцією. Засоби, що використовуються, забезпечують реалізацію функцій програми.

Для програмної реалізації можливостей ПЗ, що онлайн-магазину продажу годинників, виділених під час аналізу у попередньому розділі, проведено вибір засобів та обґрунтовано структуру системи, що проєктується.

Аналізуючи обрану структуру ПЗ, можна зробити висновок, що вона є лаконічною, але, разом з тим, досить працездатною, використовує ресурси ПК раціонально.

3 ОСНОВНІ РІШЕННЯ ЩОДО РЕАЛІЗАЦІЇ КОМПОНЕНТІВ СИСТЕМИ

3.1 Проєктування системи на основі предметної області

UML (Unified Modeling Language) – уніфікована мова моделювання, що використовується у парадигмі об’єктно-орієнтованого програмування та є невід’ємною частиною уніфікованого процесу розроблення ПЗ. UML заснована на діаграмах, що дають можливість представити систему у такому вигляді, щоб її можна було легко перевести у програмний код.

Діаграми класів – статичні структурні діаграми, що описують структуру системи, демонструють класи систем, їх атрибути, методи і залежності між класами [15]-[16].

Діаграма класів, розроблена для системи «Деканат» має вигляд представлений на рисунку 3.1.

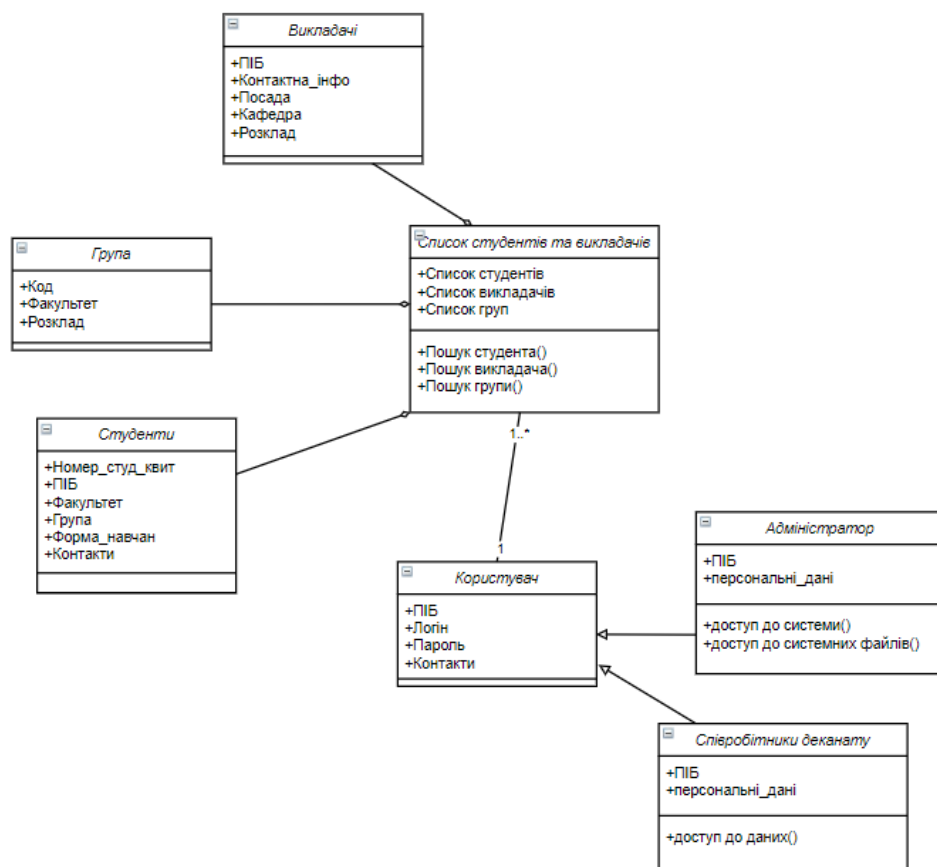


Рисунок 3.1 – Діаграма класів системи

Діаграми пакетів – структурні діаграми, основним вмістом яких є пакети і відношення між ними [15]-[16]. Ці діаграми слугують, в першу чергу, для організації елементів в групі по яким-небудь ознакам з метою спрощення структури і організації роботи з моделлю системи.

Система “Деканат” складається з трьох пакетів, що представлені на рисунок 3.2.

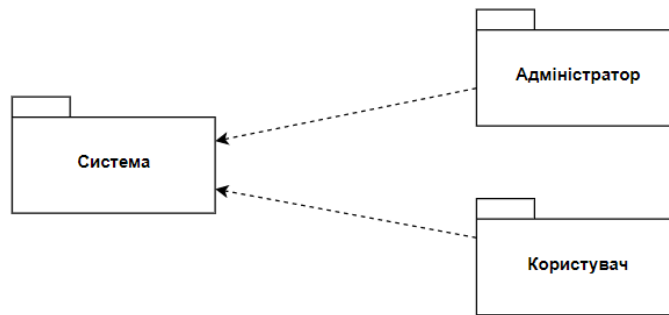


Рисунок 3.2 – Діаграма пакетів системи

Діаграма компонентів – статична структурна діаграма, що показує розбиття програмної системи на структурні компоненти і зв’язки (залежності) між компонентами. В якості фізичних компонент можуть виступати файли, бібліотеки, модулі, виконавчі файли, пакети і т.п. [15]-[16].

На рисунку 3.3 показана схема компонентів системи «Деканат».

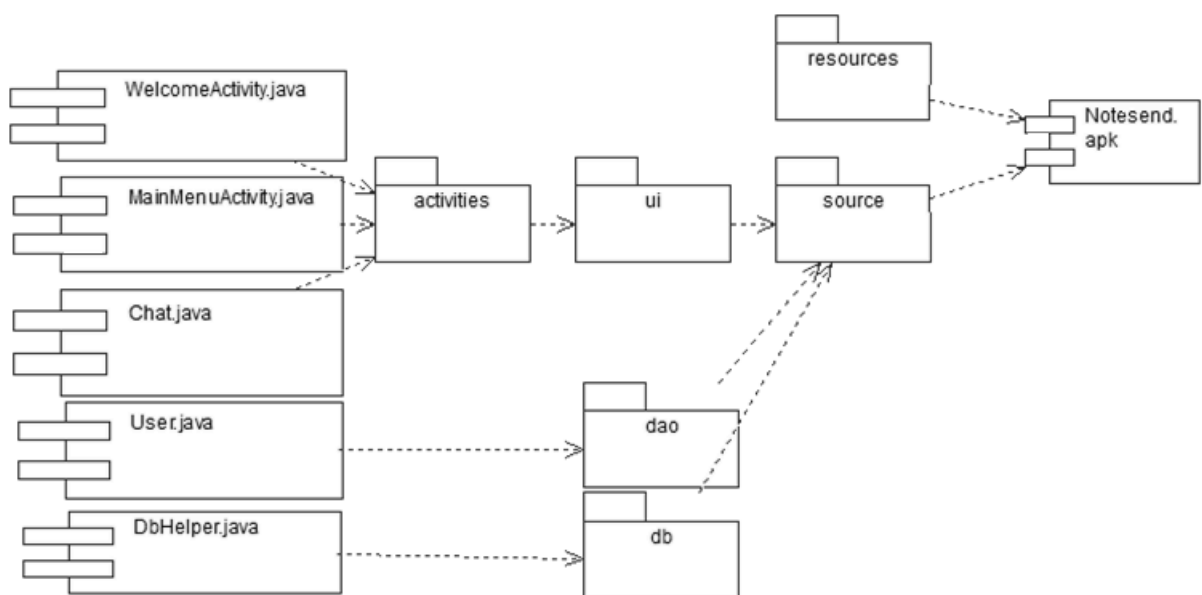


Рисунок 3.3 – Діаграма компонентів системи

Діаграми прецедентів застосовуються для моделювання вигляду системи з точки зору прецедентів (або варіантів використання) [15]-[16]. Даний вид діаграм використовується для моделювання динамічних аспектів системи, так само, як і діаграми станів та діяльності.

На рисунках 3.4 – 3.6 зображено діаграми прецедентів для системи «Деканат». Для них приведено базові та альтернативні потоки виконання.

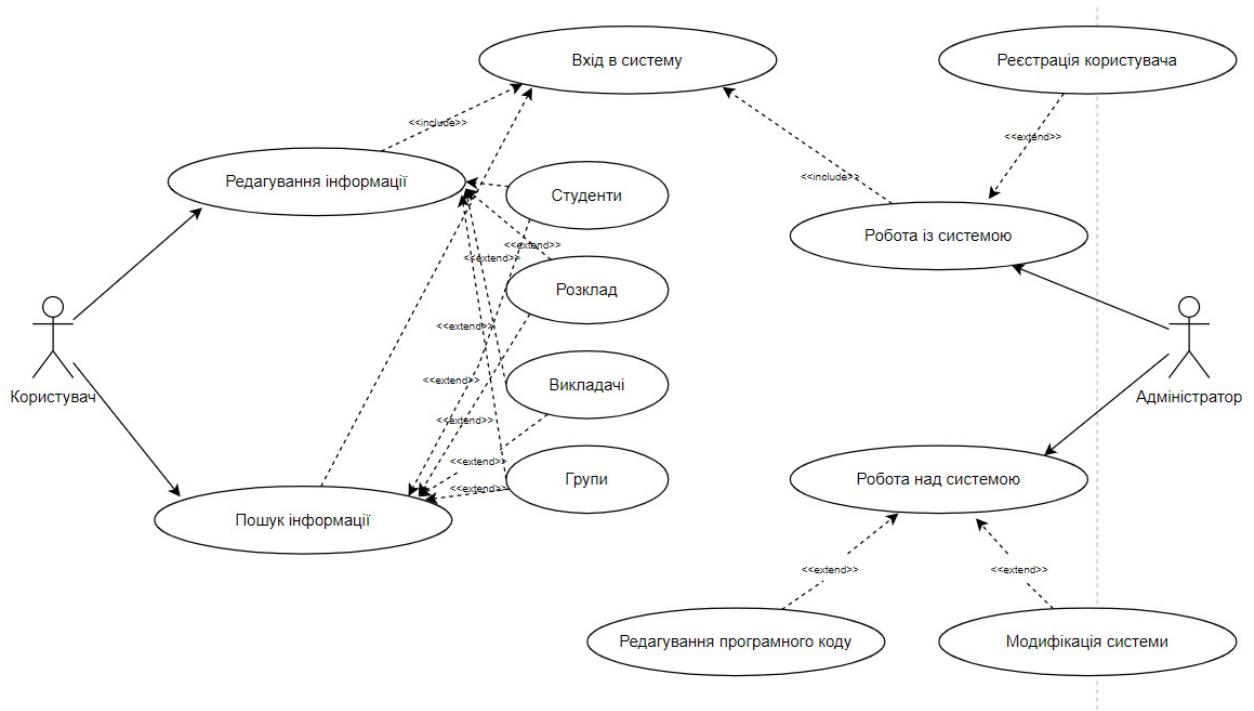


Рисунок 3.4 – Загальна діаграма прецедентів

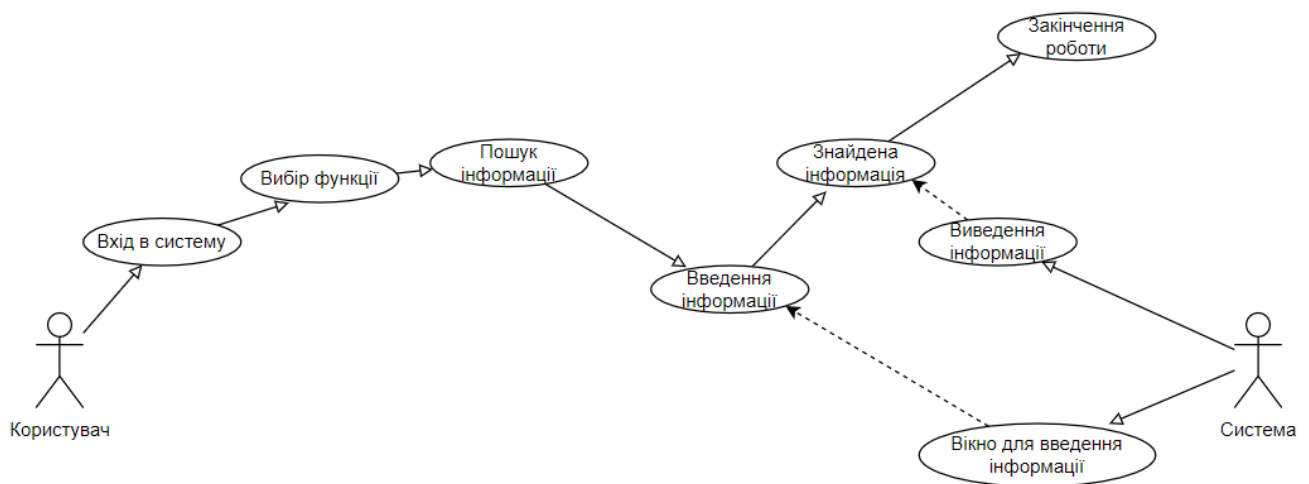


Рисунок 3.5 – Діаграма прецедентів «Пошук інформації»

Базовий сценарій:

- користувач обирає функцію пошуку;
- система виводить на екран вікно для введення інформації;
- користувач вводить інформацію, що потрібно знайти;
- система шукає введену інформацію;
- система видає знайдену інформацію на екран користувачеві;
- користувач отримує інформацію, яку шукав.

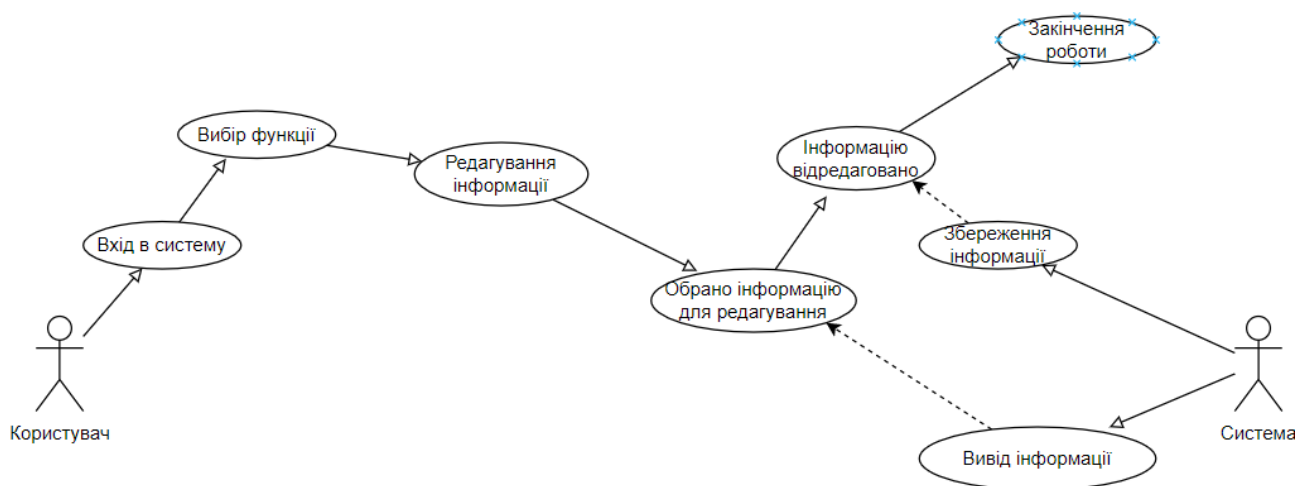


Рисунок 3.6 – Діаграма прецедентів «Редагування інформації»

Базовий сценарій:

- користувач обирає функцію редагування;
- система виводить на екран усю інформацію ;
- користувач обирає потрібну інформацію для редагування;
- користувач редагує інформацію;
- система зберігає внесені зміни.

На рисунках 3.7 – 3.9 зображено діаграми прецедентів для системи «Деканат».

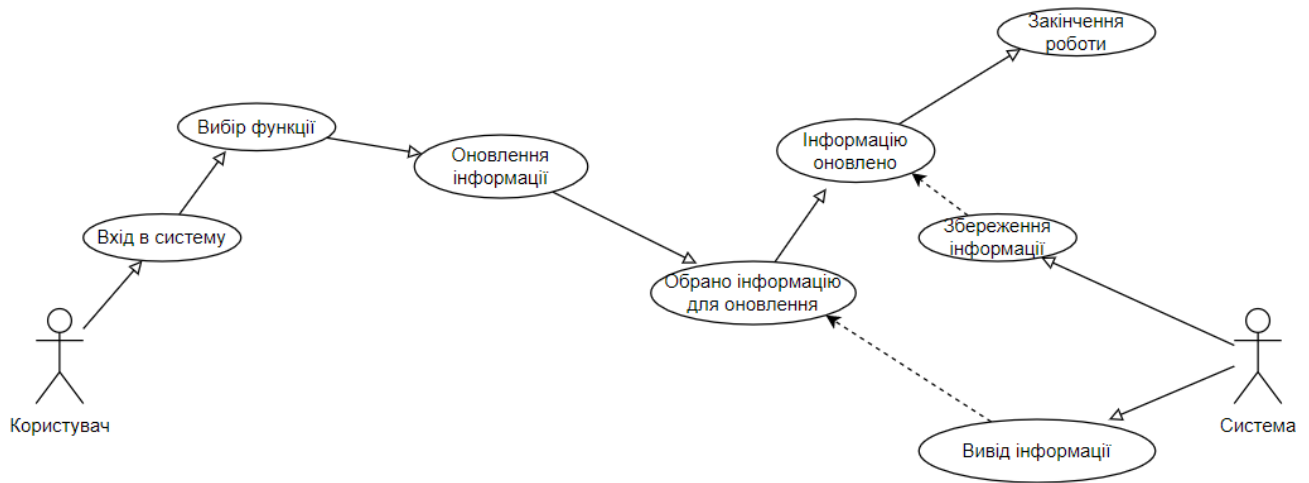


Рисунок 3.7 – Діаграма прецедентів «Оновлення інформації»

Базовий сценарій:

- користувач обирає функцію оновлення;
- система оновлює усю інформацію, що міститься у системі;
- користувач отримує оновлену інформацію.

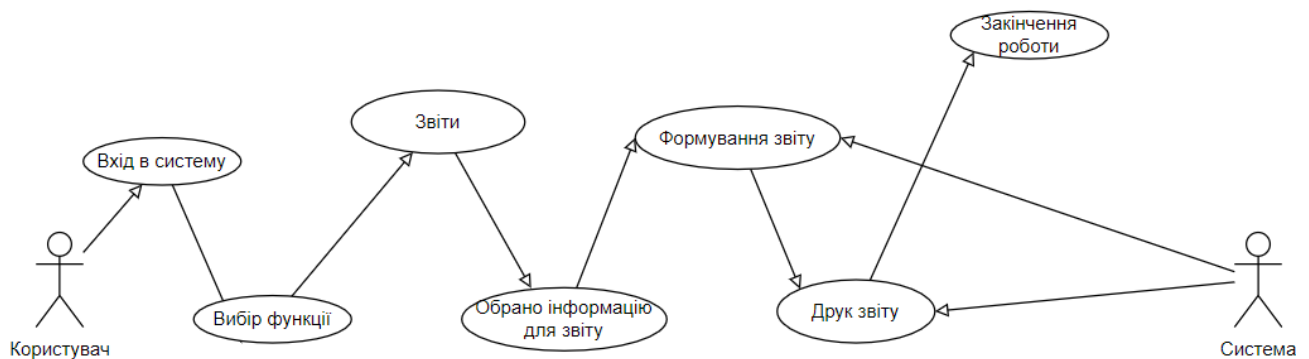


Рисунок 3.8 – Діаграма прецедентів «Формування звіту з інформації»

Базовий сценарій:

- користувач обирає функцію формування звіта;
- система виводить на екран вікно для формування звіта ;
- користувач обирає потрібну інформацію для формування звіту;

- користувач формує звіт;
- система зберігає зроблений звіт;
- користувач виводить звіт на друк;
- система запускає функцію друку;
- користувач отримує роздрукований звіт користувачу

Діаграми станів зображають всі можливі стани, в яких може знаходитись конкретній об'єкт, а також зміни стану об'єкту, які відбуваються в результаті впливу деяких подій на цей об'єкт [15]-[16].

На рисунку 3.9 зображено діаграму станів користувача у системі «Деканат». На ній показано переходи між станами користувача.

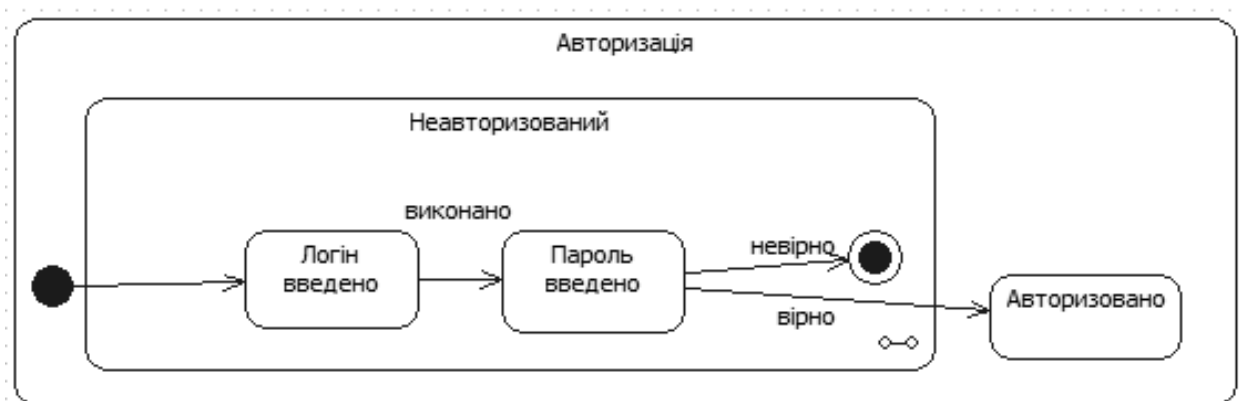


Рисунок 3.9 – Діаграма станів

Діаграма діяльності показує переходи між видами діяльності [15]-[16]. Модель видів діяльності може подавати в графічній формі потік подій для прецеденту.

Діаграма діяльності переходу користувачів до системи показана на рисунку 3.10.

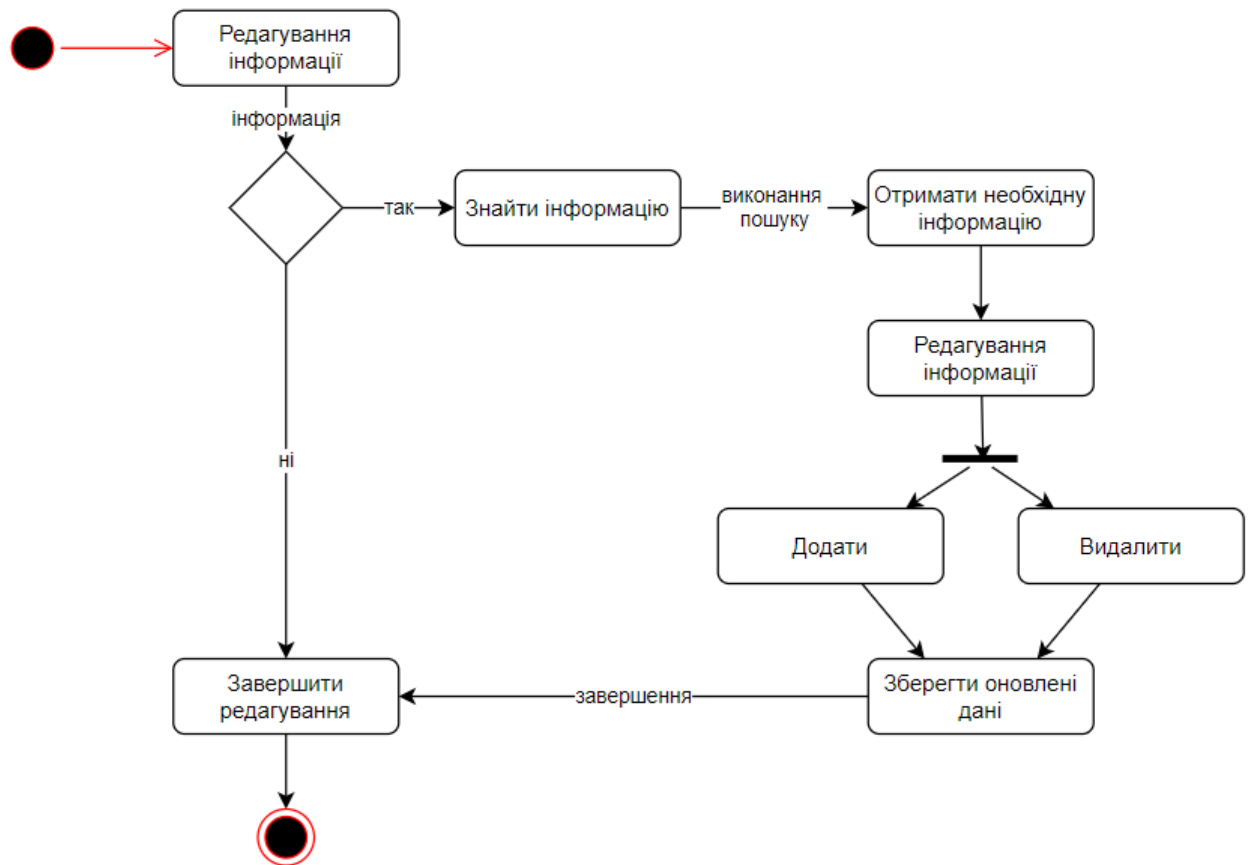


Рисунок 3.10 – Діаграма діяльності: редагування інформації

Діаграма кооперацій призначена для специфікації структурних аспектів взаємодії [15]-[16]. Головна особливість діаграми кооперації полягає в можливості графічно уявити не тільки послідовність взаємодії, але і всі структурні відносини між об'єктами, що беруть участь в цій взаємодії.

На рис. 3.11 – 3.12 зображено діаграми кооперацій показано взаємодію користувача та системи, коли виконуються операції: редагування, оновлення, пошук інформації, формування звіту, реєстрація користувача, видалення користувача. Всі інші операції у системі здійснюються аналогічним чином.

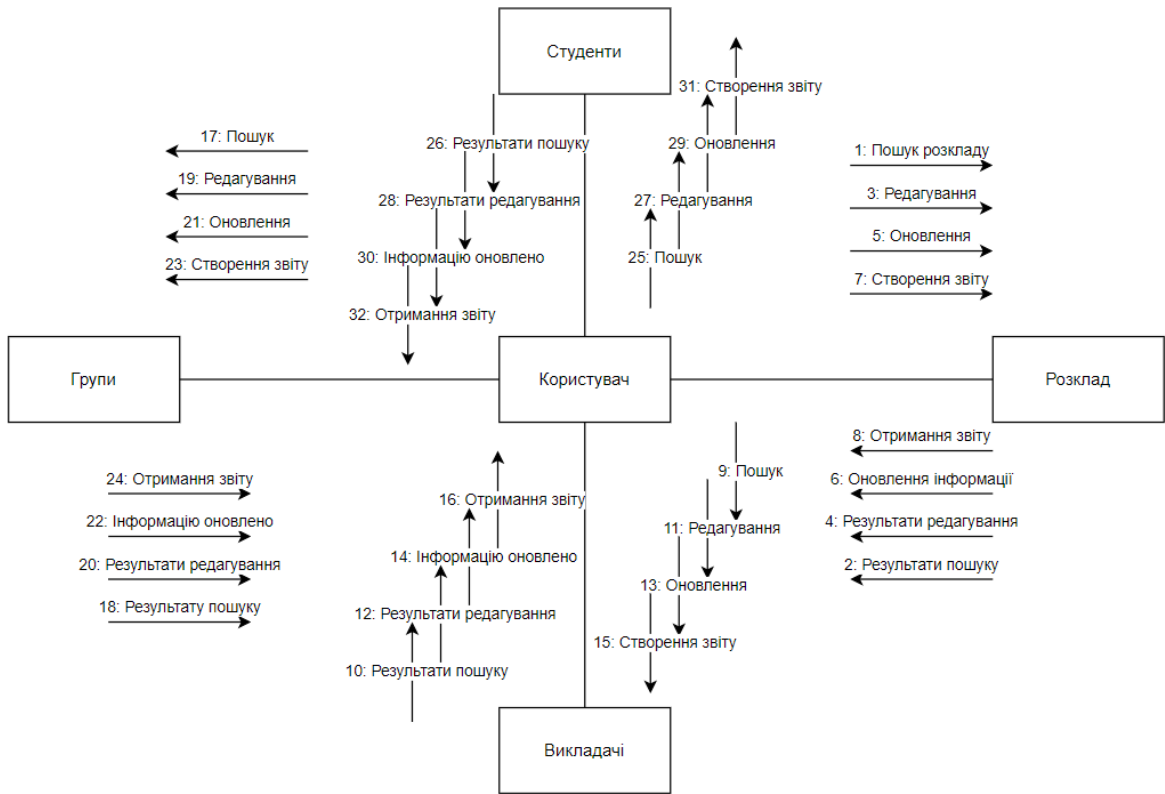


Рисунок 3.11 – Діаграма кооперації «Взаємодія користувача з даними»

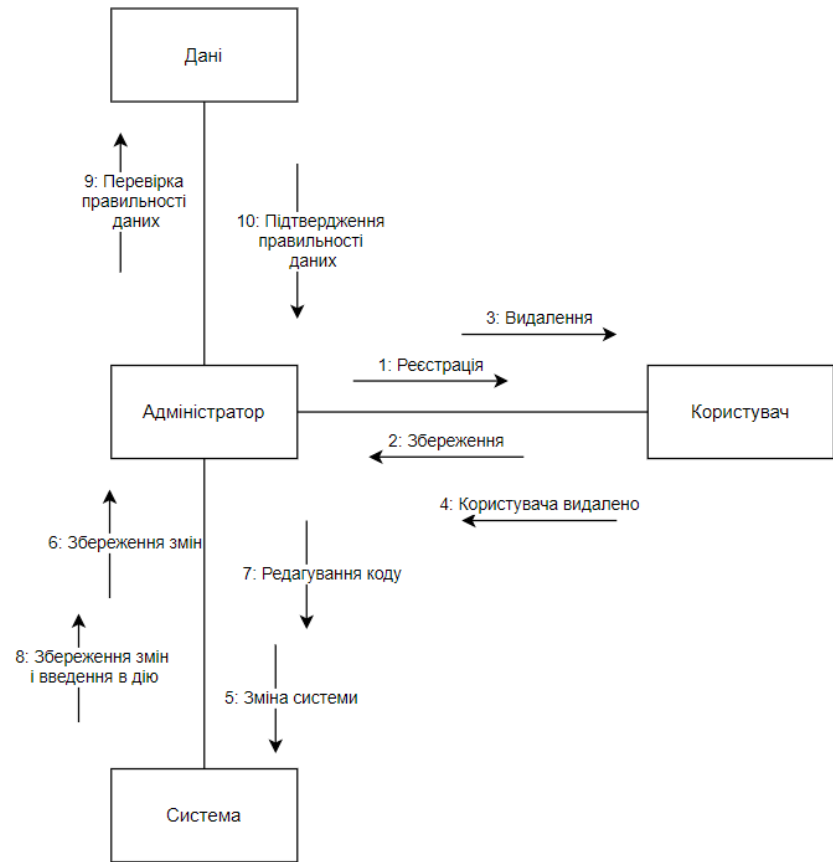


Рисунок 3.12 – Діаграма кооперації «Взаємодія адміністратора з системою»

3.2 Розробка архітектури застосунку

Для побудови архітектурного каркасу додатку використовують схему проєктування Model-view-controller (MVC) [17]. Це схема використання декількох шаблонів проєктування, за допомогою яких модель даних програми, користувацький інтерфейс і взаємодія з користувачем розділені на три окремих компоненти.

Логіка відображення – відповідає за збір даних від користувачів в різних формах та обробку їх у відповідності зі специфікацією [17].

Логіка управління даними – обробляє дані, отримані від користувача для передачі на наступний рівень.

Бізнес-логіка - сукупність правил, принципів, залежностей поведінки об'єктів предметної області. Це реалізація предметної області в інформаційній системі[17].

Особливості розробки архітектури веб-застосунку на основі предметної області [17]:

- можливість більш докладного опису логіки взаємодії об'єктів предметної області;
- можливість гнучкої зміни логіки роботи додатку;
- відсутність надлишкового коду в кінцевій реалізації;
- поділ логіки відображення, логіки управління даними і бізнес-логіки.

Недоліки розробки архітектури веб-застосунку на основі предметної області:

- велика трудомісткість при створенні проєкту;
- більш жорсткі вимоги до кваліфікації розробника.

На рисунку 3.13 показано, який вигляд має архітектура веб-застосунку системи на основі предметної області, а на рисунку 3.14 – на основі використання фреймворку технології сервлетів, але у даному випадку тільки розробка на основі предметної області може задовольнити потреби розробки.

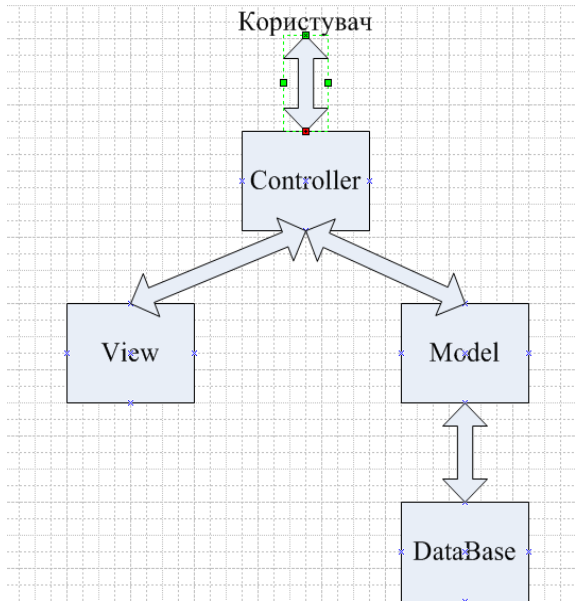


Рисунок 3.13 – Архітектура системи «Деканат» на основі предметної області

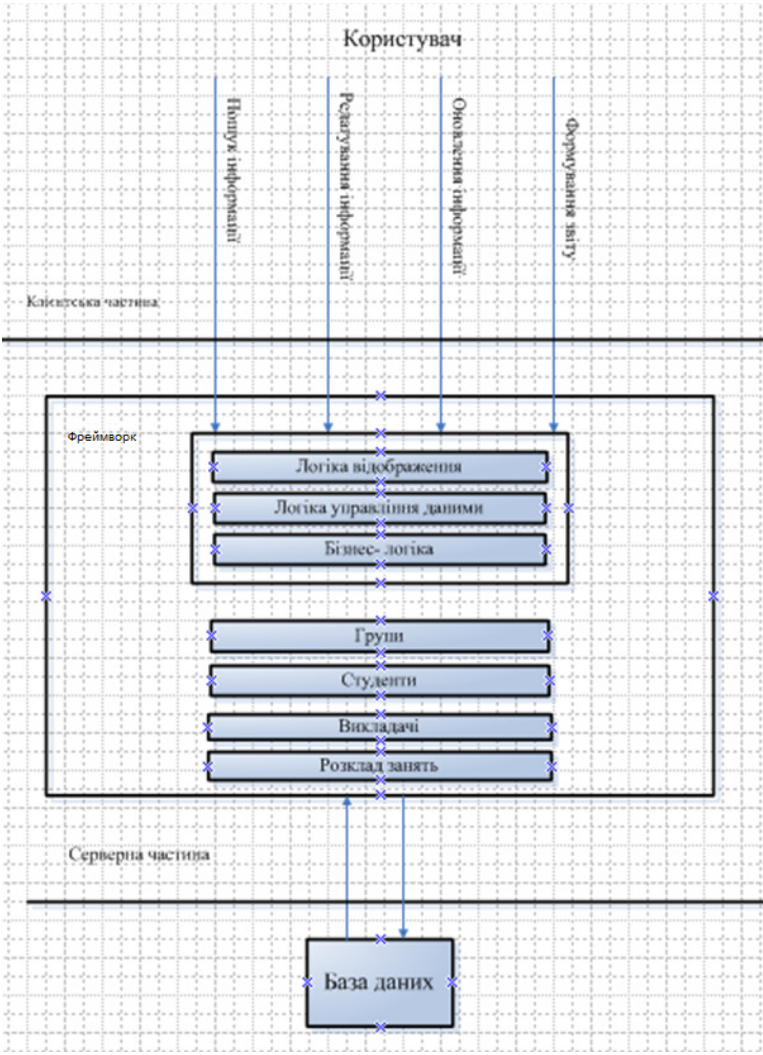


Рисунок 3.14 – Архітектура системи на основі використання фреймворку

3.3 Висновки за розділом 3

Для програмної реалізації системи автоматизованого моніторингу успішності студентів було проаналізовано сучасні засоби для веб-розробки на основі виявлених особливостей та архітектури застосунку, що розробляється в рамках даної дипломної кваліфікаційної роботи бакалавра. Було виявлені необхідні засоби. Отже, для розробки практичної частини роботи обрано:

- ① мову програмування Java;
- ↑ ① технологію Java--сервлетів;
- ↓ ① патерн архітектури MVC.

Проаналізувавши обрану клієнт-серверну архітектуру можна зробити висновок: застосування, що розроблюється є надійним, зручним у використанні, має відкриту структуру, що відкриває широкі можливості по адаптації та масштабуванню.

4 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ

4.1 Опис застосування

Розробка системи деканат для ЗВО, яка займається забезпеченням роботи з інформацією про дисципліни, студентів та викладачів, виконанням певних операцій над інформацією, має важливе місце у наданні зручності роботи працівників університету та деканатів. Система дозволить здійснювати швидко та упорядковано операції над інформацією та документами деканату. Дані переваги дозволять підвищити працездатність працівників університету.

Найменування розробки: «Розробка Системи Деканат». Умовне позначення теми розробки: розробка системи, що реалізує роботу деканатів ЗВО.

4.2 Умови виконання програми

Мінімальні системні вимоги та програмне забезпечення, що необхідні для функціонування серверної частини мобільного застосунку:

- ОС – будь-яка з різновидів операційної системи Unix (включаючи Mac OS X) або Windows;
- веб-сервер – HTTP-сервер Apache для Unix і Windows, сервер Microsoft Internet Information або сервер Netscape Enterprise;
- інтерпретатор PHP5;
- MySQL-сервер із правами створення та вилучення користувачів БД (або база SQLite з аналогічними правами).

Для експлуатації клієнтської частини необхідні такі програмно-технічні засоби:

- ПК;
- процесор із тактовою частотою від 1,5 GHz;

- обсяг оперативної пам'яті – не менше 3GB;
- підключення до мережі Internet;
- встановлений веб-оглядач останньої версії.

4.3 Інструкція по експлуатації програми

4.3.1 Звернення до програми

Для звернення до системи необхідно запустити сервер за допомогою іконки Start XAMPP, відкрити браузер, ввести адресу сайту та за необхідністю зареєструватися або авторизуватися на головній сторінці. Після відкриття доступу до системи, можливо переходити за посиланнями на головній сторінці веб-застосунку для виконання відповідних функцій.

4.3.2 Вхідні й вихідні дані

Вхідними даними є інформація про користувача: логін та пароль людини, що запрошує доступ до системи. Вхідна інформація зберігається у текстовій формі в таблицях бази даних на сервері.

Вихідною інформацією є дані, що хоче отримати користувач від системи, систематизовані в таблицях бази даних з можливістю фільтрування за допомогою SQL-запитів для представлення користувачеві на сторінках веб-застосунку.

4.4 Методика та результати тестування

При запуску додатку початковим екраном є екран авторизації/реєстрації (рисунок 4.1).

Система Деканат



Вітаємо у системі!

Введіть особисті дані для початку роботи із системою

Логін:

Пароль:

[Забули пароль?](#)

[Реєстрація](#)

Рисунок 4.1 – Початковий екран авторизації/реєстрації

Вікно призначене для авторизації користувача або його реєстрації. Якщо користувач вже зареєстрований то він повинен ввести свій логін та пароль та натиснути кнопку «Вхід» для подальшої роботи з системою. Якщо користувач не зареєстрований він обирає посилання «Реєстрація» та потрапляє на екран реєстрації (рисунок 4.2). Після завершення реєстрації користувач отримує персональні дані для входу та має можливість увійти у систему.

Система Деканат



Реєстрація

Логін:

ПІБ:

Пароль:

Повторіть пароль:

Посада:

E-mail:

Телефон:

Рисунок 4.2 – Початок реєстрації

Користувач може одразу завершити реєстрацію, натиснувши «Зареєструватися» (рисунок 4.3).

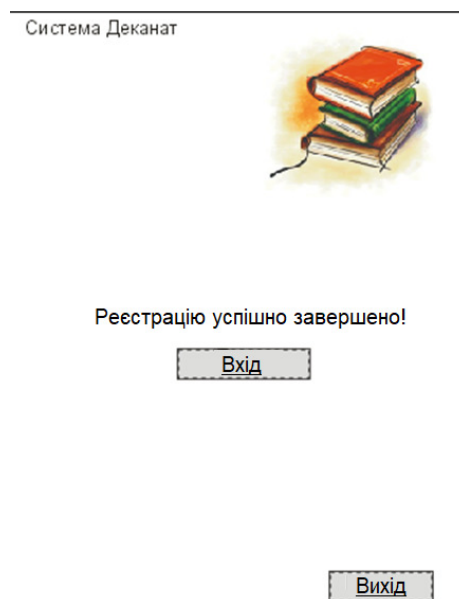


Рисунок 4.3 – Завершення реєстрації

Після авторизації користувач потрапляє у головне меню системи (рисунок 4.4).

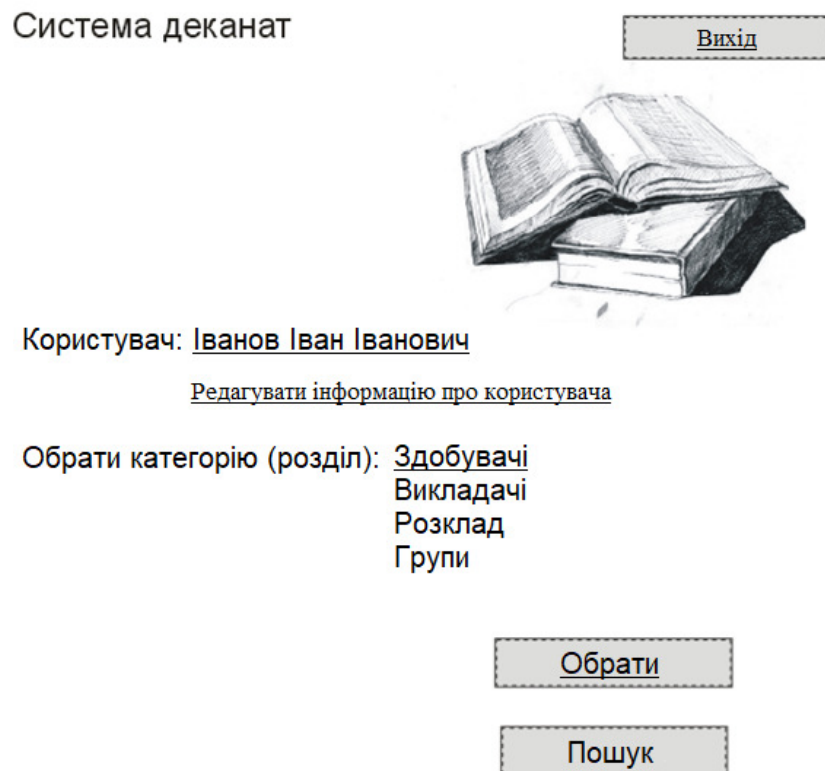


Рисунок 4.4 – Головне меню системи

Меню містить категорії інформації та функцію пошуку інформації.(рисунок 4.5). Також відображається статус користувача, а саме що він зараз у системі. Для виконання роботи з інформацією потрібно обрати певну категорію інформації та функцію, таку як: редагування,оновлення, формування звіту.

Система деканат



Користувач: Іванов Іван Іванович

Прізвище:	Петров_
Посада:	▼ Викладач Здобувач
Факультет:	КНТ ▼ ФБАД ІФ ЕФ ...

[Пошук](#)

Рисунок 4.5 – Виконання пошуку інформації

Користувач вводить інформацію та обирає її тип. Наприклад, для пошуку інформації про людину,потрібно ввести його фамілію, вибрати посаду, а саме студент він чи викладач;при обирання посади «Здобувач» потрібно вказати назву факультету, якщо вона відома, та натиснути кнопку «Пошук» та отримати результат (рисунок 4.6).

Система деканат

Користувач: Іванов Іван Іванович

Посада ▼	ПІБ ▼	Група ▼	Форма навчання ▼
Здобувач	Петров П.П.	КНТ-123	Денна
Здобувач	Петров І.М.	ІФ-321	Заочна

ОК

Рисунок 4.6 – Результат пошуку інформації

З прикладу, результатом пошуку інформації є знайдена інформація про студента. Для завершення пошуку та переходу до головного меню користувач повинен натиснути кнопку «Ок». Для завершення роботи з системою користувач повинен натиснути кнопку «Вихід» (рисунок 4.7). Для виконання функцій редагування, оновлення та формування звіту потрібно повернутися у головне меню та обрати потрібну категорію (рис. 3.11-3.14).

Система деканат

Вихід



Користувач: Іванов Іван Іванович

ПІБ	Група	Місце проживання	Телефон	Форма навчання	Стать
Ахонін Р.В.	ІНТ-123	вул.Дорошова 1	0123456789	Денна	Ж
Вірич В.С.	ІНТ-123	вул.Томаш 78	0123456789	Денна	М
Патокін І.П.	ІНТ-231	вул.Полуна 5	9876543210	Денна	М
Полін В.У	ІНТ-123	вул.Полуна 15	0123456789	Денна	М
Томашук Л.В.	ІНТ-123	вул.Полуна 9	0123456789	Денна	М
Торощ Г.Д.	ІНТ-231	вул.Полуна 71	9876543210	Денна	Ж
Торум З.О.	ІНТ-231	вул.Полуна 9	9876543210	Денна	Ж

Зберегти

Видалити

Рисунок 4.7 – Редагування інформації

Для збереження доданої або відредагованої інформації користувач повинен натиснути кнопку «Зберегти», а для видалення певної інформації користувач повинен натиснути кнопку «Видалити» (рисунок 4.8).

Система деканат



Користувач: Іванов Іван Іванович

Вихід

Інформацію оновлено!
Зміни збережено

ОК

Рисунок 4.8 – Завершення редагування

Після успішного редагування користувач для роботи з іншою категорією інформації або виконання іншої операції повинен натиснути кнопку «Ок». Якщо користувач бажає вийти з системи, він повинен натиснути кнопку «Вихід» (рисунок 4.9).

Система деканат

Вихід



Користувач: Іванов Іван Іванович

ПІБ	Група	Місце проживання	Телефон	Форма навчання	Стать
Ахонін Р.В.	КНТ-123	вул.Дорошова 4	0123456789	Денна	Ж
Вірич В.К.	КНТ-123	вул.Томаш 78	0123456789	Денна	М
Вітохін І.І.	КНТ-231	вул.Полуна 5	9876543210	Денна	М
Волод В.У.	КНТ-123	вул.Волод 12	0123456789	Денна	М
Томашук Л.В.	КНТ-123	вул.Полуна 9	0123456789	Денна	М
Торона Г.Д.	КНТ-231	вул.Порт 21	9876543210	Денна	Ж
Торум З.В.	КНТ-231s	вул.Полуна 9	9876543210	Зачіна	Ж

Звіти:

Заборгованності 2020

Заборгованності 2019

Заборгованності 2018

Успішність навчання

Додати
до звіту

Пуск

Рисунок 4.9 - Формування звіту

Користувач обирає потрібну операцію курсором миші та додає її у звіт натисненням клавіші «Додати до звіту». На сторінці існує список попередньо створених звітів, переглянути які користувач може, просто натиснувши на назву звіту. Для створення нового звіту після додавання у нього інформації потрібно натиснути кнопку «Пуск» (рисунок 4.10).

Система деканат

Вихід

Користувач: Іванов Іван Іванович

ПІБ	г	Група	Місце проживання	Телефон	Форма навчання	Стать
Ахонін Р.В.		КНТ-123	вул.Дорошова 4	0123456789	Денна	Ж
Вірил В.К.		КНТ-123	вул.Томаш 78	0123456789	Денна	М
Вітаскін І.І.		011-211	вул.Полуна 5	9876543210	Денна	М
Віткін В.У.		011-121	вул.Полуна 1	0123456789	Денна	М
Томашук Л.В.		КНТ-123	вул.Полуна 9	0123456789	Денна	М
Тороса С.Д.		КНТ-231	вул.Порт 21	9876543210	Денна	Ж
Торум З.В.		011-231s	вул.Полуна 9	9876543210	Звечірня	Ж

Друк

Зберегти

Рисунок 4.10 - Результат формування звіту

Після формування звіту користувач може зберегти його або роздрукувати. Для збереження користувач повинен натиснути кнопку «Зберегти», а для його друку користувач натискає кнопку «Друк». Після цього з'явиться діалогове вікно, що повідомляє про завершення обраної дії (рисунок 4.11).

При натисненні на кнопку «ОК» користувач переходить до головного меню. Для завершення роботи з системою користувач має натиснути кнопку «Вихід».

Система Деканат



Друк!
Принтер: Canon Pixma

ОК

Рисунок 4.11 – Завершення роботи зі звітами

ВИСНОВКИ

В ході виконання даної дипломної кваліфікаційної роботи бакалавра було проведено огляд та аналіз методів розробки та підтримки автоматизованих систем моніторингу для закладів ЗВО. Відібрані матеріали були систематизовані, проаналізовані, узагальнені та надані у першому розділі пояснювальної записки.

Для цього було виконано наступні завдання:

- розглянуто проблемну область;
- проаналізовано сучасні найбільш популярні аналоги систем моніторингу та документообігу;
- виконано проєктування веб-застосунку;
- розроблено веб-застосунок;
- розроблено сценарій та проведено тестування.

Для програмної реалізації автоматизованої системи моніторингу у вигляді веб-застосунку було проаналізовано сучасні засоби для проєктування та веб-застосунків на основі виявлених особливостей та архітектури застосунку, що розробляється в рамках даної дипломної кваліфікаційної роботи бакалавра. Було виявлені необхідні засоби. Отже, для розробки практичної було використано: мову програмування Java, технологію Java сервлетів; СКБД MySQL для проєктування та розробки бази даних з метою централізованого збереження інформації; веб-сервер XAMPP.

В результаті виконання завдання було отримано веб-застосунок, що відповідає всім вимогам представленим в технічному завданні, зручний для використання та виконує всі необхідні функції, для автоматизації моніторингу в ЗВО. Застосунок виконує наступні функції:

- створювати індивідуальний профіль, що містить в собі певну інформацію про користувача;
- мати доступ до інформації, що міститься у базі даних системи;
- знаходити необхідну інформацію;

- оформлювати звіти з, обраної користувачем, інформації;
- редагувати, а саме додавати, змінювати та видаляти інформацію;
- виводити створені або збережені звіти на друк.

В результаті виконання завдання було отримано систему, що відповідає всім вимогам представленим в технічному завданні, зручна для використання та виконує всі необхідні функції.

Усі завдання дипломної кваліфікаційної роботи бакалавра повністю виконано.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Гусєва О. Ю Діджиталізація – як інструмент удосконалення бізнес-процесів, їх оптимізація [Текст] / О. Ю Гусєва, С. В. Легомінова // Економіка. Менеджмент. Бізнес. – 2018. – № 1. – С. 33-39.
2. Ломовських Л. О. Діджиталізація економічних бізнес-процесів при прийнятті управлінських рішень у маркетинговій діяльності [Текст] / Л. О. Ломовських, М. В. Марченко, А. К. Гоел // Галицький економічний вісник. – 2019. – № 6. – С. 104-110.
3. Юхименко Ю. П. Проблеми інформатизації документообігу у вищих навчальних закладах України [Текст] / Ю. П. Юхименко // Бібліотекознавство. Документознавство. Інформологія. – 2008. – № 3. – С. 50-53.
4. Мартинюк Г. Ф. Вивчення електронного документообігу за дистанційною формою навчання з використанням інформаційно-комунікаційних технологій [Текст] / Г. Ф. Мартинюк. // Ефективна економіка. – 2017. – № 2.
5. Шепель І. В. Переваги системи електронного документообігу на підприємствах [Текст] / І. В. Шепель // Вчені записки Таврійського національного університету імені В. І. Вернадського. Серія : Економіка і управління. – 2020. – Т. 31(70), № 2(2). – С. 203-208.
6. Мельничук І. В. Формування внутрішнього документообігу в процесі створення ефективної системи управління інноваціями [Текст] / І. В. Мельничук // Економічний аналіз. – 2014. – Т. 17, № 2. – С. 160-167.
7. Політек-Софт «Деканат» / [Електрон. ресурс]. – Режим доступу: <http://www.politek-soft.kiev.ua/index.php?do=products&product=deanery>
8. Система «FossDoc»/ [Електрон. ресурс]. – Режим доступу: <https://fossdoc.com/ru>
9. Learning Platform «Moodle» / [Електрон. ресурс]. – Режим доступу: <https://moodle.org/?lang=ru>

10. Матвієнко О. Вимоги до функціональних можливостей програмних засобів (для бібліотечних процесів) [Текст] / О. Матвієнко // Бібліотечний вісник. – 1996. – № 4. – С. 8-9.
11. Kruglik V. Information system of software distribution / V. Kruglik // Інформ. технології в освіті : зб. наук. пр. – 2011. – Вип. 9. – С. 169-174.
12. Кадомський К. К. Java. Теорія і практика [Текст] : навч. посіб. / К. К. Кадомський, П. К. Ніколюк ; Донец. нац. ун-т ім. Василя Стуса, Фізико-техн. ф-т, Каф. комп'ютер. технологій. – Вінниця : ДонНУ ім. Василя Стуса, 2019. – 229 с.
13. Коновалюк М.М. Java 13. Практический курс [Текст] : [учеб. пособие] / М. М. Коновалюк. – Киев : Шмидко Т. С., 2019. – 362 с.
14. IntelliJ IDEA / [Електрон. ресурс]. – Режим доступу: <https://www.jetbrains.com/ru-ru/idea/>
15. Императивное программирование и объектно-ориентированное моделирование: Java, UML, OCL [Текст] : учеб. пособие для студентов высш. учеб. заведений / А. Ф. Верлань [и др.]. – Одесса : Экология, 2013. – 430 с.
16. Постіл С. Д. UML. Уніфікована мова моделювання інформаційних систем [Текст] : навч. посіб. / С. Д. Постіл ; Ун-т держ. фіск. служби України. – Ірпінь : Ун-т держ. фіск. служби України, 2019. – 321 с.
17. Пурський О. І. Метод побудови мережі вітрин інтернет-магазинів на основі архітектури MVC [Текст] / О. І. Пурський, Д. П. Мазоха // Бізнес Інформ. – 2017. – № 10. – С. 319-324.

ДОДАТОК А
Технічне завдання

Вступ

Розробка системи деканат для ЗВО, яка займається забезпеченням роботи з інформацією про дисципліни, студентів та викладачів, виконанням певних операцій над інформацією, має важливе місце у наданні зручності роботи працівників університету та деканатів. Система дозволить здійснювати швидко та упорядковано операції над інформацією та документами деканату. Дані переваги дозволять підвищити працездатність працівників університету.

Найменування розробки: «Розробка Системи Деканат». Умовне позначення теми розробки: розробка системи, що реалізує роботу деканатів ЗВО.

A.1 Підстава для розробки

Підставою для розробки слугує завдання на дипломну кваліфікаційну роботу на тему «Програмна реалізація автоматизованої системи моніторингу для деканату», затверджене наказом Національного університету «Запорізька політехніка» № 103 від 30 березня 2021 р.

A.2 Призначення розробки

Система призначена для роботи з інформацією та документацією університетів. Обробка, керування та редагування інформації буде проводитися за допомогою нашої системи у деканатах університету.

А.3 Основні вимоги до програми, що розробляється

А.3.1 Вимоги до функціональних характеристик

Нижче представлено перелік функціональних характеристик системи:

- наявність та редагування списку студентів, що вчаться на факультеті;
- наявність та редагування інформації про форму навчання;
- можливість розрахунку оцінок студентів за навчальний період;
- можливість розрахунку середнього балу студентів згідно з оцінками;
- наявність та оновлення списку студентів, що отримують стипендію;
- наявність та редагування списку студентів, що отримують освіту екстерном;
- наявність та оновлювання списку іноземних студентів;
- наявність та редагування списку груп факультету та загальна їх кількість;
- оновлення списку студентів, що навчаються на пільговій основі;
- редагування розклад занять для студентів;
- редагування розклад занять для викладачів факультету;
- можливість видачі довідок;
- система повинна здійснювати збір інформації про студентів та викладачів;
- можливість виконувати пошук потрібної інформації у системі ;
- легке користування системою.

A.3.2 Вимоги до надійності

Система потребує виконання наступних вимог до надійності, безпеки та експлуатації:

- система повинна працювати постійно без збою;
- займатися підтримкою системи повинен адміністратор;
- доступ к даним повинен мати тільки користувач за наявності персональних логіна та пароля для входу у систему;
- система обов’язково повинна зберігати дані при виникненні роботи з нею;
- бажано щоб програма працювала без додаткового налаштування.

A.3.3 Умови експлуатації

Програма може зберігатися та переноситися на будь-яких існуючих носіях інформації.

Експлуатація програми здійснюється згідно експлуатаційної документації на розроблену програму, яка відповідає стандартам і містить інформацію, необхідну для її освоєння та експлуатації.

A.3.4 Вимоги до складу та параметрів технічний засобів

Мінімальні системні вимоги та програмне забезпечення, що необхідні для функціонування серверної частини мобільного застосунку:

- ОС – будь-яка з різновидів операційної системи Unix (включаючи Mac OS X) або Windows;
- веб-сервер – HTTP-сервер Apache для Unix і Windows, сервер Microsoft Internet Information або сервер Netscape Enterprise;
- інтерпретатор PHP5;

- MySQL-сервер із правами створення та вилучення користувачів БД (або база SQLite з аналогічними правами).

Для експлуатації клієнтської частини необхідні такі програмно-технічні засоби:

- ПК;
- процесор із тактовою частотою від 1,5 GHz;
- обсяг оперативної пам'яті – не менше 3GB;
- підключення до мережі Internet;
- встановлений веб-оглядач останньої версії.

A.3.5 Вимоги до маркування і пакування

Програма може бути записана на переносний носій або розповсюджуватися через хмарні сховища. На впакуванні повинна бути назва програми – «Система Деканат».

A.3.6 Вимоги до транспортування й зберігання

Вимоги до транспортування й зберігання аналогічні вимогам, пропонованим до носія, на якому записана програма.

A.3.7 Вимоги до програмної документації

Програмний продукт повинен подаватися у наступному складі:

- технічне завдання;
- опис програми;
- текст програми.

A.4 Стадії та етапи розробки

Виділяють такі етапи в розробці програмного забезпечення:

→① збирання та аналіз вимог. Проводиться збір вимог до програмного забезпечення, їх систематизація, документування, аналіз, виявлення протиріч, неповноти, рішення конфліктів в процесі розробки програмного забезпечення;

↑① розробка ТЗ. На основі раніше зібраних вимог формується вихідний документ (ТЗ) на проєктування програмного забезпечення. Описується всі вимоги до створюваного продукту;

↓① створення логотипу та дизайн основних сторінок;

↗① налаштування БД;

↗① розробка структури. Дозволяє оцінити інформативність та компоновку елементів системи на початковому етапі розробки;

↘① підключення адміністративного інтерфейсу. Має містити повне управління настройками системи, окремих модулів та контентом сайту;

↘① робота над програмним кодом. Розробка основного функціоналу системи;

←① наповнення та тестування. Професійні тестувальники знаходять «баги» у системі та фіксують їх. Заповнюється базовий «контент», щоб сайт мав привабливіший вигляд для перших користувачів;

→① розміщення та відкритий доступ, технічна підтримка. Соціальна мережа розміщується на сайті унтернет-магазину та стає доступною для всіх користувачів;

↓① оформлення документації.

A.5 Порядок контролю та приймання системи

Приймання системи здійснюється відповідно до складу та змісту робіт, наведених у Технічному завданні, і відповідно до умов контракту.

Звіт про проведену роботу направляється Замовнику на паперовому носії в 2-х екземплярах, підписаних належним чином, а також на електронному носії.

Вимоги до оформлення проміжних звітів і підсумкового звіту узгоджуються з Виконавцем при підписанні контракту.

Одночасно з підсумковим звітом про виконання робіт Виконавець надає Замовнику підписаний зі сторони Виконавця Акт здачі-приймання виконаної роботи.

ДОДАТОК Б
Опис програми

Б.1 Загальні відомості

Назва програмного продукту – «Система Деканат».

ПЗ було розроблено з використанням мови програмування Java та Java сервлетів.

Б.2 Функціональне призначення

Програмний продукт виконує наступні функції:

- реєстрація користувачів;
- вивід інформації про категорії даних;
- перегляд інформації;
- пошук інформації;
- додавання інформації;
- видалення інформації;
- оформлення звіту з обраної інформації;
- вивід створених або збережених звітів на друк.

Б.3 Опис логічної структури

Логічну структуру програми визначає її функціональне призначення можливості обраних програмних і інструментальних засобів проєктування.

Логічну структуру програмного продукту можна представити у вигляді ієрархічної моделі, представленої на рисунку Б.1.

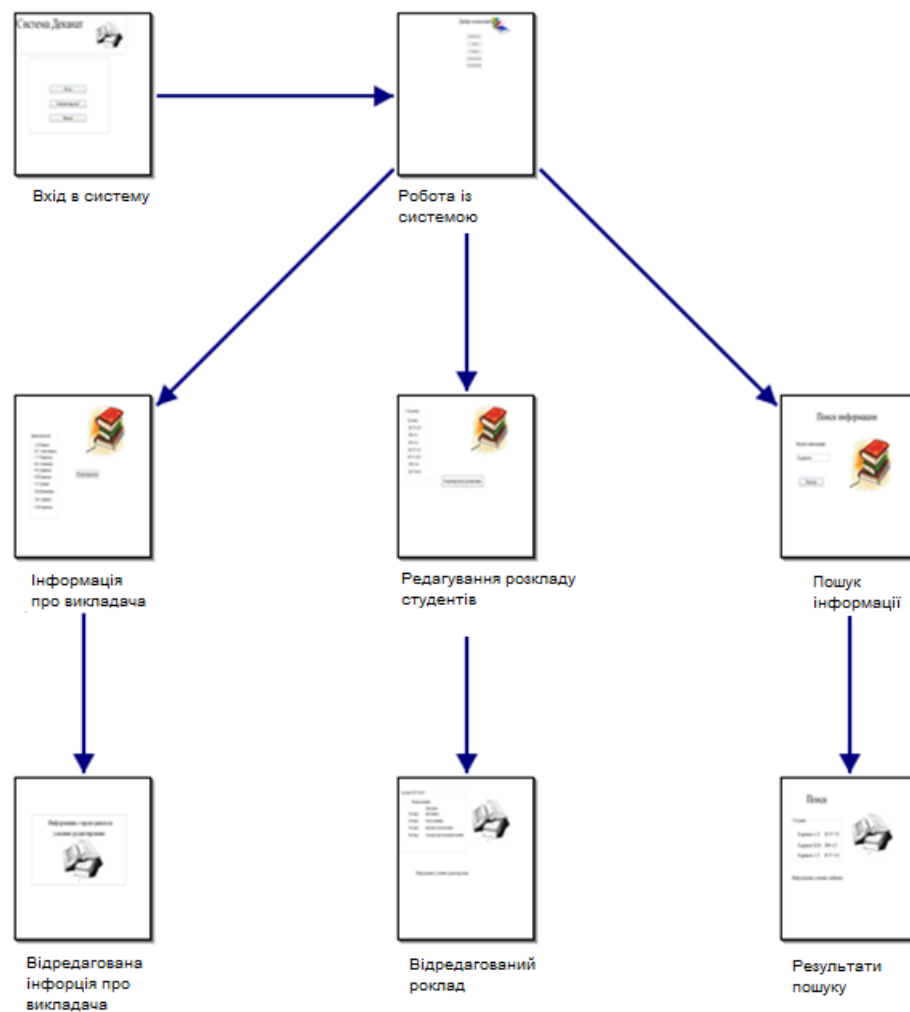


Рисунок Б.1 – Логічна структура програми

Б.4 Використані технічні засоби

Мінімальні системні вимоги та програмне забезпечення, що необхідні для функціонування серверної частини мобільного застосунку:

- ОС – будь-яка з різновидів операційної системи Unix (включаючи Mac OS X) або Windows;
- веб-сервер – HTTP-сервер Apache для Unix і Windows, сервер Microsoft Internet Information або сервер Netscape Enterprise;

- інтерпретатор PHP5;
- MySQL-сервер із правами створення та вилучення користувачів БД (або база SQLite з аналогічними правами).

Для експлуатації клієнтської частини необхідні такі програмно-технічні засоби:

- ПК;
- процесор із тактовою частотою від 1,5 GHz;
- обсяг оперативної пам'яті – не менше 3GB;
- підключення до мережі Internet;
- встановлений веб-оглядач останньої версії.

Б.5 Виклик і завантаження

Запустити локальний сервер Денвер, натиснувши 2 рази на робочому столі «Start Denwer». Денвер треба запускати завжди, коли потрібно працювати зі своїм сайтом на локальному сервері. Потім ввести в адресному рядку браузера назву сайту – `http://site`. Далі налаштувати підключення до бази даних на стороні локального сервера. ввести у відповідні поля форми свої дані: ім'я користувача та ім'я бази даних (поле "пароль" необхідно залишити порожнім) - інші поля залишаються без змін. Ввести назву і опис сайту. Введені дані доступу до системи будуть логіном і паролем, які треба використовувати для авторизації в панелі управління сайтом.

Б.6 Вхідні та вихідні дані

Вхідними даними є інформація про користувача: логін та пароль людини, що запрошує доступ до системи. Вхідна інформація зберігається у текстовій формі в таблицях бази даних на сервері. Також до вхідної інформації можна віднести всі дані, що відносяться до певного користувача.

Вихідною інформацією є дані, які хоче отримати користувач від системи, систематизована в таблицях бази даних з можливістю фільтрування за допомогою SQL-запитів для представлення користувачеві на сторінках веб-застосунку.

ДОДАТОК В
Текст програми

```

public class AdminSession extends JFrame {
    static AdminSession frame;
    private JPanel contPane;
    JScrollPane sp;

    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    frame = new AdminSession();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    /**
     * Create the frame.
     */
    public AdminSession() {
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setBounds(100, 100, 450, 300);
        contPane = new JPanel();
        contPane.setBorder(new EmptyBorder(5, 5, 5, 5));

        setContentPane(contPane);

        JLabel lblAdminSession = new JLabel("Admin
Section");
        lblAdminSession.setBackground(Color.DARK_GRAY);
        lblAdminSession.setFont(new Font("Tahoma",
Font.PLAIN, 20));

        JButton btnAddAccant = new JButton("Add Accant");
        btnAddAccant.addActionListener(new
ActionListener() {
            public void actionPerformed(ActionEvent e) {
                AddAccant.main(new String[]{});
                frame.dispose();
            }
        });

        JButton btnViewAccant = new JButton("View
Accant");
        btnViewAccant.addActionListener(new
ActionListener() {
            public void actionPerformed(ActionEvent e) {
                ViewAccant.main(new String[]{});
            }
        });
    }
}

```

```

        JButton btnLogout = new JButton("Logout");
        btnLogout.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                FeeReport.main(new String[]{});
                frame.dispose();
            }
        });
        GroupLayout gl_contPane = new
GroupLayout(contPane);
        gl_contPane.setHorizontalGroup(

            gl_contPane.createParallelGroup(Alignment.LEADING)

                .addGroup(gl_contPane.createSequentialGroup())

                .addGroup(gl_contPane.createParallelGroup(Alignment.LEADING)

                    .addGroup(gl_contPane.createSequentialGroup()
                        .addGap(161)

                    .addComponent(lblAdminSession))

                .addGroup(gl_contPane.createSequentialGroup()
                    .addGap(149)

                .addGroup(gl_contPane.createParallelGroup(Alignment.LEADING)

                    .addComponent(btnViewAccant, GroupLayout.PREFERRED_SIZE,
130, GroupLayout.PREFERRED_SIZE)

                    .addComponent(btnAddAccant, GroupLayout.PREFERRED_SIZE, 130,
GroupLayout.PREFERRED_SIZE)

                    .addComponent(btnLogout, GroupLayout.PREFERRED_SIZE, 130,
GroupLayout.PREFERRED_SIZE)))
                .addContainerGap(136,
Short.MAX_VALUE))
        );
        gl_contPane.setVerticalGroup(

            gl_contPane.createParallelGroup(Alignment.LEADING)

                .addGroup(gl_contPane.createSequentialGroup()
                    .addComponent(lblAdminSession)
                    .addGap(29)
                    .addComponent(btnAddAccant,
GroupLayout.PREFERRED_SIZE, 34, GroupLayout.PREFERRED_SIZE)
                    .addGap(18)
                    .addComponent(btnViewAccant,
GroupLayout.PREFERRED_SIZE, 34, GroupLayout.PREFERRED_SIZE)
                    .addGap(18)
                    .addComponent(btnLogout,
GroupLayout.PREFERRED_SIZE, 34, GroupLayout.PREFERRED_SIZE)

```

```

        .addContainerGap(59,
Short.MAX_VALUE))
        );
        contPane.setLayout(gl_contPane);
    }

}

public class AddStud extends JFrame {
    static AddStud frame;
    private JPanel contPane;
    private Jtxt_fld txt_fld;
    private Jtxt_fld txt_fld_1;
    private Jtxt_fld txt_fld_2;
    private Jtxt_fld txt_fld_3;
    private Jtxt_fld txt_fld_4;
    private Jtxt_fld txt_fld_5;
    private Jtxt_fld txt_fld_6;
    private Jtxt_fld txt_fld_7;
    private Jtxt_fld txt_fld_8;
    private Jtxt_fld txt_fld_9;
    JTextArea textArea;
    /**
     * Launch the application.
     */
    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    frame = new AddStud();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    /**
     * Create the frame.
     */
    public AddStud() {
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setBounds(100, 100, 450, 503);
        contPane = new JPanel();
        contPane.setBorder(new EmptyBorder(5, 5, 5, 5));
        setContentPane(contPane);

        JLabel lblAddAccant = new JLabel("Add Stud");
        lblAddAccant.setForeground(Color.DARK_GRAY);
        lblAddAccant.setFont(new Font("Tahoma",
Font.PLAIN, 20));

        JLabel lblName = new JLabel("Name:");

```

```

JLabel lblEmail = new JLabel("Email:");

JLabel lblCourse = new JLabel("Course:");

JLabel lblFee = new JLabel("Fee:");

JLabel lblPaid = new JLabel("Paid:");

JLabel lblDue = new JLabel("Due:");

JLabel lblAddress = new JLabel("Address:");

JLabel lblCity = new JLabel("City:");

JLabel lblState = new JLabel("State:");

JLabel lblCountry = new JLabel("Country:");

JLabel lblContactNo = new JLabel("Contact No:");

JButton btnAddAccant = new JButton("Add Stud");
btnAddAccant.addActionListener(new
ActionListener() {
    public void actionPerformed(ActionEvent e) {
        String name=txt_fld.getText();
        String email=txt_fld_1.getText();
        String course=txt_fld_2.getText();
        int
fee=Integer.parseInt(txt_fld_3.getText());
        int
paid=Integer.parseInt(txt_fld_4.getText());
        int
due=Integer.parseInt(txt_fld_5.getText());
        String address=textArea.getText();
        String city=txt_fld_6.getText();
        String state=txt_fld_7.getText();
        String country=txt_fld_8.getText();
        String contactno=txt_fld_9.getText();

        Stud s=new
Stud(name,email,course,fee,paid,due,address,city,state,country,c
ontactno);

        int status=StudDao.save(s);

        if(status>0){

            JOptionPane.showMessageDialog(AddStud.this,"Stud added
successfully!");

            txt_fld.setText("");txt_fld_1.setText("");txt_fld_2.setText(
"");

```

```

        txt_fld_3.setText("");txt_fld_4.setText("");txt_fld_5.setTex
t("");

        txt_fld_6.setText("");txt_fld_7.setText("");txt_fld_8.setTex
t("");

        txt_fld_9.setText("");textArea.setText("");
        }else{

        JOptionPane.showMessageDialog(AddStud.this,"Sorry, Unable to
add Stud!");

        }

    });

    txt_fld = new Jtxt_fld();
    txt_fld.setColumns(10);

    txt_fld_1 = new Jtxt_fld();
    txt_fld_1.setColumns(10);

    txt_fld_2 = new Jtxt_fld();
    txt_fld_2.setColumns(10);

    txt_fld_3 = new Jtxt_fld();
    txt_fld_3.setColumns(10);

    txt_fld_4 = new Jtxt_fld();
    txt_fld_4.setColumns(10);

    txt_fld_5 = new Jtxt_fld();
    txt_fld_5.setColumns(10);

    txt_fld_6 = new Jtxt_fld();
    txt_fld_6.setColumns(10);

    txt_fld_7 = new Jtxt_fld();
    txt_fld_7.setColumns(10);

    txt_fld_8 = new Jtxt_fld();
    txt_fld_8.setColumns(10);

    txt_fld_9 = new Jtxt_fld();
    txt_fld_9.setColumns(10);

    textArea = new JTextArea();
    textArea.setRows(3);
    textArea.setColumns(20);

    JButton btnBack = new JButton("back");
    btnBack.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {

```

```

        AccantSection.main(new String[]{});
        frame.dispose();
    }
});
    GroupLayout gl_contPane = new
GroupLayout (contPane);
    gl_contPane.setHorizontalGroup(

gl_contPane.createParallelGroup(Alignment.LEADING)

.addGroup(gl_contPane.createSequentialGroup())

.addGroup(gl_contPane.createParallelGroup(Alignment.LEADING)

.addGroup(gl_contPane.createSequentialGroup()
                .addGap(142)

.addComponent(lblAddAccant))

.addGroup(gl_contPane.createSequentialGroup()
                .addContainerGap()

.addGroup(gl_contPane.createParallelGroup(Alignment.LEADING)

.addComponent(lblName)

.addComponent(lblCourse,      GroupLayout.PREFERRED_SIZE,      48,
GroupLayout.PREFERRED_SIZE)

.addComponent(lblFee,      GroupLayout.PREFERRED_SIZE,      31,
GroupLayout.PREFERRED_SIZE)

.addComponent(lblPaid,      GroupLayout.PREFERRED_SIZE,      31,
GroupLayout.PREFERRED_SIZE)

.addComponent(lblDue,      GroupLayout.PREFERRED_SIZE,      31,
GroupLayout.PREFERRED_SIZE)

.addComponent(lblAddress,      GroupLayout.PREFERRED_SIZE,      64,
GroupLayout.PREFERRED_SIZE)

.addComponent(lblEmail,      GroupLayout.PREFERRED_SIZE,      56,
GroupLayout.PREFERRED_SIZE))
                .addGap(33)

.addGroup(gl_contPane.createParallelGroup(Alignment.LEADING,
false)

.addComponent(textArea,      GroupLayout.PREFERRED_SIZE,
GroupLayout.DEFAULT_SIZE, GroupLayout.PREFERRED_SIZE)

.addComponent(txt_fld_5,      GroupLayout.DEFAULT_SIZE,      193,
Short.MAX_VALUE)

```

```

        .addComponent(txt_fld_4,      GroupLayout.DEFAULT_SIZE,      193,
Short.MAX_VALUE)

        .addComponent(txt_fld_3,      GroupLayout.DEFAULT_SIZE,      193,
Short.MAX_VALUE)

        .addComponent(txt_fld_2,      GroupLayout.DEFAULT_SIZE,      193,
Short.MAX_VALUE)

        .addComponent(txt_fld_1,      GroupLayout.DEFAULT_SIZE,      193,
Short.MAX_VALUE)

        .addComponent(txt_fld,        GroupLayout.DEFAULT_SIZE,      193,
Short.MAX_VALUE)

        .addComponent(txt_fld_7,      GroupLayout.DEFAULT_SIZE,      193,
Short.MAX_VALUE)

        .addComponent(txt_fld_8,      GroupLayout.DEFAULT_SIZE,      193,
Short.MAX_VALUE)

        .addComponent(txt_fld_9,      GroupLayout.DEFAULT_SIZE,      193,
Short.MAX_VALUE)

        .addComponent(txt_fld_6,      GroupLayout.DEFAULT_SIZE,      193,
Short.MAX_VALUE)))
        .addContainerGap(124,
Short.MAX_VALUE)

        .addGroup(gl_contPane.createSequentialGroup()
        .addContainerGap()
        .addComponent(lblContactNo)
        .addContainerGap(356,
Short.MAX_VALUE))

        .addGroup(gl_contPane.createSequentialGroup()
        .addContainerGap()
        .addComponent(lblCountry,
GroupLayout.DEFAULT_SIZE, 64, Short.MAX_VALUE)
        .addGap(350))

        .addGroup(gl_contPane.createSequentialGroup()
        .addContainerGap()
        .addComponent(lblCity,
GroupLayout.PREFERRED_SIZE, 31, GroupLayout.PREFERRED_SIZE)
        .addContainerGap(383,
Short.MAX_VALUE))

        .addGroup(gl_contPane.createSequentialGroup()
        .addGap(128)
        .addComponent(btnAddAccant,
GroupLayout.PREFERRED_SIZE, 145, GroupLayout.PREFERRED_SIZE)

```

```

        .addGap(18)
        .addComponent(btnBack)
        .addContainerGap(44,
Short.MAX_VALUE))

        .addGroup(gl_contPane.createSequentialGroup()
            .addContainerGap()
            .addComponent(lblState,
GridLayout.PREFERRED_SIZE, 50, GridLayout.PREFERRED_SIZE)
            .addContainerGap(364,
Short.MAX_VALUE))
        );
        gl_contPane.setVerticalGroup(

        gl_contPane.createParallelGroup(Alignment.LEADING)

        .addGroup(gl_contPane.createSequentialGroup()
            .addComponent(lblAddAccant)
            .addGap(18)

        .addGroup(gl_contPane.createParallelGroup(Alignment.BASELINE
)
            .addComponent(lblName)
            .addComponent(txt_fld,
GridLayout.PREFERRED_SIZE,
GridLayout.DEFAULT_SIZE,
GridLayout.PREFERRED_SIZE))

        .addPreferredGap(ComponentPlacement.UNRELATED)

        .addGroup(gl_contPane.createParallelGroup(Alignment.BASELINE
)
            .addComponent(lblEmail)
            .addComponent(txt_fld_1,
GridLayout.PREFERRED_SIZE,
GridLayout.DEFAULT_SIZE,
GridLayout.PREFERRED_SIZE))

        .addPreferredGap(ComponentPlacement.UNRELATED)

        .addGroup(gl_contPane.createParallelGroup(Alignment.BASELINE
)
            .addComponent(lblCourse)
            .addComponent(txt_fld_2,
GridLayout.PREFERRED_SIZE,
GridLayout.DEFAULT_SIZE,
GridLayout.PREFERRED_SIZE))

        .addPreferredGap(ComponentPlacement.UNRELATED)

        .addGroup(gl_contPane.createParallelGroup(Alignment.BASELINE
)
            .addComponent(lblFee)
            .addComponent(txt_fld_3,
GridLayout.PREFERRED_SIZE,
GridLayout.DEFAULT_SIZE,
GridLayout.PREFERRED_SIZE))

```

```

        .addPreferredGap(ComponentPlacement.UNRELATED)

        .addGroup(gl_contPane.createParallelGroup(Alignment.BASELINE
)
                                .addComponent(lblPaid)
                                .addComponent(txt_fld_4,
GroupLayout.PREFERRED_SIZE, GroupLayout.DEFAULT_SIZE,
GroupLayout.PREFERRED_SIZE))

        .addPreferredGap(ComponentPlacement.UNRELATED)

        .addGroup(gl_contPane.createParallelGroup(Alignment.BASELINE
)
                                .addComponent(lblDue)
                                .addComponent(txt_fld_5,
GroupLayout.PREFERRED_SIZE, GroupLayout.DEFAULT_SIZE,
GroupLayout.PREFERRED_SIZE))

        .addPreferredGap(ComponentPlacement.UNRELATED)

        .addGroup(gl_contPane.createParallelGroup(Alignment.BASELINE
)
                                .addComponent(lblAddress)
                                .addComponent(textArea,
GroupLayout.PREFERRED_SIZE, 49, GroupLayout.PREFERRED_SIZE))

        .addPreferredGap(ComponentPlacement.RELATED,
GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

        .addGroup(gl_contPane.createParallelGroup(Alignment.TRAILING
)

        .addGroup(gl_contPane.createSequentialGroup()
                                .addComponent(lblCity)
                                .addGap(18))

        .addGroup(gl_contPane.createSequentialGroup()
                                .addComponent(txt_fld_6,
GroupLayout.PREFERRED_SIZE, GroupLayout.DEFAULT_SIZE,
GroupLayout.PREFERRED_SIZE)

        .addPreferredGap(ComponentPlacement.UNRELATED))

        .addGroup(gl_contPane.createParallelGroup(Alignment.BASELINE
)
                                .addComponent(lblState)
                                .addComponent(txt_fld_7,
GroupLayout.PREFERRED_SIZE, GroupLayout.DEFAULT_SIZE,
GroupLayout.PREFERRED_SIZE))

        .addPreferredGap(ComponentPlacement.UNRELATED)

```

```

        .addGroup(gl_contPane.createParallelGroup(Alignment.BASELINE
    )
                                .addComponent(lblCountry)
                                .addComponent(txt_fld_8,
GroupLayout.PREFERRED_SIZE,          GroupLayout.DEFAULT_SIZE,
GroupLayout.PREFERRED_SIZE))

        .addPreferredGap(ComponentPlacement.UNRELATED)

        .addGroup(gl_contPane.createParallelGroup(Alignment.LEADING)
                                .addComponent(lblContactNo)
                                .addComponent(txt_fld_9,
GroupLayout.PREFERRED_SIZE,          GroupLayout.DEFAULT_SIZE,
GroupLayout.PREFERRED_SIZE))
                                .addGap(7)

        .addGroup(gl_contPane.createParallelGroup(Alignment.LEADING)
                                .addComponent(btnAddAccant,
GroupLayout.PREFERRED_SIZE, 38, GroupLayout.PREFERRED_SIZE)
                                .addComponent(btnBack))
                                .addContainerGap())
    );
    contPane.setLayout(gl_contPane);
}
}

```

ДОДАТОК Д
Слайди презентації

Національний Університет «Запорізька політехніка»

кафедра Програмних засобів

Програмна реалізація автоматизованої системи моніторингу для деканату

Виконав: Ржанніков К.В. (КНТ-218сп)

Керівник: Гофман Є.О., к.т.н., доцент

Рецензент: Скрупський С.Ю., к.т.н., доцент

Рисунок Д.1 – Слайд 1

Об'єкт, предмет та мета роботи

Об'єкт дослідження – процес автоматизації моніторингу успішності здобувачів освіти.

Предмет дослідження – методи автоматизації моніторингу успішності здобувачів освіти.

Метою роботи є програмна реалізація автоматизованої системи моніторингу для деканату.

2

Рисунок Д.2 – Слайд 2

Обґрунтування актуальності розробки

Застосунок	Недоліки	Переваги
Використання незалежної розробки	– надмірна комерціалізація; – складність налаштування під особливості ЗВО; – низька ергономічність дизайну	– якісна документація; – сервісна підтримка; – широкий набір функціональних можливостей
Власна розробка ЗВО	– нижчий рівень стійкості; – залежність від супроводу розробників; – неякісна документованість	– більша відповідність особливостям конкретного ЗВО; – більш гнучке та оперативне оновлення на основі власних нормативних документів; – зручність для використання

3

Рисунок Д.3 – Слайд 3

Порівняння існуючих аналогів

Критерій порівняння	Політек-Софт	FossDoc	Moodle	Система Деканат
Спеціалізованість під специфіку ЗВО	Високий рівень	Низький рівень	Високий рівень	Високий рівень
Форма застосунку	Веб-застосунок	Десктопний застосунок	Веб-застосунок	Високий рівень
Складність підтримки	Низький рівень	Високий рівень	Низький рівень	Низький рівень
Рівень підтримки процесів документообігу	Високий рівень	Високий рівень	Низький рівень	Високий рівень

4

Рисунок Д.4 – Слайд 4

Функціональні вимоги до розробки

Функціональність системи має забезпечуватися за допомогою:

- наявність та редагування списку студентів, що вчаться на факультеті;
- наявність та редагування інформації про форму навчання;
- можливість розрахунку оцінок студентів за навчальний період;
- можливість розрахунку середнього балу студентів згідно з оцінками;
- наявність та оновлення списку студентів, що отримують стипендію;
- наявність та редагування списку студентів, що отримують освіту екстерном;
- наявність та оновлювання списку іноземних студентів;
- наявність та редагування списку груп факультету та загальна їх кількість;
- оновлення списку студентів, що навчаються на пільговій основі;
- редагування розклад занять для студентів;
- редагування розклад занять для викладачів факультету;
- можливість видачі довідок;
- система повинна здійснювати збір інформації про студентів та викладачів;
- можливість виконувати пошук потрібної інформації у системі.

5

Рисунок Д.5 – Слайд 5

Обґрунтування вибору технічних засобів розробки

	Java (сервлети)	C# (.Net)	Python	PHP
Швидкість роботи застосунку	++	+	±	+
Спільнота та підтримка	+	+	+	++
Захист даних	++	++	+	+
Багатопоточність	++	±	±	±
Інтерпретованість застосунку	+	±	+	+

6

Рисунок Д.6 – Слайд 6

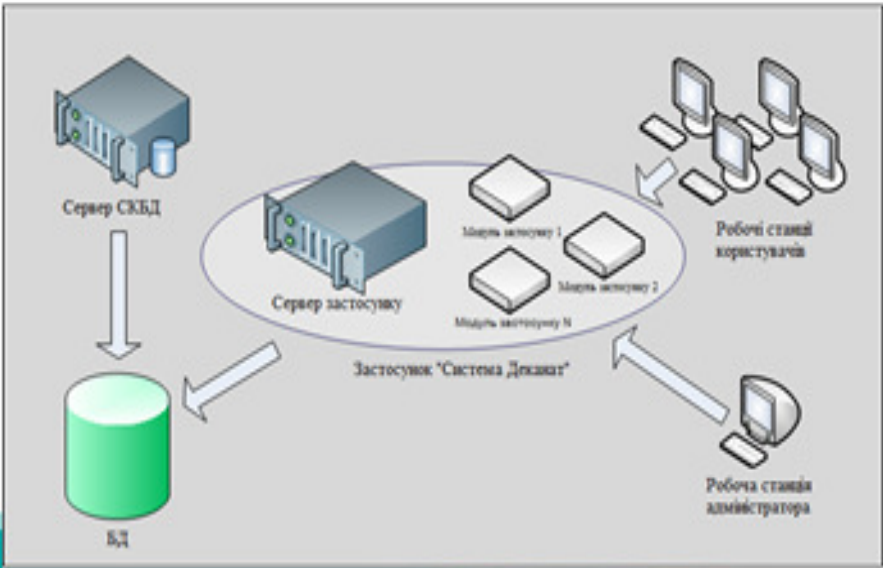
Обґрунтування вибору технічних засобів розробки

	IntelliJ IDEA	Eclipse	NetBeans
Робота із сервлетами	++	±	+
Покращені механізми роботи із СКБД	++	±	+
Механізми роботи із Git	++	++	++
Кроссфлатформність	++	++	-
Плагіни та розширення	+	++	±
Вартість використання	±	++	++

7

Рисунок Д.7 – Слайд 7

Архітектура застосунку «Система Деканат»



8

Рисунок Д.8 – Слайд 8

Діаграма класів системи



9

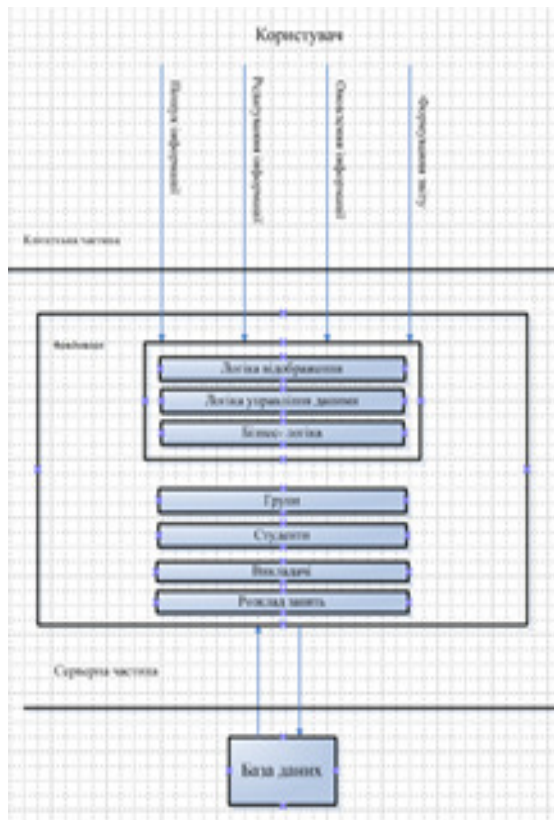
Рисунок Д.9 – Слайд 9

Загальна діаграма прецедентів



10

Рисунок Д.10 – Слайд 10



Архітектура системи на основі використання фреймворку

11

Рисунок Д.11 – Слайд 11

Приклади сторінок користувачів

Система Деканат

Вітаємо у системі!
Введіть особисті дані для початку роботи із системою

Логін:

Пароль:

[Забудувати пароль?](#)

Система деканат

Користувач: **Іванов Іван Іванович**

Посада	ПІБ	Група	Форми
Здобувач	Петров П.П.	ЮНТ-123	Диплом
Здобувач	Петров І.М.	ІФ-321	Зачислює

Користувач: **Іванов Іван Іванович**

ІМ	ПІБ	Вступивши в систему	Виходивши з системи	Статус
Іванов І.	Іванов І.	2019-09-01	2019-09-01	активний
Петров П.	Петров П.	2019-09-01	2019-09-01	активний
Сидоров С.	Сидоров С.	2019-09-01	2019-09-01	активний
Козлов К.	Козлов К.	2019-09-01	2019-09-01	активний
Васильєв В.	Васильєв В.	2019-09-01	2019-09-01	активний
Смирнов С.	Смирнов С.	2019-09-01	2019-09-01	активний
Михайлов М.	Михайлов М.	2019-09-01	2019-09-01	активний
Іванов І.	Іванов І.	2019-09-01	2019-09-01	активний
Петров П.	Петров П.	2019-09-01	2019-09-01	активний
Сидоров С.	Сидоров С.	2019-09-01	2019-09-01	активний

Звіт:

- Заборгованості 2020
- Заборгованості 2019
- Заборгованості 2018
- Успішність навчання

12

Рисунок Д.12 – Слайд 12

Висновки

Розроблений застосунок виконує наступні функції:

- створювати індивідуальний профіль, що містить в собі певну інформацію про користувача;
- мати доступ до інформації, що міститься у базі даних системи;
- знаходити необхідну інформацію;
- оформлювати звіти з, обраної користувачем, інформації;
- редагувати, а саме додавати, змінювати та видаляти інформацію;
- виводити створені або збережені звіти на друк.

В результаті виконання завдання було отримано систему, що відповідає всім вимогам представленим в технічному завданні, зручна для використання та виконує всі необхідні функції.

Усі завдання випускної дипломної роботи бакалавра повністю виконано.

Рисунок Д.13 – Слайд 13