МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

ЦЕНТР ЗАОЧНОЇ, ДИСТАНЦІЙНОЇ ТА ВЕЧІРНЬОЇ ФОРМ НАВЧАННЯ

КАФЕДРА КОМП'ЮТЕРНИХ НАУК

КВАЛІФІКАЦІЙНА МАГІСТЕРСЬКА РОБОТА

на тему:

«Інформаційна система підтримки роботи атестаційної комісії: модуль секретаря»

Завідувач випускаючої кафедри — **Довбиш А.С.**

Керівник роботи — **Великодний Д.В.**

Студента групи ІН.мз – 92с — **Ващенко С.М.**

СУМИ 2021

Сумський державний університет
(назва вузу)

Факультет *ЦЗДВФН* Кафедра *Комп'ютерних наук*
Спеціальність *«Інформатика»*

Затверджую:
зав.кафедрою _____
_____
"_____" _____ 20___р.

ЗАВДАННЯ
НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ) СТУДЕНТОВІ

Ващенко Світлані Михайлівні
(прізвище, ім'я, по батькові)

1. Тема проекту (роботи) *Інформаційна система підтримки роботи атестаційної комісії: модуль секретаря*

затверджую наказом по інституту від "____" _____ 20 ___ р. № _____

2. Термін здачі студентом закінченого проекту (роботи) _____

3. Вхідні данні до проекту (роботи) *нормативна документація Сумського державного університету щодо організації процесу атестації здобувачів вищої освіти*

4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити) *Аналіз процесу проведення атестації. Моделювання інформаційної системи: загальна структура та концепція інформаційної системи; модель роботи підсистеми роботи секретаря; Інформаційне та програмне забезпечення системи: інформаційна підтримка роботи модуля секретаря; розробка інтерфейсу користувача; програмна реалізація. Висновки. Додатки*

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

6. Консультанти до проекту (роботи), із значенням розділів проекту, що стосується їх

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання _____

Керівник _____
(підпис)

Завдання прийняв до виконання _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів дипломного проекту (роботи)	Термін виконання проекту (роботи)	Примітка
1.	*Визначення актуальності теми роботи, мети роботи, об'єкту та предмету дослідження*	*15.09.2020*	*Викон.*
2.	*Аналіз процесу проведення атестації на секції*	*10.10.2020*	*Викон.*
3.	*Створення загальної концепції системи та опис її структури*	*15.10.2020*	*Викон.*
4.	*Моделювання модуля секретаря, визначення функіональних вимог та принципу роботи*	*01.11.2020*	*Викон.*
5.	*Проектування бази даних та фізична реалізація створеної моделі*	*10.11.2020*	*Викон.*
6.	*Проектування інтерфейсу користувача*	*20.11.2020*	*Викон.*
7.	*Програмна реалізація модуля «Секретар»*	*25.12.2020*	*Викон.*
8.	*Оформлення звітної документації*	*14.01.2021*	*Викон.*

Студент – дипломник Ващенко С.М.
(підпис)

Керівник проекту Великодний Д.В.
(підпис)

РЕФЕРАТ

Записка: 85 стор., 32 рис., 0 табл., 2 додатки, 36 джерел.

Об’єкт дослідження — процеси, які забезпечують проведення атестаційних процедур.

Мета роботи — розробка загальної концепції інформаційної системи по захисту кваліфікаційних робіт студентів освітніх програм «Інформаційні технології проектування» всіх рівнів підготовки; реалізація модуля підтримки роботи секретаря.

Методи дослідження — системний аналіз для моделювання інформаціної системи.

Результати — розроблено загальну концепцію побудови та роботи інформаційної системи підтримки роботи комісії по захисту кваліфікаційних робіт здобувачів вищої освіти освітньої програми «Інформаційні технології проектування». Отримані результати представлено у формі моделі у стандарті BPMN. Визначено структуру й системи. Змодельовано роботу підсистеми роботи секретаря комісії. Розроблено базу даних, яка працює під керівництвом СУБД MySQL Server. Мовою програмування C# з використанням API Windows Forms виконано програмну реалізацію модуля «Секретар».

АТЕСТАЦІЯ, КВАЛІФІКАЦІЙНА РОБОТА, ІНФОРМАЦІЙНА СИСТЕМА «АТЕСТАЦІЯ», КОНЦЕПТУАЛЬНА МОДЕЛЬ, МОДУЛЬ СТУДЕНТА, МОДУЛЬ КЕРІВНИКА, МОДУЛЬ СЕКРЕТАРЯ, БАЗА ДАНИХ, MYSQL SERVER, C#, MYSQL CONNECTOR, WINDOWS FORMS

ЗМІСТ

ВСТУП

Згідно стандарту спеціальності «122 Комп'ютерні науки» [1] для освітніх програм «Інформаційні технології проектування» всіх рівнів [2, 3], за якими проводиться підготовка студентів в Сумському державному університеті, визначено, що атестація здобувачів виконується за результатами виконання кваліфікаційної роботи та її захисту. Звісно, цей процес супроводжується оформленням великої кількості документації: від підготовки наказу про затвердження тем кваліфікаційних робіт, до формування звіту голови про роботу відповідної екзаменаційної комісії. В переважній більшості вся робота покладається на секретаря комісії. При цьому забезпечення проведення процесу атестацій супроводжується виконанням великої кількості рутинних операцій, на які витрачається багато часу та сил.

Враховуючи рівень інформатизації сучасного суспільства, неможливо уявити реалізацію процесів обробки інформації на рівні ручної праці. Робота екзаменаційної комісії не є винятком. Безумовно при підготовці та проведенні атестаційних процедур використовуються сучасні інформаційні технології. Але наразі різні етапи цього процесу супроводжуються використанням розрізненого програмного забезпечення. Це спричиняє низку труднощів. Тому можна стверджувати, що обрана тематика є актуальною.

Об'єктом дослідження є процеси, які забезпечують проведення атестаційних процедур.

Предмет дослідження – інформаційні технології, які використовуються при підготовці та проведенні заходів атестації здобувачів вищої освіти.

Метою роботи є розробка загальної концепції інформаційної системи по захисту кваліфікаційних робіт студентів освітніх програм «Інформаційні технології проектування» (ІТП) всіх рівнів підготовки; реалізація модуля підтримки роботи секретаря.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

Успішна діяльність будь-якого закладу в сучасних умовах фактично є неможливою без впровадження і активного використання інформаційних технологій. Адже постійно зростаючі обсяги інформації вимагають від працівників значних зусиль для обробки.

Беззаперечно, в Сумському державному університеті рівень інформатизації підтримки робочих процесів діяльності різних підрозділів, в тому числі і кафедр, є значним. Для забезпечення роботи підрозділів різного рівня використовується загальна автоматизована система «АСУ «Університет»» [4]. На сьогодні практично всі процеси забезпечення діяльності кафедри автоматизовані та підтримуються різними модулями системи «Університет» - від обліку навчально-методичної роботи, до розподілу та обліку навчального навантаження викладача. Значно спрощують роботу і особистий кабінет викладача, і механізм електроних відомостей.

Проте один їх фінальних етапів по підготовці фахівців, а саме їх атестація, на даний момент не автоматизована, і відповідного модуля підтримки проведення атестаційних заходів в АСУ «Університет» на даний час немає.

Пошук аналогів в відкритих джерелах інформації не приніс результату. Тому і було обрану дану тему кваліфікаційної роботи.

Перш ніж перейти до розробки самої системи, потрібно детально проаналізувати теперішній стан питання і використання інформаційних технологій. Це дасть змогу отримати уявлення про всі дії, які виконуються учасниками процесу, які інформація потрібна, а також виявити моменти, які вимагають доопрацювання.

1.1 Аналіз процесу проведення атестації.

Процес атестації здобувачів вищої освіти супроводжується обробкою великої кількості інформації, володіти якою, в першу чергу, має секретар комісії, оскільки сам він регламентує та виконує більшість етапів. Роботу має бути організовано так, щоб було зручно зберігати і отримувати потрібну інформацію в потрібний момент часу. Це визначає той факт, що треба якісно структурувати інформацію і забезпечити виконання всіх процесів в єдиному інформаційному просторі.

Весь процес підготовки до атестації і безпосередньо роботи екзаменаційної комісії регламентується відповідними нормативними документами СумДУ [5] та провадиться в суворій відповідності їм.

Найперше, з чим стикається секретар, це розподіл студентів за викладачами – керівниками дипломних робіт. Головне вчасно і точно зібрати інформацію від студентів. Студентам-бакалаврам пропонується ще під час проходження виробничої практики після весняної сесії 3 курсу визначитися з можливим напрямком та керівником майбутньої дипломної роботи. Студенти-магістри, зазвичай, закріпляються за викладачем, який керував кваліфікаційною роботою бакалавра.

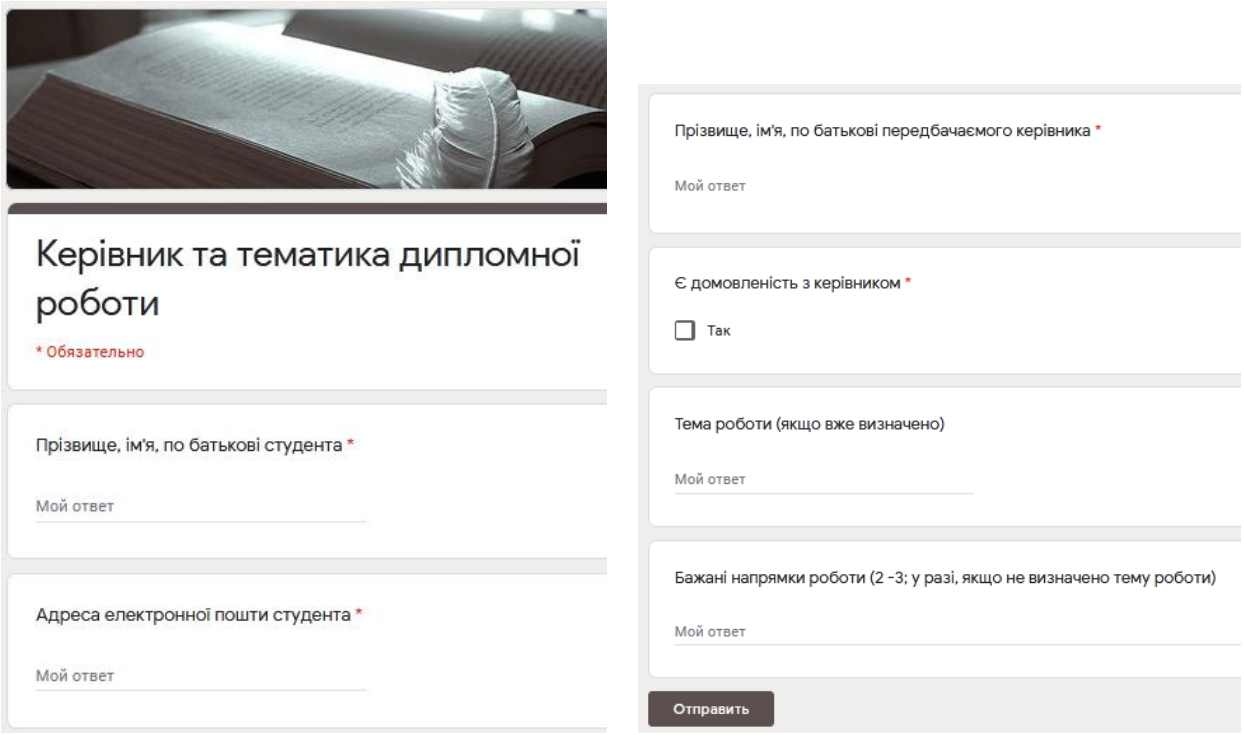


Рисунок 1.1 – Форма опитування студентів

На даний момент етап опитування студентів виконується з використанням створеної Google-форми (рис.1.1), в якій студент визначає:

- прізвище, ім’я та по батькові власне;
- прізвище, ім’я та по батькові передбачаємого керівника;
- попередня тема чи бажані напрямки роботи.

Опрацьовуючи отримані від студентів дані, секретар створює електронну таблицю, до якої надає доступ викладачам секції. Структуру таблиці наведено на рис.1.2.

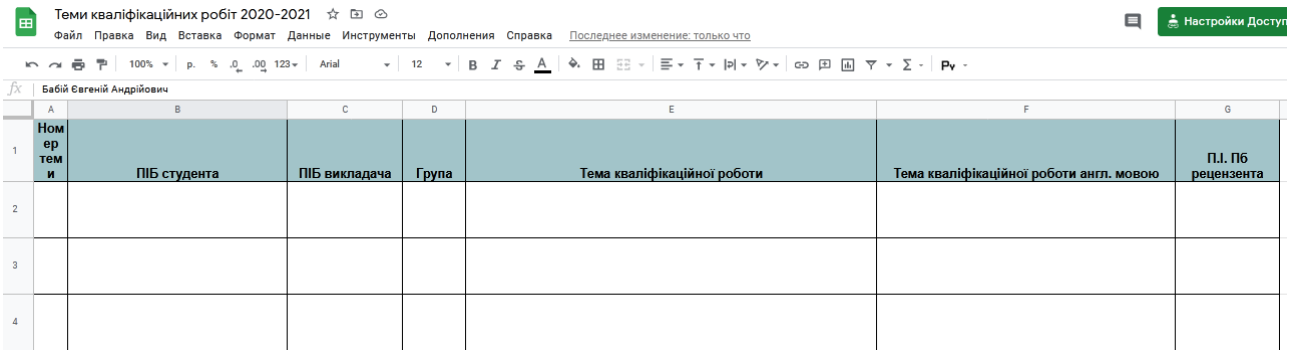
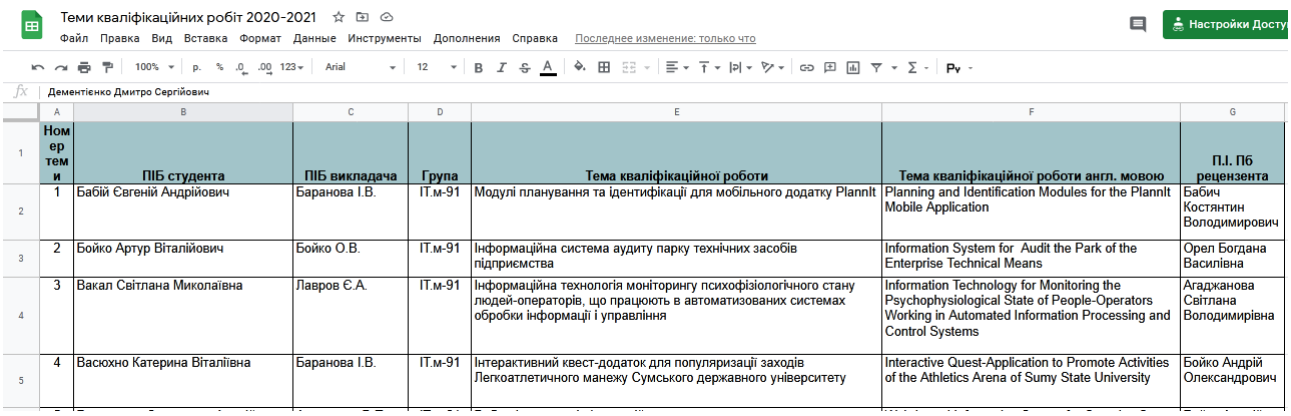


Рисунок 1.2 – Структура таблиці в робочому документі

В таблицю заноситься список студентів і виконується їх закріплення за керівниками. Протягом певного часу студент та керівник визначаються з напрямом роботи та формулюють відповідну тему роботи. Керівник, маючи доступ до цієї таблиці, вносить відповідні зміни. Звісно, такий підхід не є досконалим, оскільки будь-який керівник має доступ до всієї таблиці і є ймовірність виникнення ситуації випадкового помилкового редагування вже існуючої інформації. На виявлення та виправлення цієї ситуації може бути втрачено час.

Обов’язковим етапом атестації здобувачів вищої освіти є етап рецензування виконаної студентом роботи третьою особою. Це етап регламентується відповідним наказом [6]. Згідно цього наказу рецензентами бакалаврських робіт є викладачі кафедри. До рецензування робіт студентів освітнього рівня «магістр» залучаються фахівці відповідних до спеціальності підприємств. Після того, як остаточно визначено закріплення за керівниками робіт та сформульовано теми, секретар за погодженням з завідувачем секції формує список зовнішніх/внутрішніх рецензентів (в залежності від освітньо-кваліфікаційного рівня) та розподіляє роботи серед них. Таблиця приймає вигляд, як на рис.1.3. Секретар оформлює всю необхідну документацію згідно наказу [6].


Теми кваліфікаційних робіт 2020-2021

Файл Правка Вид Вставка Формат Данные Инструменты Дополнения Справка Последнее изменение: только что

Настройки Досту

Дементієнко Дмитро Сергійович

	A	B	C	D	E	F	G
1	Номер теми	ПІБ студента	ПІБ викладача	Група	Тема кваліфікаційної роботи	Тема кваліфікаційної роботи англ. мовою	П.І. ПІб рецензента
2	1	Бабій Євгеній Андрійович	Баранова І.В.	ІТ.м-91	Модулі планування та ідентифікації для мобільного додатку PlannIt	Planning and Identification Modules for the PlannIt Mobile Application	Бабич Костянтин Володимирович
3	2	Бойко Артур Віталійович	Бойко О.В.	ІТ.м-91	Інформаційна система аудиту парку технічних засобів підприємства	Information System for Audit the Park of the Enterprise Technical Means	Орел Богдана Василівна
4	3	Вакал Світлана Миколаївна	Лавров Є.А.	ІТ.м-91	Інформаційна технологія моніторингу психофізіологічного стану людей-операторів, що працюють в автоматизованих системах обробки інформації і управління	Information Technology for Monitoring the Psychophysiological State of People-Operators Working in Automated Information Processing and Control Systems	Агаджанова Світлана Володимирівна
5	4	Васюхно Катерина Віталіївна	Баранова І.В.	ІТ.м-91	Інтерактивний квест-додаток для популяризації заходів Легкоатлетичного манежу Сумського державного університету	Interactive Quest-Application to Promote Activities of the Athletics Arena of Sumy State University	Бойко Андрій Олександрович

Рисунок 1.3 – Зібрана інформація

Наступним етапом секретар формує проект наказу по університету про затвердження тем і керівників випускних кваліфікаційних робіт. Проект формується з використанням текстового редактора Microsoft Word. У створений за спеціальним шаблоном документ секретар копіює з електронної таблиці необхідну інформацію і виконує необхідне форматування (рис.1.4).

Міністерство освіти і науки України

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

НАКАЗ

від ___ _____ 20__ р. м. Суми №_____

Про затвердження тем і керівників кваліфікаційних робіт

З метою організації підсумкової атестації здобувачів вищої освіти у відповідності до вимог, визначених Положенням про організацію освітнього процесу в СумДУ

НАКАЗУЮ:

1. Затвердити теми та призначити керівників кваліфікаційних робіт здобувачів вищої освіти у відповідності до нижченаведеного:

№ з/п	Прізвище, ім'я, по батькові здобувача	Тема кваліфікаційної (дипломної) роботи (проекту) 2)	Посада, вчене звання та науковий ступінь, прізвище й ініціали керівника 3)
1	2	3	4
122 «Комп'ютерні науки», рівень підготовки «магістр», форма навчання денна			
Академічна група ІТ.м-91			
1	Бабій Євгеній Андрійович	Модулі планування та ідентифікації для мобільного додатку PlannIt / Planning and Identification Modules for the PlannIt Mobile Application	доцент кафедри комп'ютерних наук, доцент, к.т.н. Баранова І.В.

Рисунок 1.4 – Проект наказу про затвердження тем і керівників кваліфікаційних робіт

Додатково секретар формує зведення, скільки дипломників має кожен з викладачів і передає інформацію особі, відповідальній за навантаження.

Це ж саме стосується і рецензентів – для обліку навантаження і оплати роботи визначається, скільки робіт буде направлено кожному з рецензентів.

Також на цьому етапі секретар комісії формує та подає на узгодження графік роботи екзаменаційної комісії у строки, визначені діючим графіком навчального процесу університету.

На наступному етапі секретар приймає документи від студентів:

- готову роздруковану, зшиту та підписану роботу;
- відгук керівника з визначеним балом за визначеним шаблоном (рис. 1.5);
- рецензія на встановленому бланку з вказанням оцінки роботи (рис.1.6);
- копії документів, які дозволяють оцінити рівень оприлюднення результатів роботи. Це можуть бути ксерокопії публікацій, авторських свідоцтв тощо.

ВІДГУК КЕРІВНИКА

на кваліфікаційну роботу магістра
студента групи ІТ.м-91

Нечепорука Олександра Андрійовича

на тему
Інструментарій контейнеризації програмних додатків

Повнота розкриття поставленої задачі *повністю*
(не повністю – 1 балів, частково – 2 бали, повністю – 3 бали)

Технічний рівень виконання роботи *високий*
(достатній – 0 балів, середній – 2 бали, вище середнього – 3 бали, високий – 4 бали)

Самостійність, ініціативність *високий*
(достатній – 0 балів, середній – 1 бал, високий – 2 бали)

Уміння узагальнювати результати існуючих робіт з тематики, що розглядаються
високий
(достатній – 0 балів, середній – 1 бал, високий – 2 бали)

Знання з загальнотехнічних і спеціальних дисциплін, уміння самостійно використовувати їх при вирішенні поставлених завдань
високий
(достатній – 0 балів, середній – 2 бали, вище середнього – 3 бали, високий – 4 бали)

Можливість впровадження матеріалів роботи
може бути впроваджено без доопрацювання
(може бути впроваджено зі значним доопрацюванням – 0 балів, може бути впроваджено з незначним доопрацюванням – 1 бал, може бути впроваджено без доопрацювання – 2 бали)

Систематичність роботи, своєчасність виконання
високий
(достатній – 0 балів, середній – 2 бали, вище середнього – 3 бали, високий – 4 бали)

Зауваження *відсутні*
(суттєві – 2 бали, несуттєві –1 бал, відсутні – 0 балів)

Позитивні сторони *значні переваги*
(незначні переваги – 2 бали, суттєві переваги – 3 бали, значні переваги – 4 бали)

Загальна оцінка ***відмінно (25 б.)***

(сума всіх балів – достатньо (13-14 балів), задовільно (15-17 балів), добре (18-20 балів), дуже добре (21-23 бали), відмінно (24-25 балів))

Рисунок 1.5 – Зразок відгука керівника

РЕЦЕНЗІЯ

на кваліфікаційну роботу бакалавра

студента групи *ІТ-61*

Бєлка Ярославі Сергіївні
(ПІБ)

на тему

Інформаційна система взаємодії з клієнтами агентства нерухомості

Робота виконана в обсязі: *85* сторінок. Ілюстраційний матеріал: _____ слайдів.

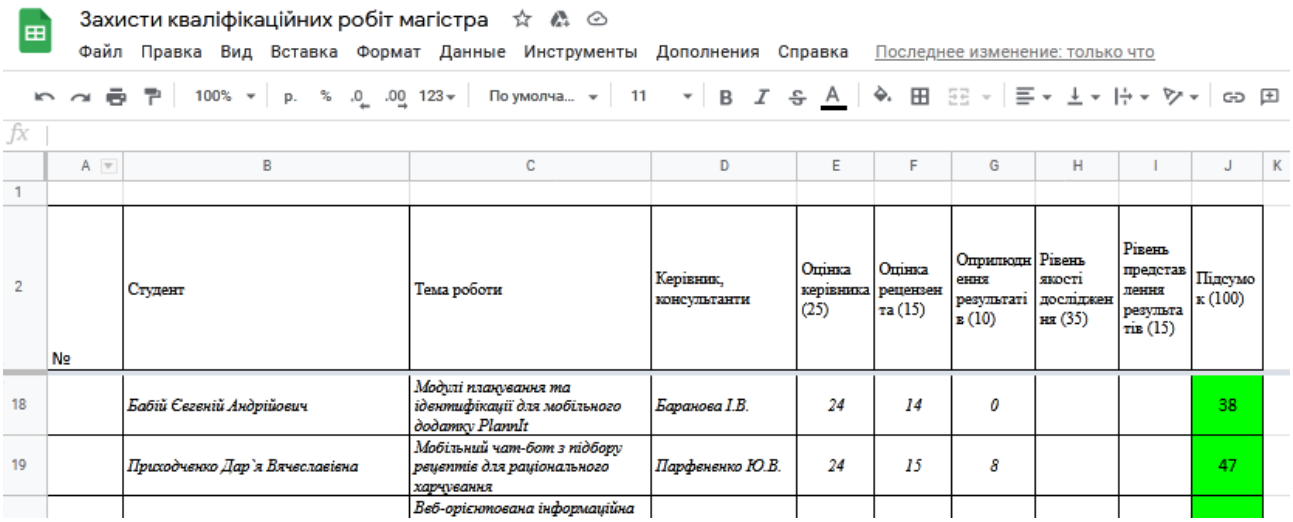
Критерій та шкала оцінювання критерію дипломного проекту	**Оцінка, бали**
Повнота розкриття теми *не повністю – 1 балів, частково –2 бал, повністю 3 бали*	3
Технічний рівень роботи *достатній – 1 бал, середній – 2 бали, високий – 3 бали*	3
Якість оформлення роботи *невисока – 1 балів, середня – 2 висока – 3 бали*	3
Позитивні сторони роботи (зазначити) *незначні переваги – 2 бали значні переваги – 3 бали*	3
Можливість впровадження матеріалів проекту *не може бути впроваджено – 0 балів, може бути впроваджено – 2 бали, впроваджено – 3 бали при наявності акту впровадження*	3
Загальна оцінка дипломного проекту: *задовільно (7-9 балів), добре (10-13 балів), відмінно (14-15 балів)* *Зауваження*_____ _____ _____	15

Рисунок 1.6 – Приклад оформлення рецензії

Паралельно з цим, студентам секретар відкриває доступ до Google-таблиці, де вони записуються на конкретну дату захисту згідно визначеного розкладу.

Для підготовки безпосередньо до засідань екзаменаційної комісії секретар узагальнює всю отриману інформацію. Для цього формується електронна таблиця, в яку вноситься вся зібрана інформація. На рис. 1.7 наведено приклад такої сторінки.

У випадку звичайного очного проведення захистів, кожному члену комісії друкується копія таблиці. У випадку он-лайн захисту, члени комісії отримують доступ до самої таблиці, щоб могли ознайомитися з потрібною інформацією.



Захисти кваліфікаційних робіт магістра

Файл Правка Вид Вставка Формат Данные Инструменты Дополнения Справка Последнее изменение: только что

№	Студент	Тема роботи	Керівник, консультанти	Оцінка керівника (25)	Оцінка рецензента (15)	Оприлюднення результатів (10)	Рівень якості дослідження (35)	Рівень представлення результатів (15)	Підсумок (100)
18	*Бабій Євгеній Андрійович*	*Модулі планування та ідентифікації для мобільного додатку PlannIt*	*Баранова І.В.*	*24*	*14*	*0*			38
19	*Приходченко Дар'я Вячеславівна*	*Мобільний чат-бот з підбору рецептів для раціонального харчування*	*Парфененко Ю.В.*	*24*	*15*	*8*			47
		Веб-орієнтована інформаційна							

Рисунок 1.7 – Підготовлена таблиця для інформаційного забезпечення проведення засідання екзаменаційної комісії

Структура таблиці максимально наближена до структури тієї таблиці, яка формується в протоколі засідання.

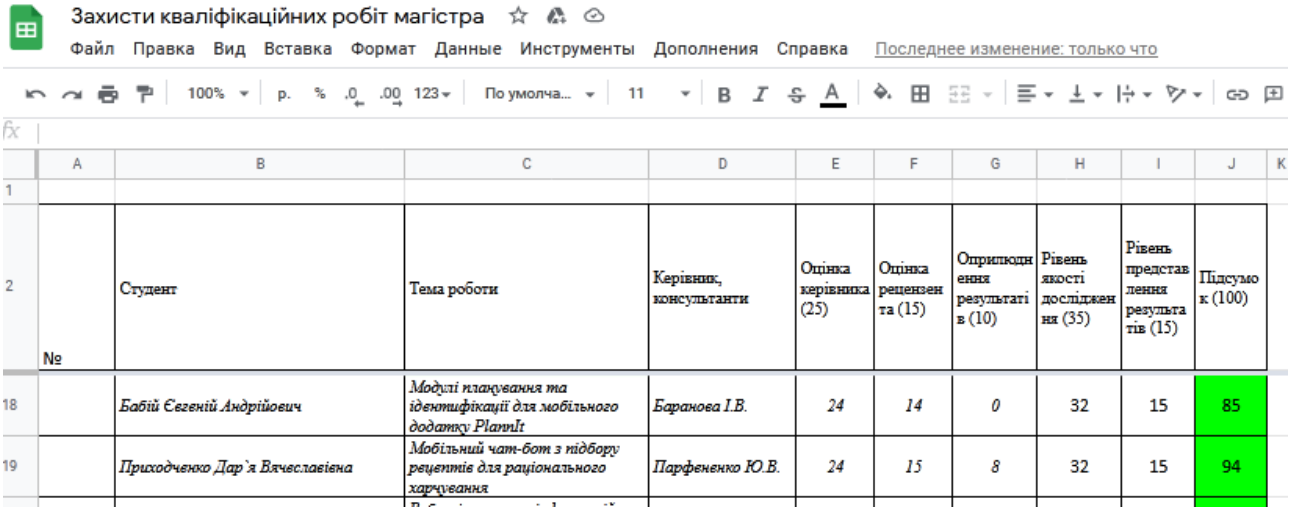
Також секретарю потрібно отримати з деканату інформацію про студентів, які є претендентами на отримання диплому з відзнакою.

Впродовж терміну роботи екзаменаційної комісії секретар веде засідання, на яких:

- доводить до відома голови та членів екзаменаційної комісії інформацію, що стосується конкретної роботи студента, представляючи її до захисту;
- слідкує за регламентом доповідей здобувачів;
- веде протокол засідань екзаменаційної комісії, фіксуючи хто ставив питання, та характери відповідей.

Під час обговорення члени комісії приймають по кожній роботі рішення щодо оцінки за критеріями «Рівень якості дослідження» та «Рівень представлення результатів». Секретар вносить інформацію в таблицю і таким чином формується загальна оцінка роботи (рис 1.8, виділено зеленим кольором). Визначаються, чи підтвердили студенти свої дипломи з відзнакою. Для магістрів також формується список тих, кого комісія рекомендує до вступу

в аспірантуру.


Захисти кваліфікаційних робіт магістра

Файл Правка Вид Вставка Формат Данные Инструменты Дополнения Справка Последнее изменение: только что

№	Студент	Тема роботи	Керівник, консультанти	Оцінка керівника (25)	Оцінка рецензента (15)	Оприлюднення результатів (10)	Рівень якості дослідження (35)	Рівень представлення результатів (15)	Підсумок (100)
18	*Бабій Євгеній Андрійович*	*Модулі планування та ідентифікації для мобільного додатку PlannIt*	*Баранова І.В.*	*24*	*14*	*0*	32	15	85
19	*Приходченко Дар'я Вячеславівна*	*Мобільний чат-бот з підбору рецептів для раціонального харчування*	*Парфененко Ю.В.*	*24*	*15*	*8*	32	15	94

Рисунок 1.8 – Оцінка роботи

На основі вказаної таблиці голова комісії оголошує результати засідання.

Після закінчення засідання екзаменаційної комісії секретар формує протокол засідання екзаменаційної комісії (рис.1.9) з атестації здобувачів вищої освіти за шаблоном, визначеним в положенні [5].

Після проведення всіх засідань, визначених графіком роботи комісії, має бути складено звіт голови про роботу екзаменаційної комісії (рис. 1.10). згідно вимог положення [5]. Для цього секретар створює текстовий файл, куди вписує основні відомості про склад комісії, вказує статистику по оцінюванню робіт (рис.1.9, виділено). Також в звіті визначається, чи були захисти англійською мовою і вказується перелік здобувачів. Голові залишається лише внести свої висновки, зауваження та пропозиції.

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

ПРОТОКОЛ № *21-18* від *«21» грудня 2020* року

засідання екзаменаційної комісії № *21* з атестації здобувачів вищої освіти щодо розгляду випускної кваліфікаційної роботи (дипломного проєкту)

для здобуття освітнього ступеня *Магістр* за спеціальністю (спеціалізацією) *122 «Комп'ютерні науки» (освітньо-професійна програма «Інформаційні технології проектування»)*

інституту (факультету, центру [2]) *факультету Електроніки та інформаційних технологій*

Кваліфікація, що присвоюється *магістр з комп'ютерних наук*

Присутні:
голова комісії *Шифрін Д. М., головний конструктор з інформаційних технологій АТ "Науково-дослідний і проектно-конструкторський інститут атомного та енергетичного насособудування"*

члени комісії 1. *Шендрик В. В., завідувач секцією інформаційних технологій проектування кафедри КН* 3. *Баранова І.В., доцент кафедри КН*

2. *Лавров Є.А., професор кафедри КН* 4. *Орел Б. В., директор ТОВ «Брокодерс»*

Засідання розпочато 10 год 00 хв. Закінчено 13 год 00 хв.

№ пор.	Прізвище, ім'я, по батькові	Тема випускної кваліфікаційної роботи (дипломного проекту)	Прізвище та ініціали керівника та консультантів	Оцінка керівника	Оцінка рецензента	Характеристика відповіді на питання членів комісії [3]	Рішення екзаменаційної комісії				Підпис голови комісії
							кількість балів	оцінка за національною шкалою	присвоєння	видання дипломів	
1	2	3	4	5	6	7	8	9	10	11	12
1	*Бабій Євгеній Андрійович*	*Модулі планування та ідентифікації для мобільного додатку Plannit*	*Баранова І.В.*	*Відмінно*	*Відмінно*	*Вірна*	*85*	*Добре*	*так*	-	
2	*Приходченко Дар`я Вячеславівна*	*Мобільний чат-бот з підбору рецептів для раціонального харчування*	*Парфененко Ю.В.*	*Відмінно*	*Відмінно*	*Повна*	*94*	*Відмінно*	*так*	-	

Рисунок 1.9 – Зразок протоколу засідання екзаменаційної комісії

ЗВІТ
про роботу екзаменаційної комісії № 21
про результати підсумкової атестації здобувачів вищої освіти

Освітній ступінь магістр
Форма здобуття освіти денна
Спеціальність 122 комп'ютерні науки
Освітньо-професійна програма Інформаційні технології проектування
Інститут/факультет Факультет Електроніки та інформаційних технологій
Навчальний рік 2020/2021 Група ІТ.м-91

Екзаменаційна комісія № 21, що була затверджена наказом ректора СумДУ № 0127-І від 07.02.2020 р., у складі:

Голови екзаменаційної комісії: Шифрін Д. М., головний конструктор з інформаційних технологій АТ «Науково-дослідний і проектно-конструкторський інститут атомного та енергетичного насособудування»

Членів екзаменаційної комісії:
- Шендрик В. В., завідувач секцією інформаційних технологій проектування кафедри КН
- Лавров Є.А, професор кафедри комп'ютерних наук
- Орел Б.В, директор ТОВ «Брокодерс»
- Баранова І. В., доцент кафедри КН

Секретаря екзаменаційної комісії: Парфененко Ю.В., доцент кафедри КН

Під час проведення підсумкової атестації ЕК було розглянуто 25 робіт, з них із оцінкою:
«відмінно» - 16;
«добре» - 9;
«задовільно» - 0.

Загальні висновки:

Рисунок 1.10 – Звіт голови екзаменаційної комісії

1.2 Постановка задачі.

Як видно з аналізу, обсяги інформації, що формуються в процесі підготовки до атестаційних заходів по присвоєнню кваліфікацій випускникам спеціальності, та кількість процесів, які виконуються при цьому, свідчать про необхідність використання інформаційних технологій з метою прискорення та полегшення цих дій.

Відсутність єдиної інформаційної системи, яка б могла забезпечити це, призводить до незручностей. Зокрема, інформація про одна й ті самі показники може бути використана на різних етапах, але в тому чи іншому вигляді зберігатися в різних сховищах. Тобто, загальний процес не має спільного інформаційного простору.

Оскільки більшість всіх описаних вище процесів виконується секретарем екзаменаційної комісії, то можна вважати, що автоматизація цих дій є першочерговою задачею.

Тому в рамках виконаня даної роботи сформульовано такі задачі, які необхідно вирішити.

1. Грунтуючись на результатах проведеного аналізу, сформувати в форматі в аспекті «to be» сформувати загальну концептуальну модель інформаційної системи підтримки роботи атестаційної комісії, яка б охоплювала всесь процес вцілому:
    - визначити структуру системи, взаємозв’язки між окремими елементами, а також встановити суб’єкти її роботи;
    - в загальному описати роботу кожної підсистеми описати всі бізнес-процеси, які мають відбуватися;
    - представити архітектуру програмної реалізації інформаційної ситеми вцілому.

Загальну концептуальну модель інформаційної системи представити у формі діаграми за стандартом BPMN.

2. Більш детально розглянути роботу модуля секретаря екзаменаційної комісії (ЕК). Визначити перелік функціональних вимог щодо роботи цього модуля. Побудувати діаграму варіантів використання.
3. Для забезпечення інформаційної підтримки роботи модуля секретаря спроектувати базу даних, визначити її структуру та виконати фізичну реалізацію з використанням СУБД MySQL Server.
4. Згідно встановлених при моделюванні вимог виконати програмну реалізацію модуля підтримки роботи секретаря. Забезпечити виконання всіх функціональних вимог.

2 МОДЕЛЮВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

2.1 Загальна структура та концепція інформаційної системи.

Запорука вдалої розробки інформаційної системи – це точне розуміння процесів, які мають відбуватися в ній, та її структури. Тому важливо на першому етапі створити модель та описати загальну концепцію інформаційної системи, що має бути розроблена [7].

Проведений детальний аналіз процесу забезпечення роботи екзаменаційної комісії дозволяє виділити в системі такі дійові особи:

- здобувач;
- керівник роботи;
- рецензент роботи;
- голова та члени комісії;
- секретар.

Кожен з них приймає участь на певних етапах і виконує обробку певної інформації. А саме:

- здобувач – подає інформацію щодо керівника і тематики робіт; отримує супровідну інформацію щодо вимог до кваліфікаційної роботи, важливих дат; забезпечує проходження рецензії; виконує доповідь на захисті та відповідає на питання;
- керівник роботи – узгоджує тему дипломної роботи; оцінює роботу студента, підготувавши відгук;
- рецензент роботи – оцінює роботу студента підготувавши відповідну рецензію;
- голова та члени комісії – приймають участь у засіданні; ставлять питання студенту; оцінюють роботу студента;

- секретар – фактично інкапсулює всю обробку інформації, отриманої від попередніх акторів, і виконує менеджмент всіх вказаних вище дій; формує наказ про затвердження тем та керівників; збирає інформацію щодо оцінювання робіт студентів керівником та рецензентом; веде засідання; формує протоколи засідань; формує шаблон звіту голови комісії.

Безпосередньо роботу з інформацією, яка має зберігатися в системі, виконують лише студент, керівник та секретар. Рецензент, голова та члени комісії, фактично, відіграють роль джерел певної інформації на відповідних етапах процесу атестації. Тому вважаємо, що доцільним є виділення в структурі інформаційної системи таких підсистем:

- модуль студента;
- модуль керівника;
- модуль секретаря.

Розглянемо більш детально кожен з них. Зазначимо, що подальший опис системи буде виконуватися в аспекті «to be», тобто як має бути.

Модуль студента призначений для забезпечення зручної роботи здобувачів впродовж процесу підготовки та захисту кваліфікаційної роботи. З точки зору доступності інформації, його можна розділити на два рівні:

- загальнодоступна – інформування здобувачів про загальноприйняту інформацію щодо забезпечення процесу їх атестації: вимоги до роботи; етапи та терміни їх проведення;
- закритий рівень – доступ до функціоналу, який забезпечить студента зручним інструментарієм подачі інформації про тематику дипломної роботи на початковому етапі, а також можливість запису на конкретну дату захисту.

Щоб розмежувати доступ до функціоналу, потрібно передбачити механізм реєстрації/авторизації студента в підсистемі.

Модуль керівника повинен також забезпечувати доступ викладача до тієї ж загальнодоступної інформації, що і для студента. Після авторизації потрібно забезпечити доступ до функціоналу по введенню та редагуванню тем дипломних робіт тільки тих студентів, які закріплені за ним. Ознакою того, що тему остаточно сформульовано, є підтвердження цього факту викладачем, про що секретарю повинно відправлятися інформаційне повідомлення.

Оскільки однією з функцій обох модулів є забезпечення доступу до довідкової інформації, то реалізацію обох модулів вважаємо за доцільне реалізувати у формі web-додатку. Це зробить додаток легким і зручним у користуванні, а також забезпечить зручність доступу з будь-якого робочого місця [8]. Прийняте рішення є доцільним ще й з огляду на те, що кількість користувачів цих модулів досить велика. Також це робить можливим інтеграцію вказаних підсистем в структуру сайту секції.

Стосовно модуля секретаря, варто відмітити що, користуватися ним буде виключно одна особа. Зазвичай, одна й та ж людина досить довгий термін виконує ці обов’язки. Не потрібно забезпечувати одночасний доступ кількох користувачів до ресурсу, тому пропонується реалізувати цей модуль у форматі windows-додатка. Детально функціональні можливості модуля секретаря буде описано в наступному пункті.

Об’єднуючим елементом, який забезпечить роботу всіх трьох модулів в єдиному інформаційному полі, буде база даних, яка зберігатиме всю необхідну для роботи системи інформацію.

Для декларування процесів, які мають відбуватися в системі, найчастіше використовують графічне представлення. Пріоритет такої форми подання інформації обумовлюється тим, що графічна інформація швидше і простіше сприймається людиною.

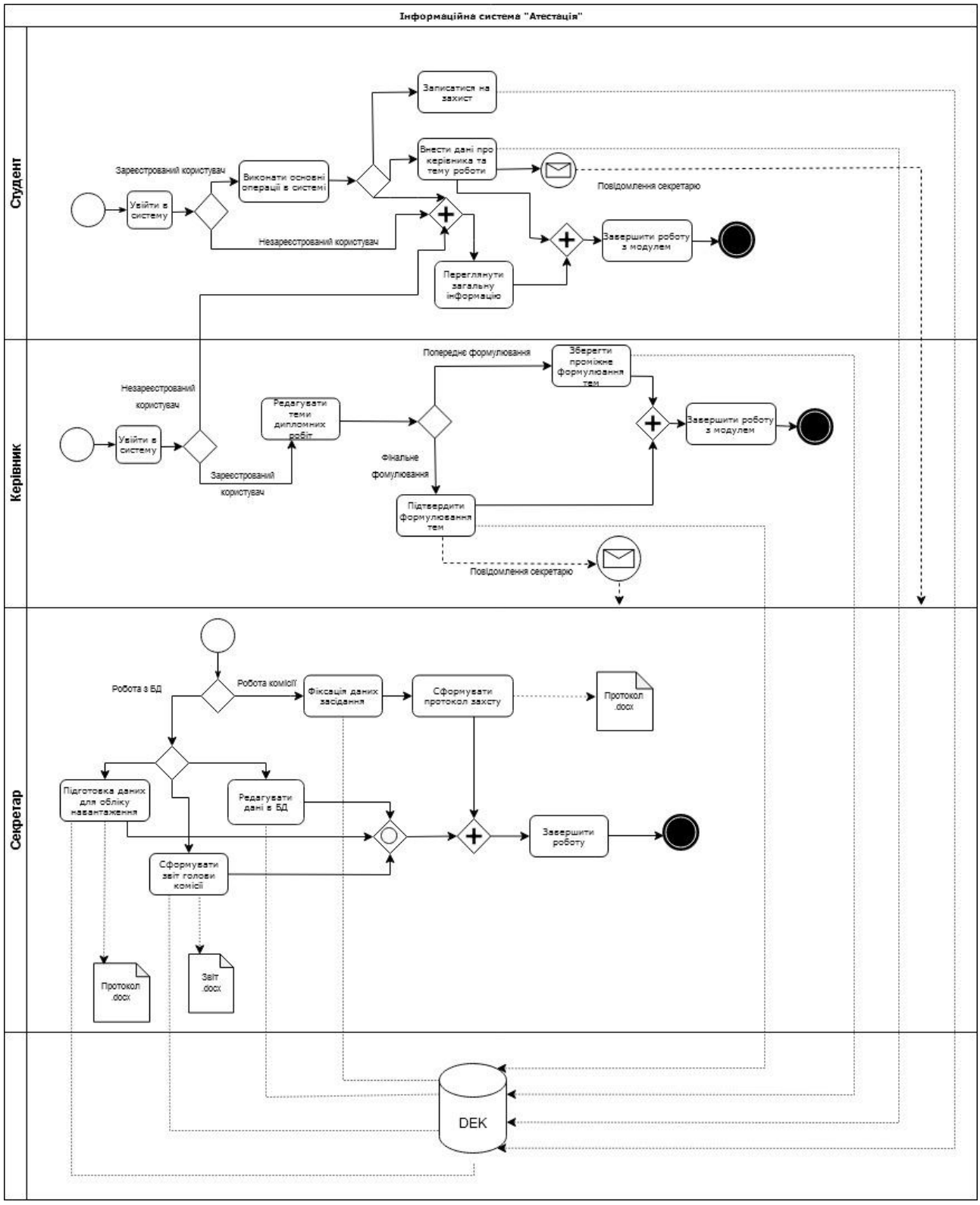


Рисунок 2.1 – Опис бізнес-процесів в системі згідно стандарту BPMN

Серед всіх існуючих стандартів опису бізнес-процесів в різної природи системах, в т.ч. і в інформаційних, наразі найпопулярнішим є BPMN [9-11]. Така популярність визначається тим, що існує достатньо невеликий перелік

графічних елементів, які дозволяють наочно представити будь-які процеси [12-13]. До того ж правила нотації є досить простими. Саме ці факти стали базою для прийняття рішення про побудови моделі бізнес-процесів саме в нотації BPMN.

Побудована модель бізнес-процесів інформаційної системи підтримки роботи атестаційної комісії представлено на рис. 2.1.

З точки зору програмної реалізації структуру інформаційної системи «Атестація» можна представити у вигляді діаграми компонентів згідно стандарту мови моделювання UML [14].

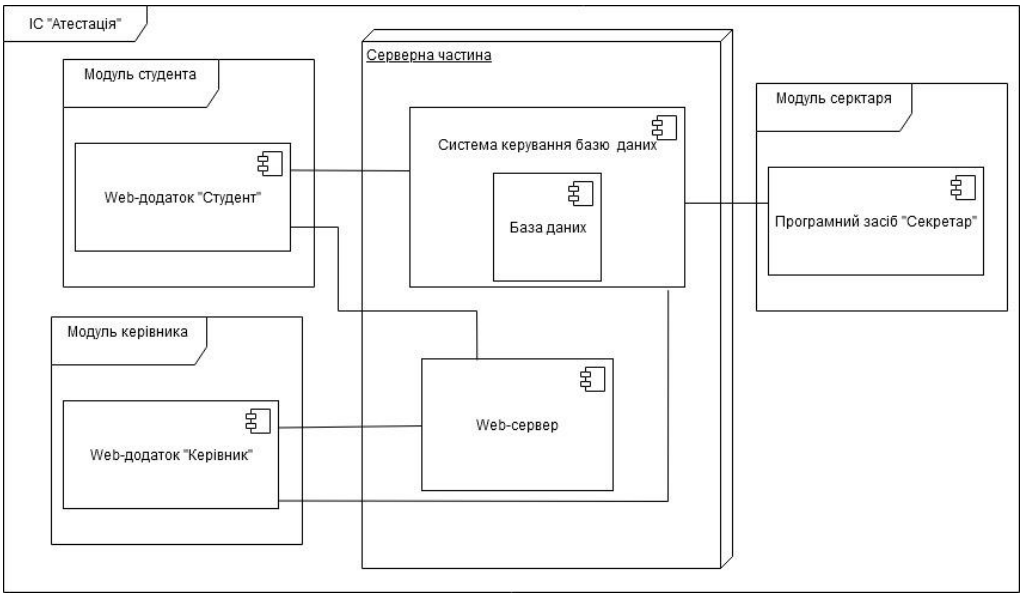


Рисунок 2.2 – Архітектура інформаційної системи

2.2 Модель роботи підсистеми роботи секретаря.

Як зазначалося раніше, прийнято рішення в першу чергу розробити підсистему, яка автоматизує роботу секретаря екзаменаційної комісії. Оскільки модулі студента та керівника не реалізовано, то збір необхідної інформації про

теми та керівників робіт буде забезпечуватися тими технологіями,які описано в п. 2.2. тому вважаємо, що вхідною інформацією для роботи підсистеми є xlsx-файл, що містить всю вказану інформацію. Також до вхідної інформації слід віднести затверджені відповідним положенням [5] шаблони документів, які створюються як результат виконання процесу захисту кваліфікаційних робіт.

Для забезпечення комфортної роботи користувача підсистема на даний момент має реалізовувати такі функції:

- введення вхідної інформації;
- імпорт даних з xlsx-таблиці;
- ведення протоколу засідання екзаменаційної комісії;
- формування звітної документації у форматі документів текстового редактора Microsoft Word.

Для збору і збереження інформації потрібно змоделювати та реалізувати базу даних.

Опис можливостей майбутньої програмної реалізації (рис. 2.3) краще виконати у форматі Use-Case діаграми (діаграми варіантів використання) стандарту мови UML [9].

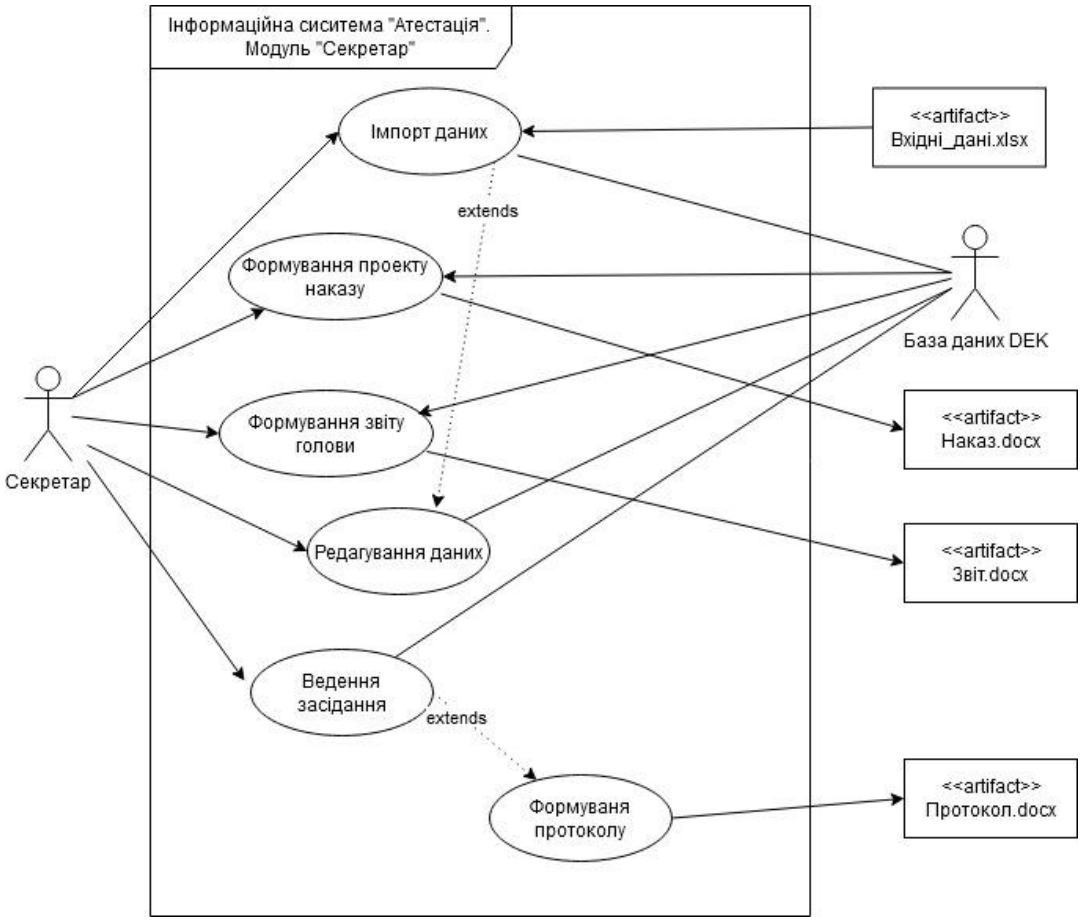


Рисунок 2.3 – Діаграма варіантів використання

Діаграма варіантів використання наочно представляє взаємозв’язок системних акторів з можливими варіантами використання функціоналу програмної реалізації [15-16]. Для модуля, який зараз розглядається, виділено два позасистемні актори: безпосередньо секретар, який може ініціювати виконання функцій (варіантів використання), та база даних, яка зберігає інформацію та опосередковано приймає участь у виконані всіх функцій.

Артефакти системи представляють собою елементи інформації, які використовуються чи генеруються в процесі роботи програмного забезпечення і зберігаються як окремий суб’єкт [17]. В даному випадку до артефактів віднесено всі текстові документи, які будуть генеруватися на основі шаблонів і представляють собою звітну документацію, а також електронна таблиця, яка містить вхідну інформацію.

2.3 Оцінювання робіт.

Загальна оцінка роботи у відповідності до прийнятої в СумДУ системи оцінювання виставляється в трьох системах: в балах (від 0 до 100 балів): національна шкала (відмінно, добре, задовільно, незадовільно) та у формі оцінки за шкалою ECTS (A, B, C, D, E, FX, F). Тобто загальний результат студента $P_{заг}$ можна представити як сукупність (2.1) трьох цих оцінок [18]

$$P_{заг} = \{O_б, O_{нац}, O_{ECTS}\}, \quad (2.1)$$

де $P_{заг}$ – загальний результат оцінювання роботи студента,

$O_б$ – оцінка студента в балах,

$O_{нац}$ – оцінка студента згідно національної системи оцінювання,

O_{ECTS} – оцінка студента за шкалою ECTS.

У визначенні загального результату роботи студента стартовою є оцінка його роботи в балах, тобто показник $O_б$. Визначається ця оцінка у відповідності до прийнятих на методичному семінарі секції інформаційних технологій проектування критеріїв, а саме:

- K_1 – загальна оцінка керівника роботи, визначається на підставі наданого в комісію відгуку керівника, варіюється в діапазоні від 0 до 25 балів;
- K_2 – загальна оцінка рецензента роботи, визначається на підставі наданої в комісію рецензії; варіюється в діапазоні від 0 до 15 балів;
- K_3 – оцінка рівня якості проекту, визначається членами комісії на основі аналізу представлених матеріалів роботи; варіюється в діапазоні від 0 до 35 балів;
- K_4 – оцінка рівня оприлюднення результатів дослідження, визначається секретарем на підставі поданих у комісію копій

відповідних документів (тез, статей, актів впровадження, тощо); варіюється в діапазоні від 0 до 10 балів;

- K_5 – оцінка рівень представлення результатів; визначається комісією за результатами доповіді та враховує якість доповіді та представлення результатів, а також якість відповідей на поставлені питання; варіюється в діапазоні від 0 до 15 балів.

Оцінка студента за 100 бальною шкалою визначається за (2.2):

$$O_б = \sum_{i=1}^{5} K_i. \qquad (2.2)$$

Визначення двох інших оцінок з (2.2) відбувається на основі загальноприйнятих в університеті співвідношень.

3 ІНФОРМАЦІЙНЕ ТА ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМИ

Виконуючи розробку структури інформаційної системи, вдаються до принципу декомпозиції, виділяючи в ній підсистеми більш низького рівня, які в сукупності забезпечать існування та дієздатність системи в цілому. При цьому кожен наступний елемент в отриманій ієрархії, в стою, чергу також розглядається як окрема система [20]. Тому можемо стверджувати, що на даному етапі розробки можна вважати модуль «Секретар» інформаційною системою першого рівня в рамках системи «Атестація» (рис. 2.2.).

У відповідності до загальноприйнятих норм, в модулі «Секретар» виділяються кілька підсистеми, що забезпечують та регламентують її діяльність [20, 21]. Технічна сторона роботи модуля «Секретар» визначається умовою наявності персонального комп’ютера з середньостатистичними технічними параметрами (безпосередньо робоче місце) та наявністю локального сервера у внутрішній мережі секції. Організаційна сторона роботи підсистеми регламентуються тим, що працювати буде лише викладач, який виконує відповідні обов’язки. Правовий аспект регламентується дотриманням чинних норм та шаблонів [5, 6], затверджених відповідними постановами в СумДУ. Інформаційне забезпечення роботи модуля будується за рахунок створення бази даних, яка буде акумулювати в собі всю необхідну інформацію. Математична складова функціонування підсистеми, необхідне для коректного виконання модулем своїх функцій, становить алгоритм визначення загального результату оцінювання роботи, наведений у п.2.3. А також повний спектр умов, які будуть використовуватися в залежності від ситуації для отримання інформації з бази даних через запити. Програмне забезпечення роботи підсистеми, що розглядається, включає в себе: комплекс системних програмних засобів, необхідних для роботи робочого місця та серверу; систему керування

базами даних; текстовий редактор та табличний процесор з пакету Microsoft Office; та безпосередньо програмний додаток «Секретар».

Фактично наразі потребують розробки суб’єкти інформаційного та програмного забезпечень системи.

3.1 Інформаційна підтримка роботи модуля секретаря.

Як зазначалося вище, інформаційне забезпечення підсистеми провадиться за рахунок використання бази даних. Вона повинна містити всю необхідну інформацію про студентів, керівників, рецензентів, склад екзаменаційної комісії, а також інформацію про проведення захисту всіх кваліфікаційних робіт та отримані результати.

Першим етапом розробки бази даних є побудова її логічної схеми, яка дасть чітке уявлення про всю інформацію, яка має зберігатися. Проаналізувавши предметну область та процеси які відбуваються, з точки збереження та обробки інформації можна виділити такі сутності:

- «Студент» з таким переліком атрибутів: прізвище ім’я по батькові (ПІБ) студента, навчальна група, прізвище ім’я по батькові керівника, прізвище ім’я по батькові рецензента, тема роботи, оцінка керівника, оцінка рецензента;
- «Керівник» з таким переліком атрибутів: прізвище ім’я по батькові місце роботи, посада, вчене звання, науковий ступінь;
- «Рецензент» з таким переліком атрибутів: прізвище ім’я по батькові місце роботи, посада, вчене звання, науковий ступінь;
- «Член комісії» з таким переліком атрибутів: номер комісії, прізвище ім’я по батькові місце роботи, посада, вчене звання, науковий ступінь, роль в комісії;

- «Засідання» з таким переліком атрибутів: номер засідання, дата проведення, час початку, час закінчення, перелік ПІБ присутніх членів комісії, перелік ПІБ членів комісії, хто поставив питання, прізвище ім’я по батькові студента, який захищався, характеристика відповідей студента, оцінка оприлюднення результатів роботи; оцінка рівня виконання роботи, оцінка рівня представлення матеріалів, загальна оцінка за 100-бальною шкалою, оцінка за національною шкалою, оцінка за шкалою ECTS, присвоєна кваліфікація, відбувався захист англійською мовою, категорія диплома (з відзнакою чи ні).

Взаємозв’язки між цими сутностями представимо у форматі ER-моделі за нотацією Чена [22-24] (рис.3.1). як видно з рисунка, для сутностей встановлено такі зв’язки:

- зв'язок «виставити оцінку» між сутностями «Керівник» та «Студент» визначається тим, що керівник керує роботою студента, визначає тему його роботи та вказує свою оцінку роботи; зв’язок типу 1:N;
- зв'язок «прорецензувати роботу» між сутностями «Рецензент» та «Студент» визначається тим, що рецензент знайомиться з роботою і виставляє свою оцінку; зв’язок типу 1:N;
- зв'язок «надати результати роботи» між сутностями «Студент» та «Засідання» визначається тим, що студент надає матеріали роботи, виконує доповідь, відповідає на питання; зв’язок типу N:1;
- зв'язок «оцінити роботу» між сутностями «Член комісії» та «Засідання» визначається тим, що член комісії приймає участь в роботі засідання, ставить питання студенту та приймає участь в загальній оцінці роботи; зв’язок типу N:1.

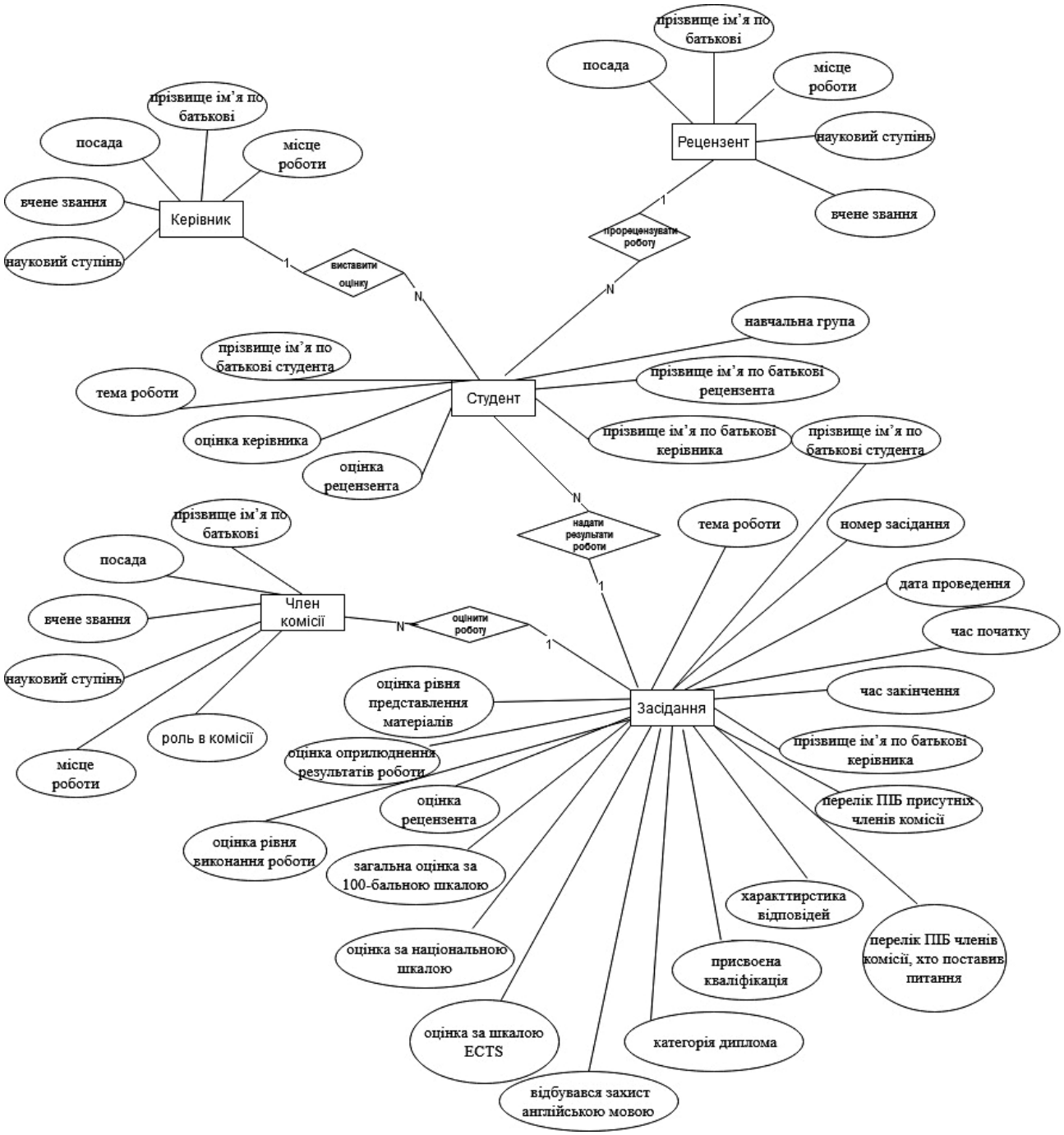


Рисунок 3.1 – ER-діаграма

Якщо проаналізувати виділені атрибути сутностей, то можна побачити, що сутності «Керівник» та «Рецензент» мають однаковий перелік атрибутів, а сутність «Член комісії» відрізняється від них лише одним атрибутом. Тому прийнято рішення об'єднати їх в одну сутність, інкапсулюючи в ній атрибути та додавши атрибути, які будуть визначати роль конкретної особистості в атестаційному процесі. Також можна помітити дублювання однієї й тієї ж

інформації в різних сутностях. Якщо всі атрибути скласти в одне універсальне відношення, то воно буде містити значну кількість збиткової інформації. Тому перш ніж переходити до фізичної реалізації, потрібно провести нормалізацію цього відношення [25-27].

У відповідності до загальноприйнятих підходів на першому етапі уникли дублювання інформації за рахунок винесення її в окремі таблиці. Наприклад, з таблиці student винесено в окрему таблицю qualification всю інформацію, яка стосується освітнього рівня, за яким захищається студент, та кваліфікації, яку він здобуває (копія з екрану наведена на рис. 3.2).

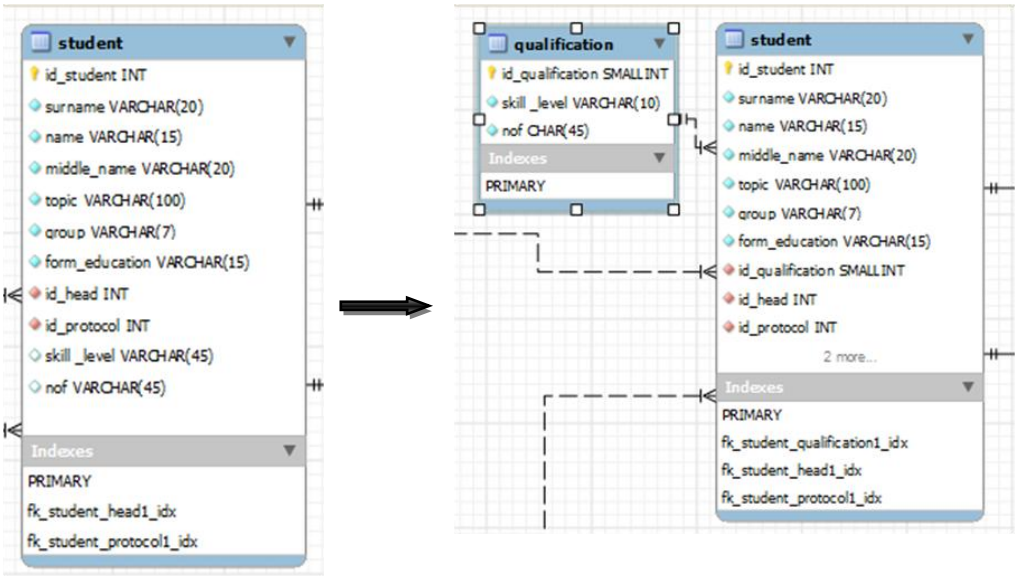


Рисунок 3.2 – Зміни в структурі бази даних

Також було розділено інформацію про саме засідання та присутніх на ньому членів комісії на окремі таблиці, з прив’язкою до таблиці з інформацією про учасників процесу (рис. 3.3).

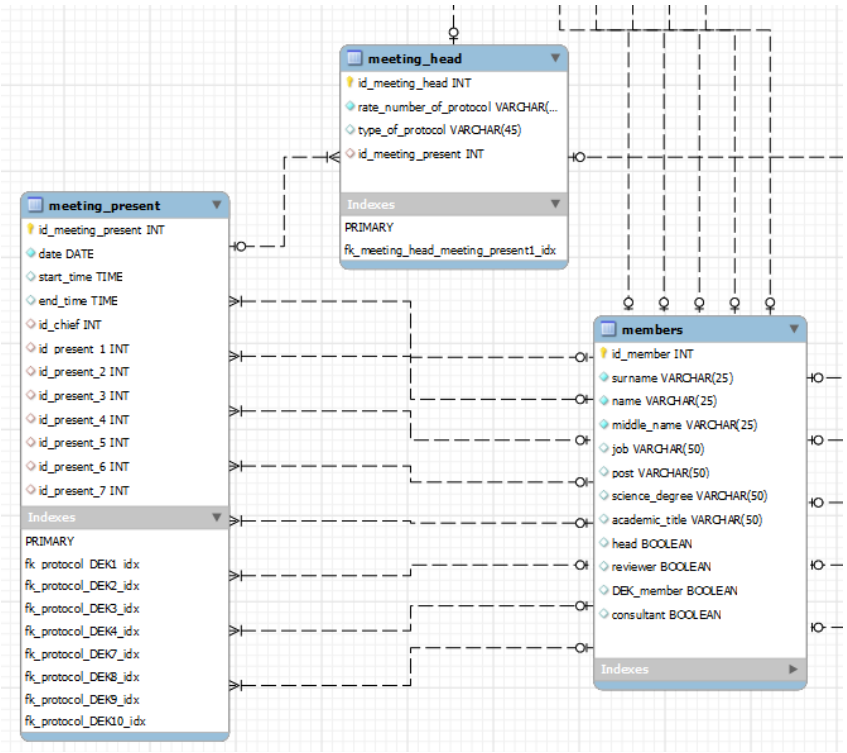


Рисунок 3.3 – Виділення додаткових таблиць

В структурі бази даних прийнято рішення створити три довідкові таблиці, які не мають відношення з будь-якими іншими таблицями і слугують для збереження певних характеристик. Це таблиці з даними про три шкали оцінювання роботи, характеристиками відповідей студента, інформацією про кваліфікації, що можуть здобуватися, а також таблиця, в якій будуть зберігатися номери екзаменаційних комісій.

Отримана в результаті проведеної роботи схема бази даних представлена на рис.3.4. Розробка виконувалася у програмному продукті MySQL Workbench, який надає розробнику зручний графічний інструментарій для побудови схем баз даних. Також значною перевагою використання цього програмного продукту є те, що він містить вбудований засіб перетворення отриманої графічної інтерпретації у скрипт мовою sql, який наведено у додатку А.

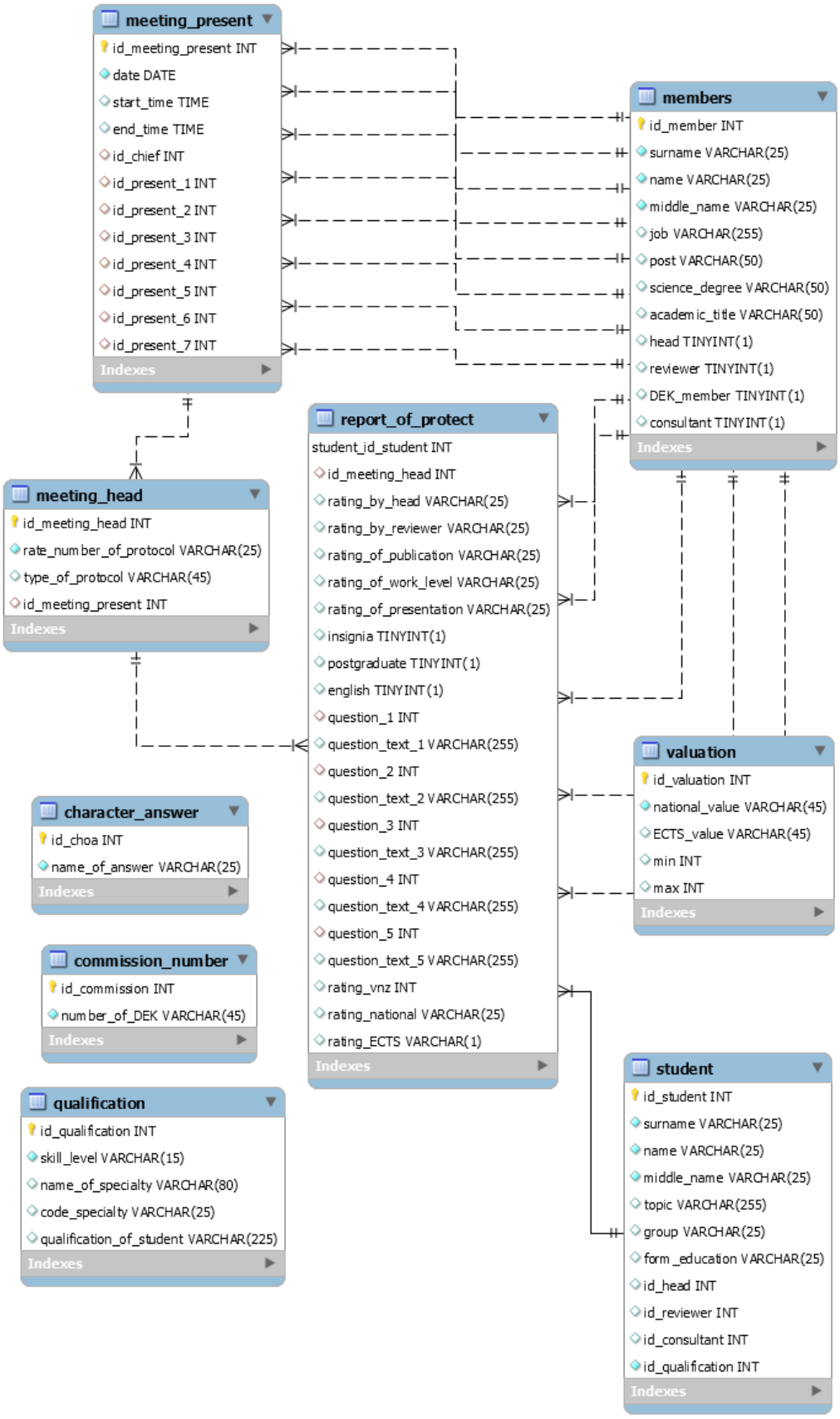


Рисунок 3.4 – Схема бази даних

Отриманий скрипт імпортовано на сервер системи керування базами даних MySQL. В результаті створено структуру бази даних, визначено таблиці та налаштовано їх поля. В довідкові таблиці за допомогою інструментарію Workbench введено інформацію. На рис. 3.5 наведено копію вікна з заповненою таблицею.

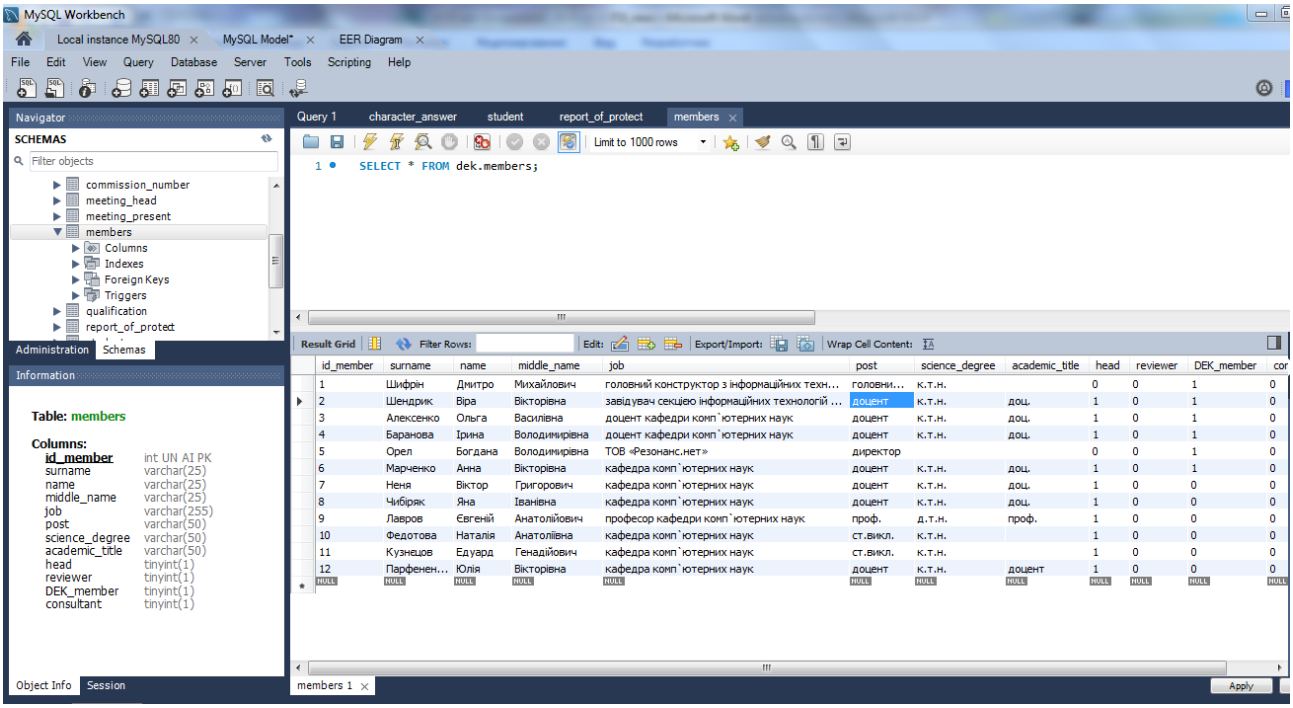


Рисунок 3.5 – Створена база даних з заповненою таблицею

3.2 Розробка інтерфейсу користувача.

Перше, з чим стикається користувач будь-якої програми, це графічний інтерфейс користувача. Тому при розробці програмного забезпечення варто приділити цьому питанню особливу увагу, щоб забезпечити зручність і зрозумілість в майбутній роботі.

Згідно загальноприйнятих рекомендацій при розробці інтерфейсу користувача модуля «Секретар» дотримувалися таких правил [28, 29]:

- бажано використовувати усталені компоненти, які притаманні більшості програмного забезпечення, щоб користувач інтуїтивно розумів їх призначення і не витрачав час на вивчення нових елементів;
- потрібно контролювати дії користувача та передбачити зворотній зв’язок через інформативні повідомлення;
- не потрібно перевантажувати увагу одночасно великою кількістю інформації чи елементів впродовж роботи програми.

Як зазначалося в п. 2.1 програмне забезпечення модуля «Секретар» вирішено виконати у формі одновіконного додатка. З урахуванням визначеного функціоналу виконання реалізації обрано мову програмування C# з використанням інтерфейсу Windows Forms [30, 31]. Розробка буде проводитися в середовищі Microsoft Visual Studio 2015.

Весь функціонал додатку можна структурно поділити на кілька логічних груп. Дотримуючись описаних вище принципів, вирішено розподілити основні дії користувача на три логічно визначені групи:

- введення/імпорт вхідної інформації;
- ведення протоколу засідання;
- перегляд бази даних;
- формування проекту звітної документації голови комісії.

Для такого логічного розподілу в інтерфейсі користувача прийнято рішення використати бібліотечний компонент TabControl, створивши в ньому відповідно чотири вкладки (рис. 3.6).

Візуальне групування блоків логічно спорідненої інформації виконано за допомогою компоненту groupBox1 (рис. 3.7, наприклад, група «Інформація про студента», «Інформація про члена комісії,керівників,рецензентів»).

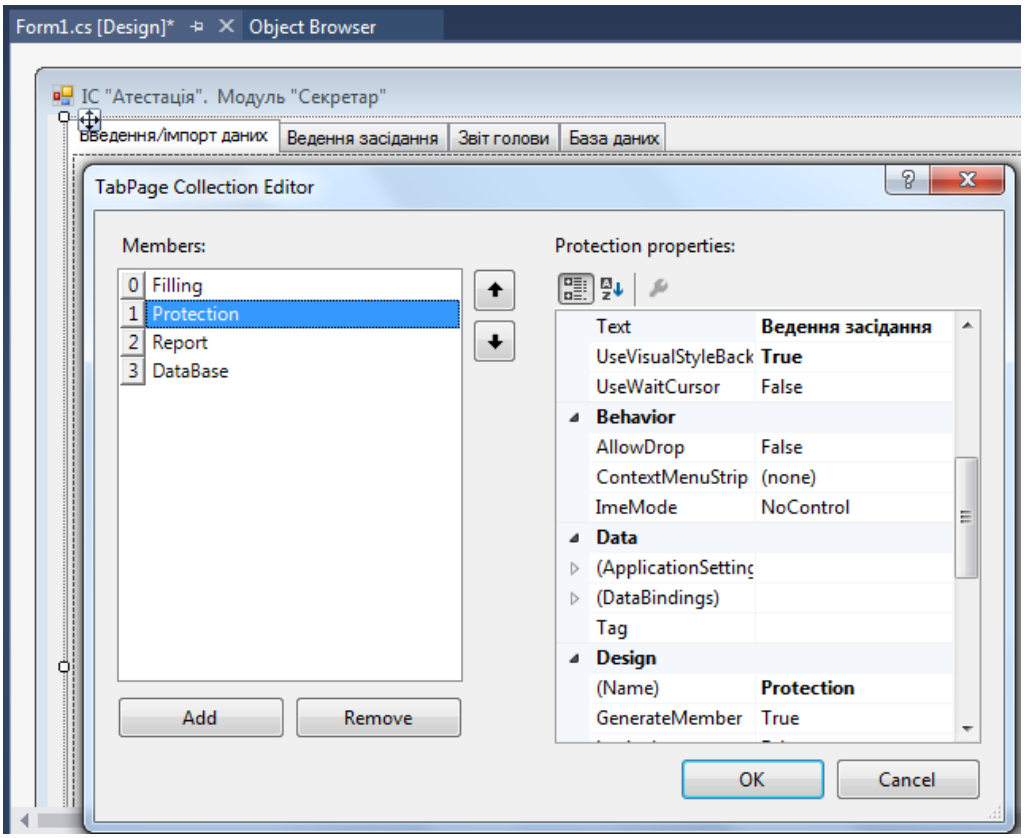


Рисунок 3.6 – Формування вкладок робочого вікна

Частину інформації, якою оперує користувач, може вводитися з клавіатури. Наприклад, при необхідності додати інформацію в базу даних. Для забезпечення такого функціонал використано стандартний компонент TextEdit (рис. 3.7).

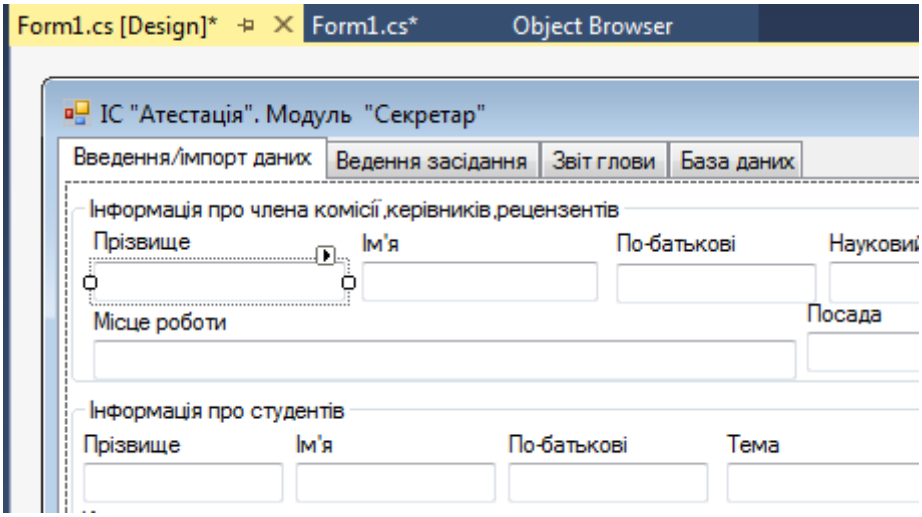


Рисунок 3.7 – Забезпечення введення інформації

Деяка інформація представляє собою вибір одного варіанту з переліку можливих. Для реалізації таких дій використовуються випадаючі списки ComboBox (рис. 3.8). В деяких міститься статична інформація (наприклад, форма навчання), до інших інформація додається з бази даних (наприклад, вибір студента в ході засідання), відповідно, сам список формується динамічно, програмними засобами.

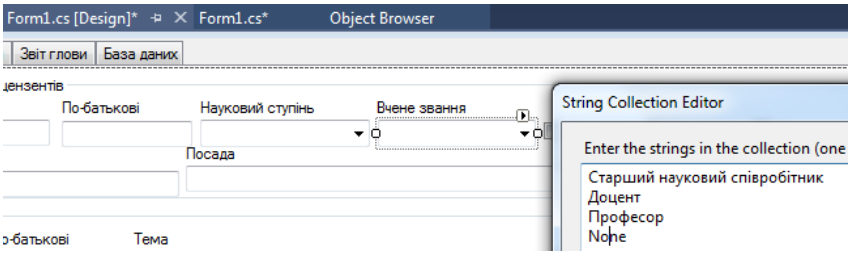


Рисунок 3.8 – Формування статичного переліку випадаючого списку

Для організації визначення статусу учасника атестаційного процесу при введенні даних використано перелік компонентів CheckBox (рис.3.9).

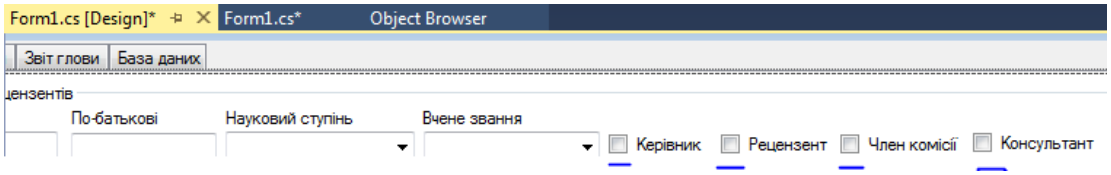


Рисунок 3.9 – Організація вибору

Оскільки користувач буде виконувати не таку вже й велику кількість дій, то вважаємо, що в формуванні головного меню немає сенсу, а всі дії будуть ініціюватися натисканням кнопок (компонент Button) чи настанням події вибору окремого елемента в випадаючому списку (наприклад, вибір бази даних для перегляду).

Окрім цього, у проекті використано стандартний компонент відображення таблиць DataGridView, який дозволяє відобразити дані з вказаного джерела-таблиці у прийнятному форматі. У якості джерела інформації в системі

використовуються дані із запитів до бази даних, тому заповнення виключно відбувається динамічно під час роботи програми.

Для виконання операцій імпорту чи створення текстових документів передбачено можливість обрати відповідні файли. Тому в проекті використано стандартні компоненти діалогових вікон відкриття OpenFileDialog та збереження SaveFileDialog файлів. Діалогове вікно відкриття налаштоване відображати у вікні діалогу файли додатка MS Excel із розширенням *.xls та *.xlsx. При збереженні результатів в діалоговому вікні відображаються та зберігаються файли формату MS Word (*.doc та *.docx).

Інформування користувача про успішне виконання дії, наприклад, при роботі з базою даних відбувається за допомогою рядка стану, в якому виводиться повідомлення в залежності від того, який вид робот виконується в поточний момент. Реалізовано за допомогою стандартного компоненту рядка стану StatusStrip (рис. 3.10).

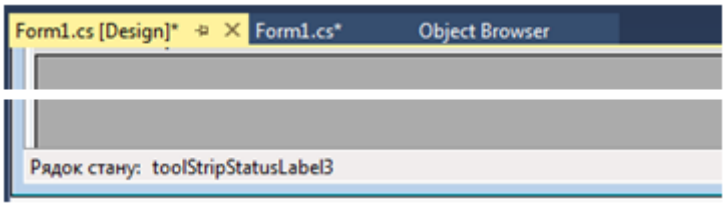


Рисунок 3.10 – Організація вибору

Такий підхід водночас не залишає користувача без інформації, але й не відволікає його від роботи у разі, якщо все працює в шатному режимі. На противагу цьому, у разі виникнення помилки (наприклад, не вдалося з’єднатися з базою даних чи при спробі занести в базу даних інформацію про засідання користувач вказав менше чотирьох присутніх членів комісії), варто одразу привернути увагу користувача та призупинити робочий процес програми. В цьому випадку на екран буде виводитися відповідне повідомлення (рис.3 11).

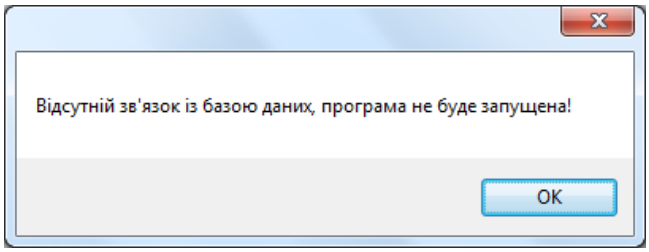


Рисунок 3.11 – Зразок вікна з повідомленням про помилку

На рис. 3.12 наведено прототип інтерфейсу однієї з вкладок робочого вікна програми.

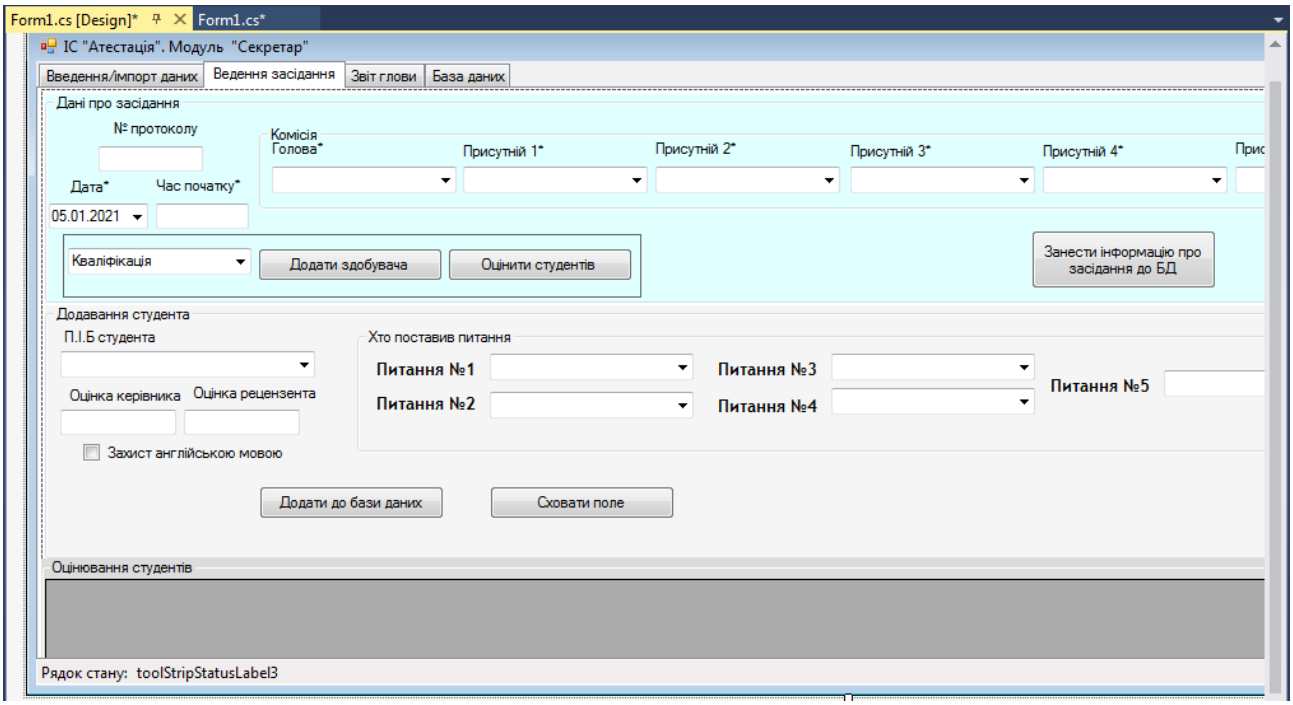


Рисунок 3.12 – Інтерфейс користувача при роботі секретаря по веденню засідання

3.3 Програмна реалізація.

Програмне забезпечення модуля секретаря виконується у парадигмі подіє-орієнтованого програмування. Тобто робота програми визначається набором

обробників тих подій, які будуть згенеровані операційною системою у відповідь на дії користувача у створеному інтерфейсі. Тому при розробці програмного коду слід запрограмувати реакцію модуля на ці дії. Як зазначалося раніше, мовою програмування обрано C# [32].

Оскільки вся робота секретаря пов’язана з обробкою інформації з бази даних, то перш за все потрібно налаштувати з’єднання з нею. Для реалізації дій з базою даних, якою керує сервер MySQL, було додатково встановлено та додано до залежностей проекту спеціальну бібліотеку MySQL Connector NET [33].

Для роботи в проект імпортовано стандартні бібліотеки конектора MySql.Data.MySqlClient (простір імен для створення об’єктів класів MySQL Server, які забезпечують з’єднання та роботу з базою даних) та MySql.Data (більш широкий простір імен додаткових класів MySQL Server).

Крім того у проекті використовуються такі класи та компоненти, які відносяться до стандартних класів конектора:

- DataTable – описує одну таблицю з даними в пам’яті;
- MySqlCommand – визначає собою об'єктно-орієнтоване представлення SQL-запиту;
- MySqlDataReader – реалізує можливість читання потоку рядків з бази даних.

З’єднання з базою даних встановлюється за допомогою створеного об’єкту класу. В конструкторі об’єкта вказується рядок підключення, в якому зазначаються IP-севера, логін та пароль доступу, тощо. Всі ці параметри, щоб не вводити кожного разу при запуску програми, зберігаються в спеціальному текстовому файлі (рис.3.13). Зчитування інформації з файлу виконується за допомогою стандартного класу потокового читання StreamReader, попередньо перевіривши, чи в папці додатку існує цей файл.

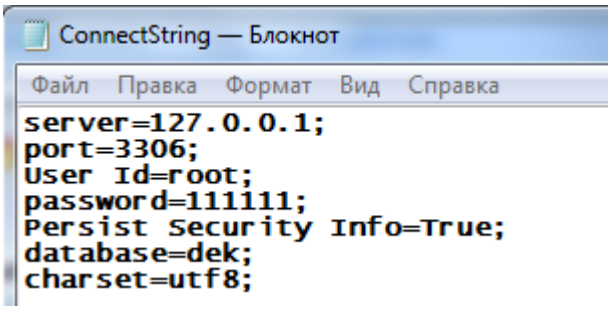


Рисунок 3.13 – Конфігураційний файл

Створення підключення виконується в обробнику події завантаження об’єкта форми Load. У випадку невдалого з’єднання виводиться вікно з відповідним повідомленням і переривається робота програми. В іншому випадку, перш ніж виконувати дії з базою даних, потрібно відкрити це з’єднання методом MySqlConnection.Open(). Фрагмент коду наведено на рис. 3.14.

```
mySQLConnection = new MySqlConnection(Connect);
mySQLCommand = new MySqlCommand();
mySQLCommand.Connection = mySQLConnection;
```

Рисунок 3.14 – Код встановлення з’єднання

Як і зазвичай, вся робота з інформацією з бази даних виконується за допомогою мови SQL запитів [34].

Створення і відправлення запиту на сервер виконується з використанням класу MySqlCommand за допомогою таких членів:

- в полі MySqlCommand.Connection вказуємо об’єкт поточного підключення;
- поле MySqlCommnd.CommandText – рядок, який містить текст запиту для відправки до бази даних.

Якщо запит на вибірку (SELECT), то його виконання здійснюється за допомогою методу MySQLCommand.ExecuteReader(), в результаті роботи якого повертається отримана інформація у вигляді об’єкта класу MySqlDataReader.

У випадку, коли запит спрямовано на оновлення інформації в базі даних (INSERT чи UPDATE), то його виконання ініціюється методом MySqlCommand.ExecuteNonQuery().

Для реалізації функції імпорту даних про студентів та членів екзаменаційної комісії було розроблено власну систему класів (рис. 3.15). Основний клас Import, методи якого виконують всі необхідні дії по обміну даними між електронною книгою та базою даних. Код класу наведено в додатку Б.

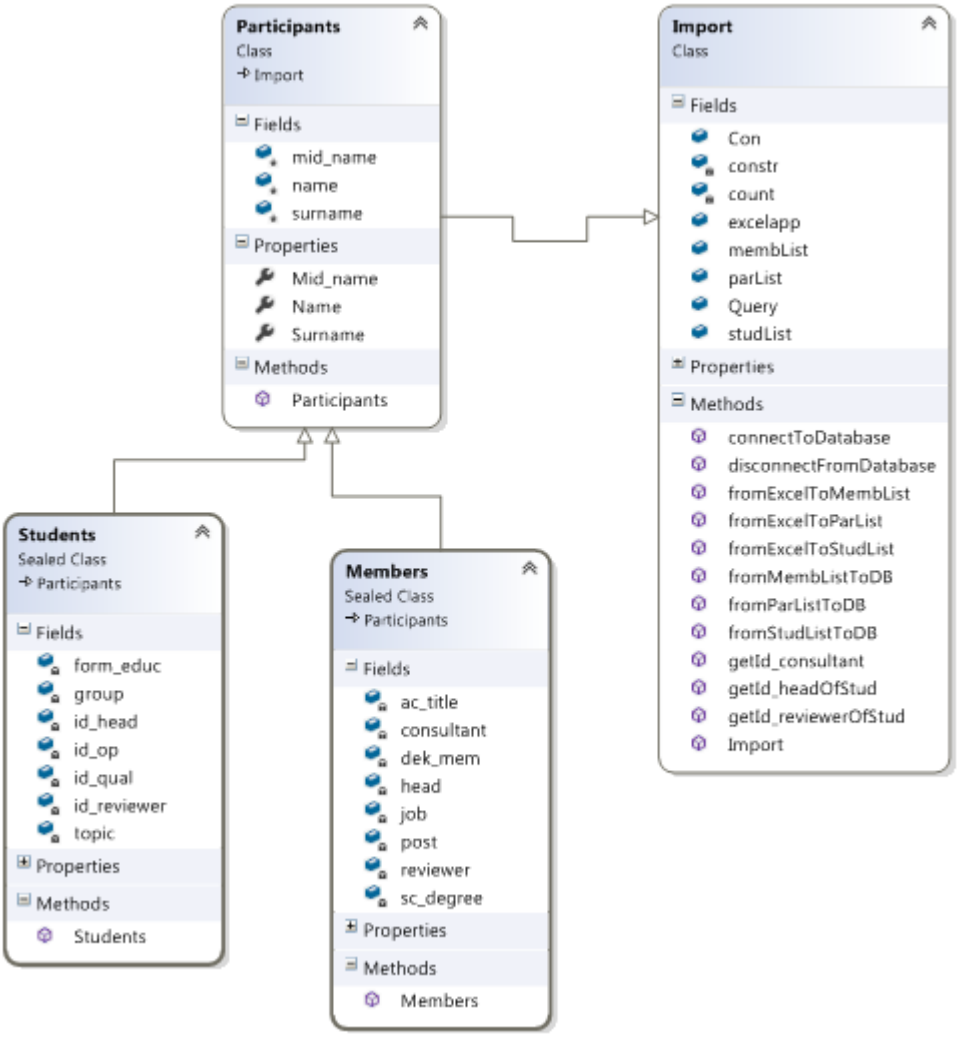


Рисунок 3.15 – Діаграма створених класів

Ще одна функціональна можливість програми передбачає автоматизацію процесу створення звітної документації. Як для цього етапу, так і для забезпечення можливості імпорту, було використано пакет Microsoft.Office.Interop.Word [35], який містить систему класів для доступу до об’єктної моделі програмних продуктів пакету Microsoft Office.

Для забезпечення роботи пакету було додано до переліку залежностей, що дало змогу скористатися класами цього простору імен, зокрема:

- Microsoft.Office.Interop.Word.Application – клас об’єкта-додатка MS Word;
- Microsoft.Office.Interop.Word.Document – клас, призначений роботи з документом MS Word.

Документи Word створюються для протоколу, звіту голови та проекту наказу. Для кожного з них розроблено відповідні шаблони документів, які зібрано у відповідну окрему папку в рамках проекту. При натисканні відповідної кнопки, яка відповідає за створення того чи іншого виду документу, на сонові відповідного шаблону створюється документ, відкривається стандартний діалог для збереження файлу. Відповідний фрагмент коду наведено на рис. 3.16.

```
//створення документа за шаблоном
WordApp = new Microsoft.Office.Interop.Word.Application();
WordDoc = WordApp.Documents.Open(Application.StartupPath + "\\templates\\protection.dot");

//виведення вікна для збереження готового звіту
saveFileDialog1.ShowDialog();
saveFileDialog1.AddExtension = true;
saveFileDialog1.CheckPathExists = true;
if (saveFileDialog1.FileName != "")
{
    WordDoc.SaveAs(saveFileDialog1.FileName);
}
```

Рисунок 3.16 – Створення документу Word за шаблоном

Для внесення інформації в документ використано метод «пошуку-заміни». Заздалегідь у створених шаблонах в потрібних місцях виконано

спеціальні позначки-мітки. Кожна позначка однозначно визначає одну єдину категорію даних в рамках документа. При виконанні безпосереднього процесу заповнення виконується пошук потрібної мітки і заміна відповідними даним. На рис. 3.17 наведено приклад такого шаблону.

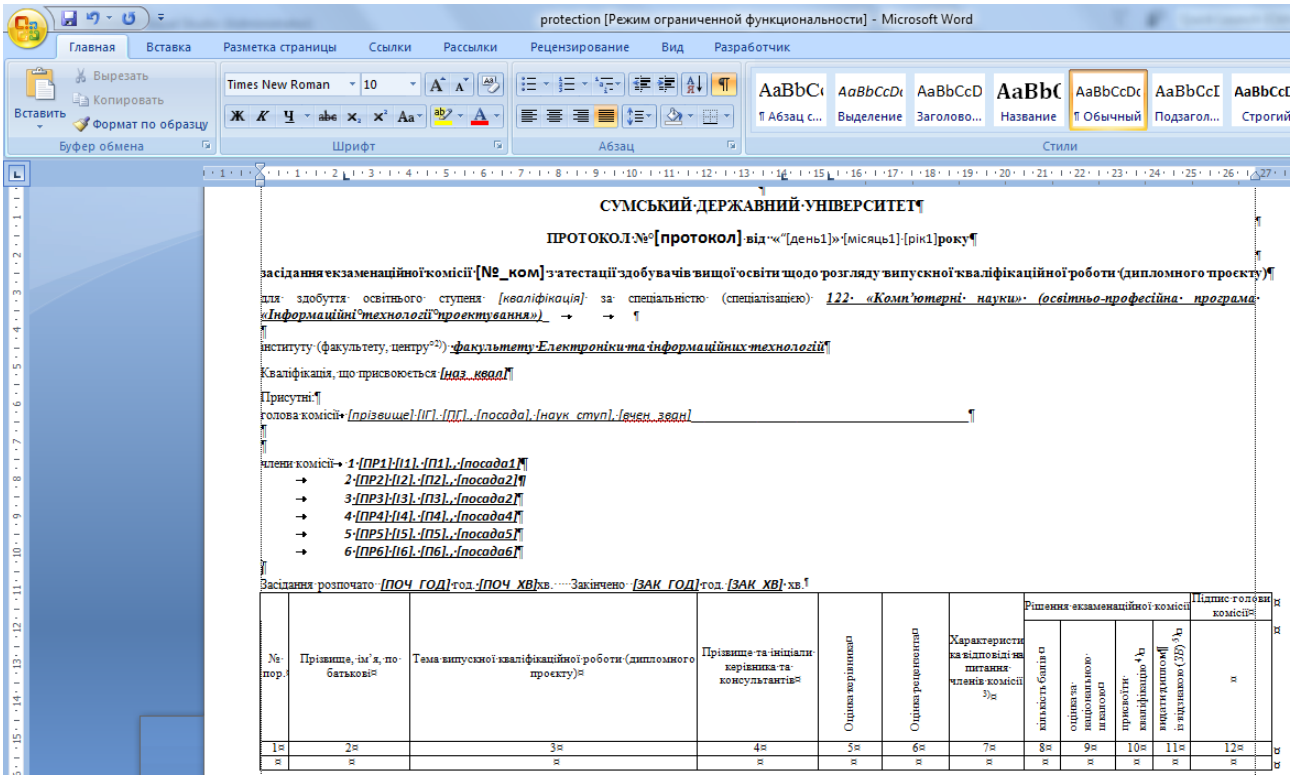


Рисунок 3.17 – Шаблон протоколу засідання з мітками для заміни

Для реалізації описаних вище дій використовується клас пошуку Microsoft.Office.Interop.Word.Find та такі його члени:

- ClearFormatting() – метод очистки попереднього формату пошуку;
- поле Text – містить мітку, яку треба знайти та замінити;
- метод Replacement.ClearFormatting() – метод очистки попереднього формату заміни;
- поле Replacement.Text – містить текст, на який треба замінити мітку;

- метод Execute() – запускає процес пошуку/заміни. В параметрах виклику методу вказується тип пошуку: множинний (всі мітки) чи одиночний (поточна знайдена).

```
private void replaceText(Microsoft.Office.Interop.Word.Find WordFind, string search, string replace)
{
    WordFind.ClearFormatting();
    WordFind.Text = search;
    WordFind.Replacement.ClearFormatting();
    WordFind.Replacement.Text = replace;
    WordFind.Execute(Replace: Microsoft.Office.Interop.Word.WdReplace.wdReplaceAll);
}

....

replaceText(WordFind, "[протокол]", monthToString(dateProtection.Text.Split('-')[1]));
```

Рисунок 3.18 – Код методу заміни тексту (приклад)

Для заповнення таблиці використано колекцію Tables документа Microsoft.Office.Interop.Word.Document [36]. Заповнення відбувалося даними з запитів. Інформація поміщалася в конкретну комірку рядка, доступ до якої здійснювався методом Cell об’єкта таблиці. Параметрами методу виступають координати комірки – номери рядка та стовпчика. Нові додаються також динамічно. Фрагмент коду заповнення наведено на рис. 3.19.

```
if (mySQLReader.Read())
{
    WordTable = WordDoc.Tables[1];
    i = 0;
    WordTable.Cell(i + 4, 1).Range.Text = (i + 1).ToString();
    WordTable.Cell(i + 4, 8).Range.Text = mySQLReader.GetString(3);
    WordTable.Cell(i + 4, 9).Range.Text = mySQLReader.GetString(4);
    WordTable.Cell(i + 4, 10).Range.Text = (Int32.Parse(mySQLReader.GetString(3)) >  60) ? "так" : "ні";
    WordTable.Cell(i + 4, 11).Range.Text = (mySQLReader.GetString(6) == "True") ? "ЗВ" : "-";

    while (mySQLReader.Read())
    {
        i++;
        WordTable.Cell(WordTable.Rows.Count, 1).Range.Rows.Add();
        WordTable.Cell(i + 4, 1).Range.Text = (i + 1).ToString();
        WordTable.Cell(i + 4, 8).Range.Text = mySQLReader.GetString(3);
        WordTable.Cell(i + 4, 9).Range.Text = mySQLReader.GetString(4);
        WordTable.Cell(i + 4, 10).Range.Text = (Int32.Parse(mySQLReader.GetString(3)) > 60) ? "так" : "ні";
        WordTable.Cell(i + 4, 11).Range.Text = (mySQLReader.GetString(6) == "True") ? "ЗВ" : "-";
    }
}
```

Рисунок 3.19 – Фрагмент коду заповнення таблиці

На рис. 3.19 наведено копію документа після внесення даних.

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

ПРОТОКОЛ № *21-18* від *«21» грудня 2020* року

засідання екзаменаційної комісії № *21* з атестації здобувачів вищої освіти щодо розгляду випускної кваліфікаційної роботи (дипломного проєкту)

для здобуття освітнього ступеня *магістр* за спеціальністю (спеціалізацією) *122 «Комп'ютерні науки» (освітньо-професійна програма «Інформаційні технології проектування»)*

інституту (факультету, центру [2]) *факультету Електроніки та інформаційних технологій*

Кваліфікація, що присвоюється *магістр з комп`ютерних наук*

Присутні:
голова комісії *Шифрін Д. М., головний конструктор з інформаційних технологій АТ "Науково-дослідний і проектно-конструкторський інститут атомного та енергетичного насособудування"*

члени комісії 1. *Шендрик В. В., завідувач секцією інформаційних технологій проектування кафедри КН*
2. *Баранова І.В., доцент кафедри КН*
3. *Лавров Є.А., професор кафедри КН*
4. *Орел Б. В., директор ТОВ «Брокодерс»*

Засідання розпочато **10** год.**00** хв. Закінчено **13** год. **00** хв.

№ пор.	Прізвище, ім'я, по батькові	Тема випускної кваліфікаційної роботи (дипломного проекту)	Прізвище та ініціали керівника та консультантів	Оцінка керівника	Оцінка рецензента	Характеристика відповіді на питання членів комісії [3]	Рішення екзаменаційної комісії				Підпис голови комісії
							кількість балів оцінки за національною шкалою	оцінка за національною шкалою	присвоїти кваліфікацію [4]	видати диплом із відзнакою (ЗВ) [5]	
1	2	3	4	5	6	7	8	9	10	11	12
1	*Бабій Євгеній Андрійович*	*Модулі планування та ідентифікації для мобільного додатку Plannit*	*Баранова І.В.*	*Відмінно*	*Відмінно*	*Вірна*	*85*	*Добре*	*так*	-	

Рисунок 3.19 – Приклад заповненого протоколу

Коди основних методів класу форми наведено в додатку Б.

ВИСНОВКИ

В роботі вирішено актуальну задачу забезпечення підтримки проведення процесу атестації здобувачів вищої освіти за рахунок створення відповідної інформаційної системи.

В результаті аналізу самого процесу в тому вигляді, як побудований на сьогодні, встановлено, що він супроводжується великими обсягами інформації. Обробку цієї інформації виконує секретар, використовуючи при цьому розрізнене програмне забезпечення. Такий підхід не є ефективним. Це підтверджує актуальність обраної тематики.

Виявлені в результаті дослідження переметної області недоліки використання інформаційних технологій та отримане чітке уявлення про сам процес дозволили розробити загальну концепцію інформаційної системи підтримки роботи атестаційної комісії та представити відповідну модель в нотації стандарту BPMN. Визначено структурний склад та описано роботу виділених модулів. Визначено функціональні вимоги до модуля підтримки роботи секретаря, детально змодельовано його роботу.

Для інформаційної підтримки процесу розроблено схему та створено відповідну базу даних. В якості СУБД використовується MySQL Server. Виконано програмну реалізацію модуля «Секретар» у вигляді одновіконного windows-додатка мовою програмування C#.

Створене прогар мне забезпечення «Модуль «Секретар» відповідає всім визначеним функціональним вимогам. Практична значимість отриманої розробки полягає в тому, що спрощується робота секретаря з інформацією, зменшується час на підготовку необхідних матеріалів.

В подальшому буде проводитися розробка двох інших модулів, визначених в складі інформаційної системи.

СПИСОК ЛІТЕРАТУРИ

1. Стандарт вищої освіти України першого (бакалаврського) рівня ступеня «бакалавр» за галуззю знань 12 «Інформаційні технології» спеціальністю 122 «Комп'ютерні науки». Затверджено та введено в дію наказом МОН України від 10.07.2019 р. № 962. – Київ, 2019. – 24 с.
2. Сумський державний університет. Освітньо-професійна програма «Інформаційні технології проектування», рівень підготовки - бакалавр. – https://op.sumdu.edu.ua/#/programm/1556
3. Сумський державний університет. Освітньо-професійна програма «Інформаційні технології проектування», рівень підготовки - магістр. – https://op.sumdu.edu.ua/#/programm/1557
4. Васильєв А. В. ІТ-забезпечення діяльності інноваційного університету: досвід українського вишу / А. В. Васильєв, В. О. Любчак, Ю. О. Зубань та ін.. – Суми : СумДУ, 2016. – 173 с.
5. Положення про порядок створення та організацію роботи екзаменаційних комісій Сумського державногоуніверситету з атестації здобувачів вищої освіти. Наказ №0471-І від 03 червня 2020р. – https://normative.sumdu.edu.ua/?task=getfile&tmpl=component&id=f25f99b8-f63c-ea11-912d-001a4be6d04a&kind=1
6. Про порядок рецензування кваліфікаційних робіт здобувачів вищої освіти. Наказ №0879-І від 30 жовтня 2020р. – https://normative.sumdu.edu.ua/?task=getfile&tmpl=component&id=0438b192-be1a-eb11-a666-d4856459ca35&kind=1
7. Проектування інформаційних систем. CASE – технології – https://pidru4niki.com/18580318/informatika/proektuvannya_informatsiynih_sistem_case_tehnologiyi

8. Тузовський А.Ф. Проектування і розробка web-додатків – https://stud.com.ua/97571/informatika/proektuvannya_i_rozrobka_web-dodatkiv
9. Мартинюк О. А. Особливості опису бізнес-процесів в сучасних IT-системах. – http://www.economy.nayka.com.ua/?op=1&z=3514
10. Артамонов И. В. Современные стандарты описания и исполнения бизнес-процессов. – http://ecm-journal.ru/post/Sovremennye-standarty-opisanija-i-ispolnenija-biznes-processov.aspx
11. The BPMN-XPDL-BPEL value chain – http://kswenson.wordpress.com/2012/05/26/bpmn-xpdl-bpel/
12. BPMN 2.0 Из чего состоит модель бизнес процесса – https://rzbpm.ru/knowledge/bpmn-2-0-iz-chego-sostoit-model-biznes-processa.html
13. Нотация BPMN – https://www.businessstudio.ru/wiki/docs/v4/doku.php/ru/csdesign/bpmodeling/bpmn_notation
14. Леоненков. Самоучитель по UML. – http://khpi-iip.mipk.kharkiv.edu/library/case/leon/index.html
15. Діаграма варіантів використання (USE case diagram) – https://studopedia.ru/19_284009_diagrama-variantiv-vikoristannya-USE-case-diagram.html
16. Общая характеристика языка UML – http://www.informicus.ru/default.aspx?SECTION=6&id=73&subdivisionid=2
17. Справочник UML. Объектно-ориентированное проектирование. – https://openu.ru/Books/UML/Artifact.asp
18. Чернышов В.Н. Теория систем и системный анализ / В.Н. Чернышов, А.В. Чернышов. – http://window.edu.ru/resource/188/64188/files/chernyshov.pdf
19. Грэди Буч. Объектно-ориентированный анализ и проектирование с примерами приложений. – М.:ООО "И.Д. Вильямс", 2010. – 720 с.

20. Буйницька О.П. Структура інформаційної системи – https://pidru4niki.com/1584072029374/informatika/informatsiyni_tehnologiyi_ta_tehnichni_zasobi_navchannya
21. Олійник А.В. Інформаційні системи і технології у фінансових установах / А.В. Олійник, В.М.Шацька – Львів: "Новий Світ-2000", 2006. – 436 с.
22. Проектирование информационных систем: разработка информационной модели – https://sites.google.com/site/anisimovkhv/learning/pris/lecture/tema7/tema7_5
23. Введение в базы данных – http://www.mstu.edu.ru/study/materials/zelenkov/toc.html
24. Basic Concepts of ER Model in DBMS – https://www.studytonight.com/dbms/er-model-concepts.php
25. Руководство по проектированию реляционных баз даннях – https://metanit.com/sql/tutorial/
26. Normalization of Database – https://www.studytonight.com/dbms/database-normalization.php#
27. What is Normalization? – https://www.guru99.com/database-normalization.html
28. Пользовательский інтерфейс – https://www.internet-technologies.ru/articles/newbie/polzovatelskiy-interfeys.html
29. Т. Мандел. Разработка пользовательского интерфейса – https://books.google.com.ua/books?id=rSq5DwAAQBAJ&printsec=frontcover&hl=ru&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false
30. C# Windows Forms Application Tutorial with Example – https://www.guru99.com/c-sharp-windows-forms-application.html
31. Руководство по программированию в Windows Forms – https://metanit.com/sharp/windowsforms/
32. Пахомов Б.И. C# для начинающих. – СПб.: БХВ-Петербург, 2014. – 432 с.
33. MySQL Connector/NET Developer Guide – https://dev.mysql.com/doc/connector-net/en/

34. Introduction to SQL – https://www.studytonight.com/dbms/introduction-to-sql.php
35. Microsoft.Office.Interop.Word Namespace – https://docs.microsoft.com/en-us/dotnet/api/microsoft.office.interop.word?view=word-pia
36. How to merge several cells into a single cell and split one cell into several cells in C#? – https://www.e-iceblue.com/Tutorials/Spire.Doc/Spire.Doc-Program-Guide/Table/How-to-merge-several-cells-into-a-single-cell-and-split-one-cell-into-several-cells-in-C.html

ДОДАТОК А

Лістинг скрипту створення бази даних

```
-- MySQL Script generated by MySQL Workbench
-- Thu Nov 12 21:22:29 2020
-- Model: New Model  Version: 1.0
-- MySQL Workbench Forward Engineering

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET                                                    @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ER
ROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION';

-- -----------------------------------------------------
-- Schema dek
-- -----------------------------------------------------
CREATE  SCHEMA  IF  NOT  EXISTS  `dek`  DEFAULT  CHARACTER  SET  utf8mb4  COLLATE
utf8mb4_0900_ai_ci ;
USE `dek` ;

-- -----------------------------------------------------
-- Table `dek`.`character_answer`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `dek`.`character_answer` (
 `id_choa` INT UNSIGNED NOT NULL AUTO_INCREMENT,
 `name_of_answer` VARCHAR(25) NOT NULL,
 PRIMARY KEY (`id_choa`))
ENGINE = InnoDB
AUTO_INCREMENT = 6
DEFAULT CHARACTER SET = utf8;

-- -----------------------------------------------------
-- Table `dek`.`commission_number`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `dek`.`commission_number` (
 `id_commission` INT UNSIGNED NOT NULL AUTO_INCREMENT,
 `number_of_DEK` VARCHAR(45) NOT NULL,
 PRIMARY KEY (`id_commission`))
ENGINE = InnoDB
AUTO_INCREMENT = 2
DEFAULT CHARACTER SET = utf8;

-- -----------------------------------------------------
-- Table `dek`.`members`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `dek`.`members` (
 `id_member` INT UNSIGNED NOT NULL AUTO_INCREMENT,
 `surname` VARCHAR(25) NOT NULL,
 `name` VARCHAR(25) NOT NULL,
 `middle_name` VARCHAR(25) NOT NULL,
 `job` VARCHAR(255) NULL DEFAULT NULL,
 `post` VARCHAR(50) NULL DEFAULT NULL,
 `science_degree` VARCHAR(50) NULL DEFAULT NULL,
 `academic_title` VARCHAR(50) NULL DEFAULT NULL,
 `head` TINYINT(1) NULL DEFAULT NULL,
 `reviewer` TINYINT(1) NULL DEFAULT NULL,
 `DEK_member` TINYINT(1) NULL DEFAULT NULL,
 `consultant` TINYINT(1) NULL DEFAULT NULL,
 PRIMARY KEY (`id_member`))
ENGINE = InnoDB
```

```
AUTO_INCREMENT = 13
DEFAULT CHARACTER SET = utf8;

-- -----------------------------------------------------
-- Table `dek`.`meeting_present`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `dek`.`meeting_present` (
 `id_meeting_present` INT UNSIGNED NOT NULL AUTO_INCREMENT,
 `date` DATE NOT NULL,
 `start_time` TIME NULL DEFAULT NULL,
 `end_time` TIME NULL DEFAULT NULL,
 `id_chief` INT UNSIGNED NULL DEFAULT NULL,
 `id_present_1` INT UNSIGNED NULL DEFAULT NULL,
 `id_present_2` INT UNSIGNED NULL DEFAULT NULL,
 `id_present_3` INT UNSIGNED NULL DEFAULT NULL,
 `id_present_4` INT UNSIGNED NULL DEFAULT NULL,
 `id_present_5` INT UNSIGNED NULL DEFAULT NULL,
 `id_present_6` INT UNSIGNED NULL DEFAULT NULL,
 `id_present_7` INT UNSIGNED NULL DEFAULT NULL,
 PRIMARY KEY (`id_meeting_present`),
 INDEX `fk_protocol_DEK1_idx` (`id_chief` ASC) VISIBLE,
 INDEX `fk_protocol_DEK2_idx` (`id_present_1` ASC) VISIBLE,
 INDEX `fk_protocol_DEK3_idx` (`id_present_2` ASC) VISIBLE,
 INDEX `fk_protocol_DEK4_idx` (`id_present_3` ASC) VISIBLE,
 INDEX `fk_protocol_DEK7_idx` (`id_present_4` ASC) VISIBLE,
 INDEX `fk_protocol_DEK8_idx` (`id_present_5` ASC) VISIBLE,
 INDEX `fk_protocol_DEK9_idx` (`id_present_6` ASC) VISIBLE,
 INDEX `fk_protocol_DEK10_idx` (`id_present_7` ASC) VISIBLE,
 CONSTRAINT `fk_protocol_DEK1`
  FOREIGN KEY (`id_chief`)
  REFERENCES `dek`.`members` (`id_member`),
 CONSTRAINT `fk_protocol_DEK10`
  FOREIGN KEY (`id_present_7`)
  REFERENCES `dek`.`members` (`id_member`),
 CONSTRAINT `fk_protocol_DEK2`
  FOREIGN KEY (`id_present_1`)
  REFERENCES `dek`.`members` (`id_member`),
 CONSTRAINT `fk_protocol_DEK3`
  FOREIGN KEY (`id_present_2`)
  REFERENCES `dek`.`members` (`id_member`),
 CONSTRAINT `fk_protocol_DEK4`
  FOREIGN KEY (`id_present_3`)
  REFERENCES `dek`.`members` (`id_member`),
 CONSTRAINT `fk_protocol_DEK7`
  FOREIGN KEY (`id_present_4`)
  REFERENCES `dek`.`members` (`id_member`),
 CONSTRAINT `fk_protocol_DEK8`
  FOREIGN KEY (`id_present_5`)
  REFERENCES `dek`.`members` (`id_member`),
 CONSTRAINT `fk_protocol_DEK9`
  FOREIGN KEY (`id_present_6`)
  REFERENCES `dek`.`members` (`id_member`))
ENGINE = InnoDB
AUTO_INCREMENT = 2
DEFAULT CHARACTER SET = utf8;

-- -----------------------------------------------------
-- Table `dek`.`meeting_head`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `dek`.`meeting_head` (
 `id_meeting_head` INT UNSIGNED NOT NULL AUTO_INCREMENT,
 `rate_number_of_protocol` VARCHAR(25) NOT NULL,
 `type_of_protocol` VARCHAR(45) NULL DEFAULT NULL,
```

```
 `id_meeting_present` INT UNSIGNED NULL DEFAULT NULL,
 PRIMARY KEY (`id_meeting_head`),
 INDEX    `fk_meeting_head_meeting_present1_idx`    (`id_meeting_present`    ASC)
VISIBLE,
 CONSTRAINT `fk_meeting_head_meeting_present1`
  FOREIGN KEY (`id_meeting_present`)
  REFERENCES `dek`.`meeting_present` (`id_meeting_present`))
ENGINE = InnoDB
AUTO_INCREMENT = 2
DEFAULT CHARACTER SET = utf8;

-- -----------------------------------------------------
-- Table `dek`.`qualification`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `dek`.`qualification` (
 `id_qualification` INT UNSIGNED NOT NULL AUTO_INCREMENT,
 `skill_level` VARCHAR(15) NOT NULL,
 `name_of_specialty` VARCHAR(80) NULL DEFAULT NULL,
 `code_specialty` VARCHAR(25) NULL DEFAULT NULL,
 `qualification_of_student` VARCHAR(225) NULL DEFAULT NULL,
 PRIMARY KEY (`id_qualification`))
ENGINE = InnoDB
AUTO_INCREMENT = 3
DEFAULT CHARACTER SET = utf8;

-- -----------------------------------------------------
-- Table `dek`.`student`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `dek`.`student` (
 `id_student` INT UNSIGNED NOT NULL AUTO_INCREMENT,
 `surname` VARCHAR(25) NOT NULL,
 `name` VARCHAR(25) NOT NULL,
 `middle_name` VARCHAR(25) NOT NULL,
 `topic` VARCHAR(255) NULL DEFAULT NULL,
 `group` VARCHAR(25) NULL DEFAULT NULL,
 `form_education` VARCHAR(25) NULL DEFAULT NULL,
 `id_head` INT UNSIGNED NULL DEFAULT NULL,
 `id_reviewer` INT UNSIGNED NULL DEFAULT NULL,
 `id_consultant` INT UNSIGNED NULL DEFAULT NULL,
 `id_qualification` INT UNSIGNED NOT NULL,
 PRIMARY KEY (`id_student`),
 INDEX `fk_student_members1_idx` (`id_head` ASC) VISIBLE,
 INDEX `fk_student_members2_idx` (`id_reviewer` ASC) VISIBLE,
 INDEX `fk_student_members3_idx` (`id_consultant` ASC) VISIBLE,
 INDEX `fk_student_qualification1_idx` (`id_qualification` ASC) VISIBLE)
ENGINE = InnoDB
AUTO_INCREMENT = 3
DEFAULT CHARACTER SET = utf8;

-- -----------------------------------------------------
-- Table `dek`.`report_of_protect`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `dek`.`report_of_protect` (
 `student_id_student` INT UNSIGNED NOT NULL,
 `id_meeting_head` INT UNSIGNED NULL DEFAULT NULL,
 `rating_by_head` VARCHAR(25) NULL DEFAULT NULL,
 `rating_by_reviewer` VARCHAR(25) NULL DEFAULT NULL,
 `rating_of_publication` VARCHAR(25) NULL DEFAULT NULL,
 `rating_of_work_level` VARCHAR(25) NULL DEFAULT NULL,
 `rating_of_presentation` VARCHAR(25) NULL DEFAULT NULL,
 `insignia` TINYINT(1) NULL DEFAULT NULL,
 `postgraduate` TINYINT(1) NULL DEFAULT NULL,
 `english` TINYINT(1) NULL DEFAULT NULL,
```

```
 `question_1` INT UNSIGNED NULL DEFAULT NULL,
 `question_text_1` VARCHAR(255) NULL DEFAULT NULL,
 `question_2` INT UNSIGNED NULL DEFAULT NULL,
 `question_text_2` VARCHAR(255) NULL DEFAULT NULL,
 `question_3` INT UNSIGNED NULL DEFAULT NULL,
 `question_text_3` VARCHAR(255) NULL DEFAULT NULL,
 `question_4` INT UNSIGNED NULL DEFAULT NULL,
 `question_text_4` VARCHAR(255) NULL DEFAULT NULL,
 `question_5` INT UNSIGNED NULL DEFAULT NULL,
 `question_text_5` VARCHAR(255) NULL DEFAULT NULL,
 `rating_vnz` INT UNSIGNED NULL DEFAULT NULL,
 `rating_national` VARCHAR(25) NULL DEFAULT NULL,
 `rating_ECTS` VARCHAR(1) NULL DEFAULT NULL,
 PRIMARY KEY (`student_id_student`),
 INDEX `fk_report_protect_student1_idx` (`student_id_student` ASC) VISIBLE,
 INDEX `fk_report_protect_DEK1_idx` (`question_1` ASC) VISIBLE,
 INDEX `fk_report_protect_DEK2_idx` (`question_2` ASC) VISIBLE,
 INDEX `fk_report_protect_DEK3_idx` (`question_3` ASC) VISIBLE,
 INDEX `fk_report_protect_DEK4_idx` (`question_4` ASC) VISIBLE,
 INDEX `fk_report_protect_DEK5_idx` (`question_5` ASC) VISIBLE,
 INDEX `fk_report_of_protect_meeting_head1_idx` (`id_meeting_head` ASC) VISIBLE,
 CONSTRAINT `fk_report_of_protect_meeting_head1`
  FOREIGN KEY (`id_meeting_head`)
  REFERENCES `dek`.`meeting_head` (`id_meeting_head`),
 CONSTRAINT `fk_report_protect_DEK1`
  FOREIGN KEY (`question_1`)
  REFERENCES `dek`.`members` (`id_member`),
 CONSTRAINT `fk_report_protect_DEK2`
  FOREIGN KEY (`question_2`)
  REFERENCES `dek`.`members` (`id_member`),
 CONSTRAINT `fk_report_protect_DEK3`
  FOREIGN KEY (`question_3`)
  REFERENCES `dek`.`members` (`id_member`),
 CONSTRAINT `fk_report_protect_DEK4`
  FOREIGN KEY (`question_4`)
  REFERENCES `dek`.`members` (`id_member`),
 CONSTRAINT `fk_report_protect_DEK5`
  FOREIGN KEY (`question_5`)
  REFERENCES `dek`.`members` (`id_member`),
 CONSTRAINT `fk_report_protect_student1`
  FOREIGN KEY (`student_id_student`)
  REFERENCES `dek`.`student` (`id_student`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8;

-- -----------------------------------------------------
-- Table `dek`.`valuation`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `dek`.`valuation` (
 `id_valuation` INT NOT NULL AUTO_INCREMENT,
 `national_value` VARCHAR(45) NOT NULL,
 `ECTS_value` VARCHAR(45) NULL DEFAULT NULL,
 `min` INT NULL DEFAULT NULL,
 `max` INT NULL DEFAULT NULL,
 PRIMARY KEY (`id_valuation`))
ENGINE = InnoDB
AUTO_INCREMENT = 8
DEFAULT CHARACTER SET = utf8;

SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

ДОДАТОК Б

Лістинги основних модулів

Програмний код

```
using System;
using MySql.Data.MySqlClient;
using MySql.Data;
using System.Collections.Generic;
using Microsoft.Office;
using Excel = Microsoft.Office.Interop.Excel; //Microsoft Office using
/* Import - class, which contain methods for import data from excel files to
MySQL tables.
 *   methods:   connectToDatabase(string   con_str),   disconnectFromDatabase(),
fromExcelToParList(string excelPath)
 */
public class Import
{
  // fields
  public Excel.Application excelapp; // excell app object
  private static string constr = ""; // connection string
  private static int count = 0; // count of excel rows
  public  static  MySqlConnection  Con  =  new  MySqlConnection();  //  create  of
connection
  public static MySqlCommand Query = new MySqlCommand(); // object for queries
  public  List<Participants>  parList  =  new  List<Participants>();  //  list  of
Participants members
  public List<Students> studList = new List<Students>(); // list of Students
  public List<Members> membList = new List<Members>(); // list of Members
  // properties
  static int Count
  {
    get{
      return count;
    }
    set{
      count = value;
    }
  }
  // constructors
  public Import() // defualt constructor
  {
  }
  // methods
  public  static  void  connectToDatabase(string  con_str)  //  create  connect  to
Database
  {
      Con = new MySqlConnection(con_str);
      Import.constr = con_str;
    Query.Connection = Con;
    try{
      Con.Open();// Connect
    }
    catch (MySqlException SSDB_Exception){
      // Error, return
      return;
    }
  }
  public static void disconnectFromDatabase() // disconnect from Database
  {
      Con.Close();
```

```
      Query.Dispose();
  }
  public static int getId_headOfStud(object surname, object name, object
mid_name) // method return id_head for student object
  {
    int id = -1;
    if ((surname != null) && (name != null) && (mid_name != null)){
      Query.CommandText = "SELECT id_member FROM members WHERE "
                  + "head = 1"
                  + " AND surname ='" + Convert.ToString(surname)
                  + "' AND name = '" + Convert.ToString(name)
                  + "' AND middle_name = '" + Convert.ToString(mid_name) + "';";
      MySqlDataReader MyReader = Query.ExecuteReader();
      if (MyReader.HasRows)
      {
        if (MyReader.Read())
        {
          id = MyReader.GetInt32(0);
          MyReader.Close();
          return id;
        }
      }
      else
      {
        MyReader.Close();
        return id;
      }
      MyReader.Close();
    }
    return id;
  }
  public static int getId_reviewerOfStud(object surname, object name, object
mid_name) // method return id_reviwer for student object
  {
    int id = -1;
    if ((surname != null) && (name != null) && (mid_name != null)){
      Query.CommandText = "SELECT id_member FROM members WHERE "
                  + "reviewer = 1"
                  + " AND surname ='" + Convert.ToString(surname)
                  + "' AND name = '" + Convert.ToString(name)
                  + "' AND middle_name = '" + Convert.ToString(mid_name) + "';";
      MySqlDataReader MyReader = Query.ExecuteReader();
      if (MyReader.HasRows)
      {
        if (MyReader.Read())
        {
          id = MyReader.GetInt32(0);
          MyReader.Close();
          return id;
        }
      }
      else
      {
        MyReader.Close();
        return id;
      }
      MyReader.Close();
    }
    return id;
  }
  public static int getId_consultant(object surname, object name, object
mid_name) // method return id_consultant_oxranu_truda for student object
  {
```

```
    int id = -1;
    if ((surname != null) && (name != null) && (mid_name != null)){
      Query.CommandText = "SELECT id_member FROM members WHERE "
                  + "consultant = 1"
                  + " AND surname ='" + Convert.ToString(surname)
                  + "' AND name = '" + Convert.ToString(name)
                  + "' AND middle_name = '" + Convert.ToString(mid_name) + "';";
      MySqlDataReader MyReader = Query.ExecuteReader();
      if (MyReader.HasRows)
      {
        if (MyReader.Read())
        {
          id = MyReader.GetInt32(0);
          MyReader.Close();
          return id;
        }
      }
      else
      {
        MyReader.Close();
        return id;
      }
      MyReader.Close();
    }
    return id;
  }

  public static int getId_consultant_it_projectOfStud(object surname, object
name, object mid_name) // method return id_consultant_it_project for student
object
  {
    int id = -1;
    if ((surname != null) && (name != null) && (mid_name != null)){
      Query.CommandText = "SELECT id_member FROM members WHERE "
                  + "consultant = 1"
                  + " AND surname ='" + Convert.ToString(surname)
                  + "' AND name = '" + Convert.ToString(name)
                  + "' AND middle_name = '" + Convert.ToString(mid_name) + "';";
      MySqlDataReader MyReader = Query.ExecuteReader();
      if (MyReader.HasRows)
      {
        if (MyReader.Read())
        {
          id = MyReader.GetInt32(0);
          MyReader.Close();
          return id;
        }
      }
      else
      {
        MyReader.Close();
        return id;
      }
      MyReader.Close();
    }
    return id;
  }
  public void fromExcelToParList(string excelPath) // import data from excel
file into parList
  {
    Excel.Application excelapp = new Excel.Application();
    excelapp.DisplayAlerts = false; // alerts turn off
    excelapp.Visible = true; // excel will appear
```

```
    excelapp.Workbooks.Open(@excelPath);// open file by path
    excelapp.Workbooks.get_Item(1); // get first book
    Excel.Worksheet sheet = (Excel.Worksheet)excelapp.ActiveSheet;
    // find count of filled rows in excel
    Import.Count                        =                       sheet.get_Range("A1",
Type.Missing).get_End(Microsoft.Office.Interop.Excel.XlDirection.xlDown).Special
Cells(Microsoft.Office.Interop.Excel.XlCellType.xlCellTypeLastCell,
Type.Missing).Row - 2;
    for (int i = 1; i < Import.Count; i++){
      int j = 1;
Excel.Range).Value);
      j = 1;
      this.parList.Add(new  Import.Participants((string)(sheet.Cells[i,  j++]  as
Excel.Range).Value,   (string)(sheet.Cells[i,   j++]   as   Excel.Range).Value,
(string)(sheet.Cells[i, j] as Excel.Range).Value));
    }
    excelapp.Quit();
  }
  public void fromParListToDB() // import data from parList of Participants into
Database
  {
    Query.CommandText = "START TRANSACTION;";
    Query.ExecuteNonQuery();
    for (int i = 0; i < this.parList.Count; i++){
      Console.WriteLine(i + 1 +") upload row: " + this.parList[i].Surname + ", "
+ this.parList[i].Name + ", " + this.parList[i].Mid_name);
      Query.CommandText  =  "INSERT  INTO  "  +  "student"  +  "(surname,  name,
middle_name,  id_qualification,  `year`)  VALUES('"  +  this.parList[i].Surname  +
"','" + this.parList[i].Name + "','" + this.parList[i].Mid_name + "','" + '1' +
"'); ";
      Query.ExecuteNonQuery();
    }
    Query.CommandText = "COMMIT;";
    Query.ExecuteNonQuery();
  }
  public  void  fromExcelToStudList(string  excelPath)  //  import  data  from  excel
file into studList of Students
  {
    Excel.Application excelapp = new Excel.Application();
    excelapp.DisplayAlerts = false; // alerts turn off
    excelapp.Visible = true; // excel will appear
    excelapp.Workbooks.Open(@excelPath);// open file by path
    Console.WriteLine(" -- " + excelPath + " is opened --");
    excelapp.Workbooks.get_Item(1); // get first book
    Excel.Worksheet sheet = (Excel.Worksheet)excelapp.ActiveSheet;
    // find count of filled rows in excel
    Import.Count                        =                       sheet.get_Range("A1",
Type.Missing).get_End(Microsoft.Office.Interop.Excel.XlDirection.xlDown).Special
Cells(Microsoft.Office.Interop.Excel.XlCellType.xlCellTypeLastCell,Type.Missing)
.Row - 2;
      j = 1;
      this.studList.Add(new Import.Students((string)(sheet.Cells[i + 3, j++] as
Excel.Range).Value, // surname
 (string)(sheet.Cells[i + 3, j++] as Excel.Range).Value, // name
 (string)(sheet.Cells[i + 3, j++] as Excel.Range).Value, // middle name
 (string)(sheet.Cells[i + 3, j++] as Excel.Range).Value, // id_head suraname;
 (string)(sheet.Cells[i + 3, j++] as Excel.Range).Value, // id_head name;
 (string)(sheet.Cells[i + 3, j++] as Excel.Range).Value, // id_head middle name;
 (string)(sheet.Cells[i + 3, j++] as Excel.Range).Value, // id_re suraname;
 (string)(sheet.Cells[i + 3, j++] as Excel.Range).Value, // id_re name;
 (string)(sheet.Cells[i + 3, j++] as Excel.Range).Value, // id_re middle name;
 (string)(sheet.Cells[i + 3, j++] as Excel.Range).Value, // id_op suraname;
 (string)(sheet.Cells[i + 3, j++] as Excel.Range).Value, // id_op name;
```

```
 (string)(sheet.Cells[i + 3, j++] as Excel.Range).Value, // id_op middle name;
 (string)(sheet.Cells[i + 3, j++] as Excel.Range).Value, // id_ec suraname;
 (string)(sheet.Cells[i + 3, j++] as Excel.Range).Value, // id_ec name;
 (string)(sheet.Cells[i + 3, j++] as Excel.Range).Value, // id_ec middle name;
 (string)(sheet.Cells[i + 3, j++] as Excel.Range).Value, // id_it suraname;
 (string)(sheet.Cells[i + 3, j++] as Excel.Range).Value, // id_it name;
 (string)(sheet.Cells[i + 3, j++] as Excel.Range).Value, // id_it middle name;
 (string)(sheet.Cells[i + 3, j++] as Excel.Range).Value, // topic
 (string)(sheet.Cells[i + 3, j++] as Excel.Range).Value, // group
 (string)(sheet.Cells[i + 3, j++] as Excel.Range).Value, // form education
 (sheet.Cells[i + 3, j++] as Excel.Range).Value, // id of qualification
 (sheet.Cells[i + 3, j] as Excel.Range).Value)); // year of education start
    }
    excelapp.Quit();
  }
  public void fromStudListToDB() // import data from studList of Students into
Database
  {
    Query.CommandText = "START TRANSACTION;";
    Query.ExecuteNonQuery();
    for (int i = 0; i < this.studList.Count; i++){
      Query.CommandText = "INSERT INTO " + "student" +
"(surname, name, middle_name, id_head, id_reviewer, id_consultant, topic,
`group`, form_education, id_qualification) VALUES('"+ this.studList[i].Surname +
"','"+ this.studList[i].Name + "','"
+ this.studList[i].Mid_name + "',";
      if (this.studList[i].Id_head != -1)
        Query.CommandText = Query.CommandText + " '" + this.studList[i].Id_head
+ "', ";
      else
        Query.CommandText = Query.CommandText + " null, ";
      if (this.studList[i].Id_reviewer != -1)
        Query.CommandText = Query.CommandText + " '" +
this.studList[i].Id_reviewer + "', ";
      else
        Query.CommandText = Query.CommandText + " null, ";
      if (this.studList[i].Id_op != -1)
        Query.CommandText = Query.CommandText + " '" + this.studList[i].Id_op +
"', ";
      else
        Query.CommandText = Query.CommandText + " null, ";
      if (this.studList[i].Id_ec != -1)
        Query.CommandText = Query.CommandText + " '" + this.studList[i].Id_ec +
"', ";
      else
        Query.CommandText = Query.CommandText + " null, ";
      if (this.studList[i].Id_it != -1)
        Query.CommandText = Query.CommandText + " '" + this.studList[i].Id_it +
"','";
      else
        Query.CommandText = Query.CommandText + " null, '";
      Query.CommandText = Query.CommandText
+   this.studList[i].Topic   +   "','"+   this.studList[i].Group   +   "','"+
this.studList[i].Form_educ + "','"+ this.studList[i].Id_qual + "');";
      Query.ExecuteNonQuery();
    }
    Query.CommandText = "COMMIT;";
    Query.ExecuteNonQuery();
    Con.Close();
  }
  public void fromExcelToMembList(string excelPath) // import data from excel
file into studList of Students
  {
```

```
    Excel.Application excelapp = new Excel.Application();
    excelapp.DisplayAlerts = false; // alerts turn off
    excelapp.Visible = true; // excel will appear
    excelapp.Workbooks.Open(excelPath);// open file by path
    excelapp.Workbooks.get_Item(1); // get first book
    Excel.Worksheet  sheet  =  (Excel.Worksheet)excelapp.Sheets.Item[1];  //  get
first list
    // find count of filled rows in excel
    Import.Count                        =                        sheet.get_Range("A1",
Type.Missing).get_End(Microsoft.Office.Interop.Excel.XlDirection.xlDown).Special
Cells(Microsoft.Office.Interop.Excel.XlCellType.xlCellTypeLastCell,
Type.Missing).Row - 2;
      j = 1;
this.membList.Add(new   Import.Members((string)(sheet.Cells[i   +   3,   j++]   as
Excel.Range).Value, // surname
 (string)(sheet.Cells[i + 3, j++] as Excel.Range).Value, // name
 (string)(sheet.Cells[i + 3, j++] as Excel.Range).Value, // middle name
 (sheet.Cells[i + 3, j++] as Excel.Range).Value, // job
 (sheet.Cells[i + 3, j++] as Excel.Range).Value, // post in job
 (sheet.Cells[i + 3, j++] as Excel.Range).Value, // science degree
 (sheet.Cells[i + 3, j++] as Excel.Range).Value, // academic title
 (sheet.Cells[i + 3, j++] as Excel.Range).Value, // head of student
 (sheet.Cells[i + 3, j++] as Excel.Range).Value, // reviewer of student
 (sheet.Cells[i + 3, j++] as Excel.Range).Value, // DEK member
 (sheet.Cells[i + 3, j++] as Excel.Range).Value)); // consultant
    }
    excelapp.Quit();
  }
  public void fromMembListToDB() // import data from MemdList of Students into
Database
  {
    Query.CommandText = "START TRANSACTION;";
    Query.ExecuteNonQuery();
    for (int i = 0; i < this.membList.Count; i++){
      Console.WriteLine(i + 1 + ") upload row: " + this.membList[i].Surname + ",
" + this.membList[i].Name + ", " + this.membList[i].Mid_name);
      Query.CommandText = "INSERT INTO " + "members" +
"(surname, name, middle_name, job, post, science_degree, academic_title, head,
reviewer, DEK_member, consultant) VALUES('"+ this.membList[i].Surname + "','"
+   this.membList[i].Name   +   "','"+   this.membList[i].Mid_name   +   "','"+
this.membList[i].Job     +     "','"+     this.membList[i].Post     +     "','"+
this.membList[i].Sc_degree   +   "','"+   this.membList[i].Ac_title   +   "','"+
this.membList[i].Head    +    "','"+    this.membList[i].Reviewer    +    "','"+
this.membList[i].Dek_mem + "','"+ this.membList[i].Consultant + "');";
      Query.ExecuteNonQuery();
    }
    Query.CommandText = "COMMIT;";
    Query.ExecuteNonQuery();
  }
  // classes
  /* Participants - Participants of event,
   * fields: surname, name, mid_name;
   * properties: Surname(), Name(), Mid_name();
   * constructors: Participant(..);
   * methods: genInfo();
   */
  public class Participants
  {
    // fields
    protected string surname = null; // surname
    protected string name = null;  // name
    protected string mid_name = null; // middle name
    // properties
```

```
    public string Surname // Surname property rw{
      get{        return surname;      }
      set{        surname = value;      }
    }
    public string Name // name property rw{
      get{        return name;      }
      set{        name = value;      }
    }
    public string Mid_name // middle name property rw{
      get{        return mid_name;      }
      set{        mid_name = value;      }
    }
    // constructors
    public  Participants(object  surname,  object  name,  object  mid_name)  //
constructor{
      this.Surname = Convert.ToString(surname);
      this.Name = Convert.ToString(name);
      this.Mid_name = Convert.ToString(mid_name);
    }
    }
  }
  /* Students - students of event, it extends from Participants class
  * fields: id_head, topic, group, form_educ, year;
  * properties: Id_head, Topic, Group, Form_educ, Year;
  * constructors: Students(..);
  * methods: getInfo();
  */
  public sealed class Students : Participants
  {
    // fields
    private int id_head = -1; // id of head of student
    private int id_reviewer = -1; // id of reviewer of student
    private int id_op = -1; // id of op of student
    private int id_ec = -1; // id of ec of student
    private int id_it = -1; // id of it of student
    private string topic = null; // topic name of work
    private string group = null; // group
    private string form_educ = null; // form education
    private ushort id_qual = 0; // id of qualification

    // properties
    public int Id_head // Id of head of student property rw{
      get{        return id_head;      }
      set{        id_head = value;      }
    }
    public int Id_reviewer // Id of head of student property rw{
      get{        return id_reviewer;      }
      set{        id_reviewer = value;      }
    }
    public int Id_op // Id of head of student property rw{
      get{        return id_op;      }
      set{        id_op = value;      }
    }
    public string Topic // Topic name of work property rw{
      get{        return topic;      }
      set{        topic = value;      }
    }
    public string Group // Group property rw{
      get{        return group;      }
      set{        group = value;      }
    }
    public string Form_educ // Form education property rw{
      get{        return form_educ;      }
```

```
      set{        form_educ = value;      }
    }
    public ushort Id_qual // id of qualification rw{
      get{        return id_qual;      }
      set{        id_qual = value;      }
    }

    // constructors
    public Students(object surname, object name, object mid_name,
object head_surname, object head_name, object head_mid_name,
object rev_surname, object rev_name, object rev_mid_name,
object op_surname, object op_name, object op_mid_name,
object topic, object group, object form_educ, object id_qual)
           : base(surname, name, mid_name) // constructor{
      this.Id_head    =    Convert.ToInt32(Import.getId_headOfStud(head_surname,
head_name, head_mid_name));
      this.Id_reviewer                                                    =
Convert.ToInt32(Import.getId_reviewerOfStud(rev_surname,               rev_name,
rev_mid_name));
      this.Id_op  =  Convert.ToInt32(Import.getId_consultant(op_surname,  op_name,
op_mid_name));
      this.Topic = Convert.ToString(topic); // topic name of work
      this.Group = Convert.ToString(group);  // group
      this.Form_educ = Convert.ToString(form_educ); // form education
      this.Id_qual = Convert.ToUInt16(id_qual);
    }
  }
  /* Members - members of event, it extends from Participants class
   *  fields:  job,  post,  sc_degree,  ac_title,  head,  reviewer,  dek_mem,
consultant;
   *  properties:  Job,  Post,  Sc_degree,  Ac_title,  Head,  Reviewer,  DEK_mem,
Consultant;;
   * constructors: Students(..);
   * methods: getinfo();
   */
  public sealed class Members : Participants
  {
    // fields
    private string job = null; // job
    private string post = null;  // post in job
    private string sc_degree = null; // science degree
    private string ac_title = null; // academic title
    private string head = null; // head of student
    private string reviewer = null; // reviewer of student
    private string dek_mem = null; // DEK member
    private string consultant = null;
    // properties
    public string Job // Topic name of work property rw{
      get{        return job;      }
      set{        job = value;      }
    }
    public string Post // Post in job property rw{
      get{        return post;      }
      set{        post = value;      }
    }
    public string Sc_degree // Science degree property rw{
      get{        return sc_degree;      }
      set{        sc_degree = value;      }
    }
    public string Ac_title // Academic title property rw{
      get{        return ac_title;      }
      set{        ac_title = value;      }
    }
```

```
    public string Head // Science degree property rw{
      get{        return head;      }
      set{          head = value;      }
    }
    public string Reviewer // Reviewer property rw{
      get{        return reviewer;      }
      set{        reviewer = value;      }
    }
    public string Dek_mem // DEk member property rw{
      get{        return dek_mem;      }
      set{         dek_mem = value;       }
    }
    public string Consultant // Consultant property rw{
      get{        return consultant;      }
      set{        consultant = value;      }
    }
    // constructors
    public Members(object surname, object name, object mid_name, object job,
object post, object sc_degree, object ac_title, object head, object reviewer,
object dek_mem, object consultant)
      : base(surname, name, mid_name) // constructor{
      this.Job = Convert.ToString(job);
      this.Post = Convert.ToString(post);
      this.Sc_degree = Convert.ToString(sc_degree);
      this.Ac_title = Convert.ToString(ac_title);
      this.Head = Convert.ToString(head);
      this.Reviewer = Convert.ToString(reviewer);
      this.Dek_mem = Convert.ToString(dek_mem);
      this.Consultant = Convert.ToString(consultant);
    }
    // methods
    }
}
```

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using MySql.Data.MySqlClient;
using MySql.Data;
using System.Data.SqlTypes;
using System.IO;
using Microsoft.Office;
//using Microsoft.Office.Interop.Word._Application;

namespace dek
{
  public partial class mainFormDEK : Form
  {
    public mainFormDEK()
    {
      InitializeComponent();
    }

    Microsoft.Office.Interop.Word.Document WordDoc;
    Microsoft.Office.Interop.Word.Application WordApp;
    Microsoft.Office.Interop.Word.Find WordFind;
```

```
    Microsoft.Office.Interop.Word.Range WordRange;

    string Connect = "";
    MySqlDataAdapter dataAdapter;
    DataTable dataTable;
    MySqlConnection mySQLConnection;
    MySqlCommand mySQLCommand;

    //DateTime Time; //глобальна змінна часу

    private void dekForm_Load(object sender, EventArgs e)
    {
      FileInfo file = new FileInfo("ConnectString.ini");
      if (file.Exists != true) //Если файл существует
      {
        MessageBox.Show("Файлу із строкою підключення не існує! Програма не буде
запущена.");
        Environment.Exit(0);
      }
      StreamReader streamReader = new StreamReader("ConnectString.ini");
      Connect = streamReader.ReadToEnd();
      streamReader.Close();
      try
      {
        using (MySqlConnection sqlConn =
          new MySqlConnection(Connect))
        {
          sqlConn.Open();
          sqlConn.Close();
        }
      }
      catch (MySqlException)
      {
        MessageBox.Show("Відсутній зв'язок із базою даних, програма не буде
запущена!"); Environment.Exit(0);
      }
      catch (Exception)
      {
        MessageBox.Show("Відсутній зв'язок із базою даних, програма не буде
запущена!"); Environment.Exit(0);
      }
      dateProtection.Text          =          dateQualification.Text          =
DateTime.Today.ToString("yyyy-MM-dd");
      connectCreate();

    }

    private void connectCreate()
    {

      mySQLConnection = new MySqlConnection(Connect);
      mySQLCommand = new MySqlCommand();
      mySQLCommand.Connection = mySQLConnection;
    }

    private void headProtection_DropDown(object sender, EventArgs e)
    {
      try
      {
        mySQLConnection.Open();
        mySQLConnection.Close();
      }
      catch (MySqlException)
```

```
      {
        MessageBox.Show("Перевірте зв'язок із базою даних");
        return;
      }
      dataTable = new DataTable();
      mySQLConnection.Open();
      mySQLCommand.CommandText = "SELECT CONCAT( surname, ' ', name) as pib FROM
members;";
      dataAdapter = new MySqlDataAdapter();
      dataAdapter.SelectCommand = mySQLCommand;
      mySQLConnection.Close();
      dataTable.Clear();
      dataAdapter.Fill(dataTable);
      (sender as ComboBox).DataSource = dataTable;
      (sender as ComboBox).DisplayMember = "pib";

    }

    private void timeStartProtection_KeyPress(object sender, KeyPressEventArgs
e)
    {

      if (e.KeyChar == 8)
      {
        e.Handled = false;
        return;
      }

      if ((sender as TextBox).SelectionStart == 2)
      {
        (sender as TextBox).Text = (sender as TextBox).Text + ':';
        (sender as TextBox).Select(3, 0);
      }

      if ((sender as TextBox).SelectionStart == 0 && ((e.KeyChar < 48) ||
(e.KeyChar > 50) || (e.KeyChar == 8)))
      {
        e.Handled = true;
      }

      if ((sender as TextBox).SelectionStart == 1 && ((sender as
TextBox).Text[0] == '2') && ((e.KeyChar < 48) || (e.KeyChar > 51)))
      {
        e.Handled = true;
      }

      if (((sender as TextBox).SelectionStart == 1 || (sender as
TextBox).SelectionStart == 4) && ((e.KeyChar < 48) || (e.KeyChar > 57) ||
(e.KeyChar == 8)))
      {
        e.Handled = true;
      }

      if ((sender as TextBox).SelectionStart == 3 && ((e.KeyChar < 48) ||
(e.KeyChar > 53) || (e.KeyChar == 8)))
      {
        e.Handled = true;
      }

      if ((sender as TextBox).SelectionStart >= 5)
        e.Handled = true;
    }
```

```
    private void importFromMembers_Click(object sender, EventArgs e)
    {
      openDialog.ShowDialog();
      openDialog.AddExtension = true;
      openDialog.CheckPathExists = true;
      if (openDialog.FileName == "")
      {
        MessageBox.Show("Файл необраний");
        return;
      }

      Import impMemb = new Import();
      impMemb.fromExcelToMembList(openDialog.FileName);
      Import.connectToDatabase(Connect);
      impMemb.fromMembListToDB();
      Import.disconnectFromDatabase();
    }

    private void importFromStudents_Click(object sender, EventArgs e)
    {
      openDialog.ShowDialog();
      openDialog.AddExtension = true;
      openDialog.CheckPathExists = true;
      if (openDialog.FileName == "")
      {
        MessageBox.Show("Файл необраний");
        return;
      }

      Import impStud = new Import();
      Import.connectToDatabase(Connect);
      impStud.fromExcelToStudList(openDialog.FileName);
      impStud.fromStudListToDB();
      Import.disconnectFromDatabase();
    }

    //занесення інформації про засідання із захисту
    private void insertInfoProtection_Click(object sender, EventArgs e)
    {
      try
      {
        mySQLConnection.Open();
        mySQLConnection.Close();
      }
      catch (MySqlException)
      {
        MessageBox.Show("Перевірте зв'язок із базою даних");
        statusStrip.Items[1].ForeColor = Color.Red;
        statusStrip.Items[1].Text = "Перевірте зв'язок із базою даних";
        return;
      }

      string Temp_Command = "", chief = "", present_1 = "", present_2 = "",
present_3 = "", present_4 = "", present_5 = "", present_6 = "", present_7 = "";
      string Surname = "", Name = "";

      mySQLConnection.Open();
      //Отримання номеру голови засідання
      if (headProtection.Text != "")
      {
        Surname = headProtection.Text.Split(' ')[0];
        Name = headProtection.Text.Split()[1];
```

```
        if (Name == "") { MessageBox.Show("Поле \"Голова\": Перевірте правильність введених даних (можливі зайві пробіли)!"); mySQLConnection.Close(); return; }
        mySQLCommand.CommandText = "SELECT id_member FROM members WHERE surname='" + Surname + "' and name='" + Name + "';";
        MySqlDataReader mySQLReader = mySQLCommand.ExecuteReader();
        if (mySQLReader.HasRows)
        {
          if (mySQLReader.Read())
          {
            chief = mySQLReader.GetString(0);
          }
        }
        mySQLReader.Close();

      }
      else { MessageBox.Show("Обов'язково попинен бути присутнім голова ДЕК!"); mySQLConnection.Close(); return; }

      //Отрмання номеру присутнього 1
      if (presentProtection1.Text != "")
      {
        Surname = presentProtection1.Text.Split(' ')[0];
        Name = presentProtection1.Text.Split()[1];
        if (Name == "") { MessageBox.Show("Поле \"Присутній 1\": Перевірте правильність введених даних (можливі зайві пробіли)!"); mySQLConnection.Close(); return; }
        mySQLCommand.CommandText = "SELECT id_member FROM members WHERE surname='" + Surname + "' and name='" + Name + "';";
        MySqlDataReader mySQLReader = mySQLCommand.ExecuteReader();
        if (mySQLReader.HasRows)
        {
          if (mySQLReader.Read())
          {
            present_1 = mySQLReader.GetString(0);
          }
        }
        mySQLReader.Close();
      }
      else { MessageBox.Show("Мінімум 4 присутніх! Введіть 1го присутнього"); mySQLConnection.Close(); return; }

      //Отрмання номеру присутнього 2
      if (presentProtection2.Text != "")
      {
        Surname = presentProtection2.Text.Split(' ')[0];
        Name = presentProtection2.Text.Split()[1];
        if (Name == "") { MessageBox.Show("Поле \"Присутній 2\": Перевірте правильність введених даних (можливі зайві пробіли)!"); mySQLConnection.Close(); return; }
        mySQLCommand.CommandText = "SELECT id_member FROM members WHERE surname='" + Surname + "' and name='" + Name + "';";
        MySqlDataReader mySQLReader = mySQLCommand.ExecuteReader();
        if (mySQLReader.HasRows)
        {
          if (mySQLReader.Read())
          {
            present_2 = mySQLReader.GetString(0);
          }
        }
        mySQLReader.Close();
      }
```

```
      else { MessageBox.Show("Мінімум 4 присутніх! Введіть 2го присутнього");
mySQLConnection.Close(); return; }

      //Отрмання номеру присутнього 3
      if (presentProtection3.Text != "")
      {
        Surname = presentProtection3.Text.Split(' ')[0];
        Name = presentProtection3.Text.Split()[1];
        if (Name == "") { MessageBox.Show("Поле \"Присутній 3\": Перевірте
правильність введених даних (можливі зайві пробіли)!"); mySQLConnection.Close();
return; }
        mySQLCommand.CommandText = "SELECT id_member FROM members WHERE
surname='" + Surname + "' and name='" + Name + "';";
        MySqlDataReader mySQLReader = mySQLCommand.ExecuteReader();
        if (mySQLReader.HasRows)
        {
          if (mySQLReader.Read())
          {
            present_3 = mySQLReader.GetString(0);
          }
        }
        mySQLReader.Close();
      }
      else { MessageBox.Show("Мінімум 4 присутніх! Введіть 3го присутнього");
mySQLConnection.Close(); return; }

      //Отрмання номеру присутнього 4
      if (presentProtection4.Text != "")
      {
        Surname = presentProtection4.Text.Split(' ')[0];
        Name = presentProtection4.Text.Split()[1];
        if (Name == "") { MessageBox.Show("Поле \"Присутній 4\": Перевірте
правильність введених даних (можливі зайві пробіли)!"); mySQLConnection.Close();
return; }
        mySQLCommand.CommandText = "SELECT id_member FROM members WHERE
surname='" + Surname + "' and name='" + Name + "';";
        MySqlDataReader mySQLReader = mySQLCommand.ExecuteReader();
        if (mySQLReader.HasRows)
        {
          if (mySQLReader.Read())
          {
            present_4 = mySQLReader.GetString(0);
          }
        }
        mySQLReader.Close();
      }
      else { MessageBox.Show("Мінімум 4 присутніх! Введіть 4го присутнього");
mySQLConnection.Close(); return; }

      Temp_Command = "INSERT INTO meeting_present (date, start_time, end_time,
id_chief, id_present_1, id_present_2, id_present_3, id_present_4, id_present_5,
id_present_6, id_present_7) " +
          "values ('" + dateProtection.Text + "','" + timeStartProtection.Text +
"'," + "null" + ",'" + chief + "','" + present_1 + "','" + present_2 + "','" +
present_3 + "','" + present_4 + "',";

      //Отрмання номеру присутнього 5
      if (presentProtection5.Text != "")
      {
        Surname = presentProtection5.Text.Split(' ')[0];
        Name = presentProtection5.Text.Split()[1];
```

```
        if (Name == "") { MessageBox.Show("Поле \"Присутній 5\": Перевірте правильність введених даних (можливі зайві пробіли)!"); mySQLConnection.Close(); return; }
        mySQLCommand.CommandText = "SELECT id_member FROM members WHERE surname='" + Surname + "' and name='" + Name + "';";
        MySqlDataReader mySQLReader = mySQLCommand.ExecuteReader();
        if (mySQLReader.HasRows)
        {
          if (mySQLReader.Read())
          {
            present_5 = mySQLReader.GetString(0);
          }
        }
        mySQLReader.Close();
        Temp_Command = Temp_Command + "'" + present_5 + "',";
      }
      else { Temp_Command = Temp_Command + "null,"; }

      //Отрмання номеру присутнього 6
      if (presentProtection6.Text != "")
      {
        Surname = presentProtection6.Text.Split(' ')[0];
        Name = presentProtection6.Text.Split()[1];
        if (Name == "") { MessageBox.Show("Поле \"Присутній 6\": Перевірте правильність введених даних (можливі зайві пробіли)!"); mySQLConnection.Close(); return; }
        mySQLCommand.CommandText = "SELECT id_member FROM members WHERE surname='" + Surname + "' and name='" + Name + "';";
        MySqlDataReader mySQLReader = mySQLCommand.ExecuteReader();
        if (mySQLReader.HasRows)
        {
          if (mySQLReader.Read())
          {
            present_6 = mySQLReader.GetString(0);
          }
        }
        mySQLReader.Close();
        Temp_Command = Temp_Command + "'" + present_6 + "',";
      }
      else { Temp_Command = Temp_Command + "null,"; }

      //Отрмання номеру присутнього 7
      if (presentProtection7.Text != "")
      {
        Surname = presentProtection7.Text.Split(' ')[0];
        Name = presentProtection7.Text.Split()[1];
        if (Name == "") { MessageBox.Show("Поле \"Присутній 7\": Перевірте правильність введених даних (можливі зайві пробіли)!"); mySQLConnection.Close(); return; }
        mySQLCommand.CommandText = "SELECT id_member FROM members WHERE surname='" + Surname + "' and name='" + Name + "';";
        MySqlDataReader mySQLReader = mySQLCommand.ExecuteReader();
        if (mySQLReader.HasRows)
        {
          if (mySQLReader.Read())
          {
            present_7 = mySQLReader.GetString(0);
          }
        }
        mySQLReader.Close();
        Temp_Command = Temp_Command + "'" + present_7 + "');";
      }
      else { Temp_Command = Temp_Command + "null);"; }
```

```
      //dataTable = new DataTable();
      mySQLCommand.CommandText = Temp_Command;
      mySQLCommand.ExecuteNonQuery();
      mySQLConnection.Close();
      statusStrip.Items[1].ForeColor = Color.Green;
      statusStrip.Items[1].Text = "Запис про засідання додано до бази даних";
      //Time = Convert.ToDateTime(timeStartProtection.Text);
    }

    //Перегляд таблиць
    private void Tables_TextChanged(object sender, EventArgs e)
    {
      switch (Tables.Text)
      {
        case "Студенти":
          mySQLCommand.CommandText = "SELECT " +
"`student`.`id_student` 'ІН студента', " +
"`student`.`surname` 'Прізвище', " +
"`student`.`name` 'Ім`я', " +
"`student`.`middle_name` 'По батькові', " +
"`student`.`topic` 'Тема', " +
"`student`.`group` 'Група', " +
"`student`.`form_education` 'Форма навчання', " +
"`student`.`id_head` 'ІН керівника', " +
"`student`.`id_reviewer` 'ІН рецензента', " +
"`student`.`id_consultant ` 'ІН консл. ', " +
"`student`.`id_qualification` 'ІН Кваліфікації', " +
"FROM student;";
          break;
        case "Члени комісії":
          mySQLCommand.CommandText = "SELECT " +
"`members`.`id_member` 'ІН Викладача', " +
"`members`.`surname` 'Прізвище', " +
"`members`.`name` 'Ім`я', " +
"`members`.`middle_name` 'По батькові', " +
"`members`.`job` 'Робота', " +
"`members`.`post` 'Посада', " +
"`members`.`science_degree` 'Науковий ступінь', " +
"`members`.`academic_title` 'Вчене звання', " +
"`members`.`head` 'Керівник', " +
"`members`.`reviewer` 'Рецензент', " +
"`members`.`DEK_member` 'Член комісії', " +
"`members`.`consultant` 'Консультант' " +
"FROM `dek`.`members`;";
          break;
        case "Характеристики відповідей":
          mySQLCommand.CommandText = "SELECT " +
"id_choa as 'ІН хар-ки відповіді', " +
"name_of_answer as 'Назва хар-ки відповіді' " +
"FROM character_answer;";
          break;
        case "Номери комісій":
          mySQLCommand.CommandText = "SELECT " +
"id_commission as 'ІН номеру комісії', " +
"number_of_DEK as 'Номер комісії' " +
"FROM commission_number;";
          break;
        case "Засідання":
          mySQLCommand.CommandText = "SELECT " +
"id_meeting_present as 'ІН засідання', " +
"date as 'Дата', " +
"start_time as 'Час початку', " +
```

```
"end_time as 'Час закінчення', " +
"id_chief as 'ІН голови', " +
"id_present_1 as 'Присутній 1', " +
"id_present_2 as 'Присутній 2', " +
"id_present_3 as 'Присутній 3', " +
"id_present_4 as 'Присутній 4', " +
"id_present_5 as 'Присутній 5', " +
"id_present_6 as 'Присутній 6', " +
"id_present_7 as 'Присутній 7' " +
"FROM meeting_present;";
          break;
        case "Кваліфікації":
          mySQLCommand.CommandText = "SELECT " +
"id_qualification as 'ІН кваліфікації', " +
"skill_level as 'Кваліфікаційний рівень', " +
"name_of_specialty as 'Назва спеціальності'," +
"code_specialty as 'Код спеціальності', " +
"qualification_of_student as 'Кваліфікація студента' " +
"FROM qualification;";
          break;
        case "Результати державного іспиту":
          mySQLCommand.CommandText = "SELECT " +
"student_id as 'ІН студента', " +
"start_time as 'Час початку екзамену', " +
"end_time as 'Час закінчення екзамену', " +
"id_ticket as 'Номер білету', " +
"rating_score as 'Рейтингова оцінка', " +
"national_rate as 'Національна оцінка', " +
"ECTS_rating as 'Оцінка ECTS', " +
"id_meeting_head as 'ІН номеру протоколу', " +
"question_1 as 'Хто задав питання 1', " +
"question_text_1 as 'Текст питання 1', " +
"id_character_answer1 as 'Хар-ка відповіді на питання 1', " +
"question_2 as 'Хто задав питання 2', " +
"question_text_2 as 'Текст питання 2', " +
"id_character_answer2 as 'Хар-ка відповіді на питання 2', " +
"question_3 as 'Хто задав питання 3', " +
"question_text_3 as 'Текст питання 3', " +
"id_character_answer3 as 'Хар-ка відповіді на питання 3' " +
"FROM report_of_exam;";
          break;
        case "Результати захисту дипломних робіт":
          mySQLCommand.CommandText = "SELECT " +
"student_id_student as 'ІН студента', " +
"id_meeting_head as 'ІН протоколу', " +
"pages as 'Кіл-сть сторінок', " +
"slides as 'Кіл-сть слайдів', " +
"duration as 'Тривалість захисту у хв', " +
"rating_by_head as 'Оцінка керівника', " +
"rating_by_reviewer as 'Оцінка рецензента', " +
"insignia as 'Відзнакою', " +
"postgraduate as 'Аспірантура', " +
"english as 'Англійскою', " +
"rating_vnz as 'Рейтингова оцінка', " +
"rating_national as 'Національна оцінка', " +
"rating_ECTS as 'Оцінка ECTS', " +
"question_1 as 'Хто поставив питання 1', " +
"question_2 as 'Хто поставив питання 2', " +
"question_3 as 'Хто поставив питання 3', " +
"question_4 as 'Хто поставив питання 4', " +
"question_5 as 'Хто поставив питання 5', " +
"FROM report_of_protect;";
          break;
```

```
        case "Номера протоколів":
          mySQLCommand.CommandText = "SELECT " +
"id_meeting_head as 'ІН номерів протоколів', " +
"rate_number_of_protocol as 'Справжній номер протоколу', " +
"type_of_protocol as 'Тип протоколу', " +
"id_meeting_present as 'ІН номеру засідання' " +
"FROM meeting_head;";
          break;
        case "Оцінювання":
          mySQLCommand.CommandText = "SELECT " +
"id_valuation as 'ІН оцінювання', " +
"national_value as 'Національна оцінка', " +
"ECTS_value as 'Оцінка ECTS', " +
"min as 'Мінімум', " +
"max as 'Максимум' " +
"FROM valuation;";
          break;
      }
      try
      {
        mySQLConnection.Open();
        dataAdapter = new MySqlDataAdapter(mySQLCommand);
        dataTable = new DataTable();
        dataTable.Clear();
        dataAdapter.Fill(dataTable);
        TableView.DataSource = dataTable;
        mySQLConnection.Close();
      }
      catch (MySqlException)
      {
        MessageBox.Show("Перевірте зв'язок із базою даних");
      }
    }

    //відображення панелі про додавання студента
    private void addStudent_Click(object sender, EventArgs e)
    {
      studentBoxProtection.Show();
      valuationProtection.Hide();
      try
      {
        mySQLConnection.Open();
        mySQLCommand.CommandText = "SELECT MAX(rate_number_of_protocol + 1) FROM
meeting_head;";
        MySqlDataReader mySQLReader = mySQLCommand.ExecuteReader();
        if (mySQLReader.HasRows)
        {
          if (mySQLReader.Read())
          {
            try
            {
              protocolNumberProtection.Text = mySQLReader.GetString(0);
            }
            catch (System.Data.SqlTypes.SqlNullValueException)
            {
              protocolNumberProtection.Text = "1";
            }
          }
        }
        else protocolNumberProtection.Text = "1";
        mySQLReader.Close();
        mySQLConnection.Close();
      }
```

```
      catch (MySqlException)
      {
        MessageBox.Show("Перевірте зв'язок із базою даних");
      }
    }

    //заповнення комбобоксу вибору студента
    private void studentProtection_DropDown(object sender, EventArgs e)
    {
      if ((qualification.Text == "") || (qualification.Text == "Кваліфікація"))
      {
        MessageBox.Show("Оберіть рівень кваліфікації!");
        studentProtection.Text = "";
        return;
      }

      (sender as ComboBox).DataSource = null;
      (sender as ComboBox).Items.Clear();
      mySQLConnection.Open();
      if (qualification.Text == "Магістр")
        mySQLCommand.CommandText = "SELECT CONCAT( surname, ' ', name) as pib "
+
     "FROM dek.student " +
     "where   student.id_student   not   IN(select   student_id_student   from
report_of_protect)  AND  id_qualification  IN  (Select  id_qualification  FROM
qualification    where    ((skill_level='Магістр')    AND    `year`='"    +
DateTime.Today.Year.ToString() + "'));";
      else if (qualification.Text == "Бакалавр")
        mySQLCommand.CommandText = "SELECT CONCAT( surname, ' ', name) as pib "
+
   "FROM dek.student " +
   "where   student.id_student   not   IN(select   student_id_student   from
report_of_protect)  AND  id_qualification  IN  (Select  id_qualification  FROM
qualification    where    ((skill_level='Бакалавр')    AND    `year`='"    +
DateTime.Today.Year.ToString() + "'));";
      else if (qualification.Text == "Спеціаліст")
        mySQLCommand.CommandText = "SELECT CONCAT( surname, ' ', name) as pib "
+
     "FROM dek.student " +
     "where   student.id_student   not   IN(select   student_id_student   from
report_of_protect)  AND  id_qualification  IN  (Select  id_qualification  FROM
qualification    where    ((skill_level='Спеціаліст')    AND    `year`='"    +
DateTime.Today.Year.ToString() + "'));";
      dataAdapter = new MySqlDataAdapter(mySQLCommand);
      dataTable = new DataTable();
      dataAdapter.Fill(dataTable);
      (sender as ComboBox).DataSource = dataTable;
      (sender as ComboBox).DisplayMember = "pib";
      mySQLConnection.Close();
    }

    //комбобокс кваліфікації
    private void qualification_DropDown(object sender, EventArgs e)
    {
      mySQLConnection.Open();
      mySQLCommand.CommandText    =    "SELECT    skill_level    as    pib    FROM
qualification;";
      dataAdapter = new MySqlDataAdapter(mySQLCommand);
      dataTable = new DataTable();
      //dataTable.Clear();
      dataAdapter.Fill(dataTable);
      (sender as ComboBox).DataSource = dataTable;
      (sender as ComboBox).DisplayMember = "pib";
```

```
      mySQLConnection.Close();
    }

    //оцінка керівника
    private void headValue_DropDown(object sender, EventArgs e)
    {
      mySQLConnection.Open();
      mySQLCommand.CommandText = "SELECT national_value as pib FROM valuation;";
      dataAdapter = new MySqlDataAdapter(mySQLCommand);
      dataTable = new DataTable();
      dataTable.Clear();
      dataAdapter.Fill(dataTable);
      (sender as ComboBox).DataSource = dataTable;
      (sender as ComboBox).DisplayMember = "pib";
      mySQLConnection.Close();
    }

//……

    //додавання інформації до бази даних про захист студента
    private void insertToDBProtection_Click(object sender, EventArgs e)
    {
      //перевірка на правильність введених даних
      if ((nameQuestion1.Text == "") || ((nameQuestion1.Text != "") ))
      {
        MessageBox.Show("Перевірте питання 1 та того, хто його задав! Одне із
цих полів пусте, що неприпустимо!");
        return;
      }
      if (((nameQuestion2.Text == "") ) || ((nameQuestion2.Text != "") ))
      {
        MessageBox.Show("Перевірте питання 2 та того, хто його задав! Одне із
цих полів пусте, що неприпустимо!");
        return;
      }
      if (((nameQuestion3.Text == "") && (textQuestion3.Text != "")) ||
((nameQuestion3.Text != "") && (textQuestion3.Text == "")))
      {
        MessageBox.Show("Перевірте питання 3 та того, хто його задав! Одне із
цих полів пусте, що неприпустимо!");
        return;
      }
      if (((nameQuestion4.Text == "") && (textQuestion4.Text != "")) ||
((nameQuestion4.Text != "") && (textQuestion4.Text == "")))
      {
        MessageBox.Show("Перевірте питання 4 та того, хто його задав! Одне із
цих полів пусте, що неприпустимо!");
        return;
      }
      if (((nameQuestion5.Text == "") && (textQuestion5.Text != "")) ||
((nameQuestion5.Text != "") && (textQuestion5.Text == "")))
      {
        MessageBox.Show("Перевірте питання 5 та того, хто його задав! Одне із
цих полів пусте, що неприпустимо!");
        return;
      }
      string surname = "", name = "";
      if (studentProtection.Text != "")
      {
        surname = studentProtection.Text.Split(' ')[0];
        name = studentProtection.Text.Split()[1];
        if (name == "")
        {
```

```
          MessageBox.Show("Перевірте правильність введеного прізвища. Можливі
зайві пробіли");
          return;
        }
      }
      else
      {
        MessageBox.Show("Введіть, будь ласка, прізвище та ім'я студента!");
        return;
      }
      bool check = false;
      string id_meeting_present = "", id_meeting_head = "", student_id_student =
"", question = "", Temp_Command = "";
      //робота з базою даних
      try
      {
        mySQLConnection.Open();
        mySQLCommand.CommandText   =   "SELECT   MAX(id_meeting_present)   FROM
meeting_present;";
        MySqlDataReader mySQLReader = mySQLCommand.ExecuteReader();
        if (mySQLReader.HasRows)
        {
          if (mySQLReader.Read())
          {
            try
            {
id_meeting_present = mySQLReader.GetString(0);
mySQLReader.Close();
mySQLCommand.CommandText    =    String.Format("INSERT    INTO    meeting_head
(rate_number_of_protocol, type_of_protocol, id_meeting_present) values ('{0}',
'{1}',   '{2}'   );",   protocolNumberProtection.Text,   "Протокол   захисту",
id_meeting_present);
mySQLCommand.ExecuteNonQuery();
check = true;
//Отрмання id_meeting_head із таблиці протоколів
mySQLCommand.CommandText = "SELECT MAX(id_meeting_head) FROM meeting_head;";
mySQLReader = mySQLCommand.ExecuteReader();
mySQLReader.Read();
id_meeting_head = mySQLReader.GetString(0);
mySQLReader.Close();
//Отримання ІД стедента
mySQLCommand.CommandText = "SELECT id_student FROM student WHERE surname='" +
surname   +   "'   and   name='"   +   name   +   "'   AND   `year`='"   +
DateTime.Today.Year.ToString() + "';";
mySQLReader = mySQLCommand.ExecuteReader();
mySQLReader.Read();
student_id_student = mySQLReader.GetString(0);
mySQLReader.Close();

//Отримання номеру того хто задав питання №1
if ((nameQuestion1.Text != "") && (textQuestion1.Text != ""))
{
  surname = nameQuestion1.Text.Split(' ')[0];
  name = nameQuestion1.Text.Split()[1];
  mySQLCommand.CommandText = "SELECT id_member FROM members WHERE surname='" +
surname + "' and name='" + name + "';";
  mySQLReader = mySQLCommand.ExecuteReader();
  mySQLReader.Read();
  question = mySQLReader.GetString(0);
  mySQLReader.Close();
}
else { MessageBox.Show("Комісія повинна задати хоча б 1 питання!"); return; }
```

```
Temp_Command    =    "INSERT    INTO    report_of_protect    (student_id_student,
id_meeting_head,       pages,       slides,       duration,       rating_by_head,
rating_by_reviewer,insignia, postgraduate, english, question_1, question_text_1,
question_2,        question_text_2,question_3,        question_text_3,question_4,
question_text_4,question_5, question_text_5) " +
  "values('" + student_id_student + "','" + id_meeting_head + "','" + pages.Text
+  "','"  +  slides.Text  +  "','"  +  duration.Text  +  "','"  +  "','"  +  "false," +
"false," + "false,'" + question + "','" ;

//……

//Додавання запису до бази даних
mySQLCommand.CommandText = Temp_Command;
mySQLCommand.ExecuteNonQuery();
statusStrip.Items[1].ForeColor = Color.Green;
statusStrip.Items[1].Text = "Інформацію про студента, що захистився додано";
int protocol=Convert.ToInt32(protocolNumberProtection.Text)+1;
foreach (Control c in studentBoxProtection.Controls)
{
  if (c.GetType() == typeof(TextBox))
    c.Text = string.Empty;
  if (c.GetType() == typeof(ComboBox))
    c.Text = string.Empty;

  // if (c.GetType() == typeof(GroupBox))
   //  CleanAllTextBoxesIn(c);
}
protocolNumberProtection.Text=protocol.ToString();

            }
            catch (System.Data.SqlTypes.SqlNullValueException)
            {
MessageBox.Show("Зустрічей не знайдено.");
statusStrip.Items[1].ForeColor = Color.Red;
statusStrip.Items[1].Text = "Зустрічей не знайдено";
if (check == true)
{
  mySQLCommand.CommandText = "DELETE FROM meeting_head WHERE id_meeting_head='"
+ id_meeting_head + "';";
  mySQLCommand.ExecuteNonQuery();
  MessageBox.Show("Протокол не створений.");
}
mySQLConnection.Close();
return;
            }
          }
        }
        else MessageBox.Show("Програма не отримала номер протоколу. Перевірте
підключення до бази даних або саму базу.");
        mySQLConnection.Close();
      }
      catch (MySqlException)
      {
        MessageBox.Show("Перевірте зв'язок із базою даних і правильність введних
даних");
        if (check == true)
        {
          mySQLCommand.CommandText    =    "DELETE    FROM    meeting_head    WHERE
id_meeting_head='" + id_meeting_head + "';";
          mySQLCommand.ExecuteNonQuery();
          MessageBox.Show("Протокол не створений.");
        }
        statusStrip.Items[1].ForeColor = Color.Red;
```

```
        statusStrip.Items[1].Text = "Перевірте зв'язок із базою даних і правильність введних даних";
        mySQLConnection.Close();
      }

    }

    //показ боксу для оцінки студентів
    private void valuationProtectionButton_Click(object sender, EventArgs e)
    {
      valuationProtection.Location = new Point(3, 135);
      valuationProtection.Show();
      studentBoxProtection.Hide();
      //valuationProtection.Dock = DockStyle.Bottom;
      mySQLConnection.Open();
      mySQLCommand.CommandText = "SELECT surname as 'Прізвище', name as 'Ім`я', middle_name as 'По-батькові', rating_by_head as 'Оцінка керівника', rating_by_reviewer as 'Оцінка рецензента',insignia as 'Відзнака',postgraduate as 'Аспірантура',english as 'Англійською',rating_vnz as 'Оцінка ВНЗ' " +
          "FROM student " +
          "JOIN report_of_protect " +
          "ON student.id_student=report_of_protect.student_id_student " +
          "WHERE student.id_student " +
          "IN(select student_id_student from report_of_protect where report_of_protect.id_meeting_head " +
          "IN(select id_meeting_head from meeting_head where (id_meeting_present " +
          "IN(select MAX(id_meeting_present) from meeting_present) " +
          "AND type_of_protocol='Протокол захисту') )) AND `year`='" + DateTime.Today.Year.ToString() + "';";
      dataAdapter = new MySqlDataAdapter(mySQLCommand);
      dataTable = new DataTable();
      dataTable.Clear();
      dataAdapter.Fill(dataTable);
      valuationProtectionTable.DataSource = dataTable;
      mySQLConnection.Close();
      statusStrip.Items[1].ForeColor = Color.Green;
      statusStrip.Items[1].Text = "Список сформавано";
    }

    //сховати панль для оцінки студентів
    private void hideValuationProtection_Click(object sender, EventArgs e)
    {
      valuationProtection.Hide();
    }

    //сховування боксу для захисту студентів
    private void hideProtection_Click(object sender, EventArgs e)
    {
      studentBoxProtection.Hide();
    }

    //час коли почалося засідання із захисту дипломів
    private void timeStartProtection_Enter(object sender, EventArgs e)
    {
      (sender as TextBox).SelectAll();
      (sender as TextBox).Text = DateTime.Now.ToString("HH:mm");
      (sender as TextBox).SelectAll();
    }

    //занесення даних про оцінювання студентів із захисту дипломів
    private void updateDBProtecion_Click(object sender, EventArgs e)
    {
```

```
      int i;
      string national_value = "", ECTS_value = "";
      try
      {
        mySQLConnection.Open();
        for (i = 0; i < (valuationProtectionTable.RowCount - 1); i++)
        {//отримання усіх оцінок виходячи з рейтинговоі
          mySQLCommand.CommandText = "SELECT national_value,ECTS_value " +
       "FROM valuation " +
       "WHERE                                min<='"                                +
valuationProtectionTable.Rows[i].Cells[8].Value.ToString()  +  "'  and  max>='"  +
valuationProtectionTable.Rows[i].Cells[8].Value.ToString() + "';";
          MySqlDataReader mySQLReader = mySQLCommand.ExecuteReader();
          if (mySQLReader.HasRows)
          {
            if (mySQLReader.Read())
            {
national_value = mySQLReader.GetString(0);
ECTS_value = mySQLReader.GetString(1);
            }
          }
          else
          {
            MessageBox.Show("Знайдено неправильне значення у стоіпчику \"Оцінка
ВНЗ\". Запис №" + (i + 1));
            statusStrip.Items[1].ForeColor = Color.Red;
            statusStrip.Items[1].Text   =   "Знайдено   неправильне   значення   у
стоіпчику \"Оцінка ВНЗ\". Запис №" + (i + 1);
            mySQLReader.Close();
            mySQLConnection.Close();
            return;
          }
          mySQLReader.Close();
          //Запис до БД введеної інформації про іспит
          mySQLCommand.CommandText = "UPDATE report_of_protect SET insignia = "
+ valuationProtectionTable.Rows[i].Cells[5].Value.ToString() + ", postgraduate =
" + valuationProtectionTable.Rows[i].Cells[6].Value.ToString() +
", english = " + valuationProtectionTable.Rows[i].Cells[7].Value.ToString() + ",
rating_vnz = '" + valuationProtectionTable.Rows[i].Cells[8].Value.ToString() +
"', rating_national = '" + national_value + "', rating_ECTS = '" + ECTS_value +
"' " +
"WHERE student_id_student = " +
"(SELECT id_student " +
"FROM student " +
"WHERE surname = '" + valuationProtectionTable.Rows[i].Cells[0].Value.ToString()
+ "' AND name = '" + valuationProtectionTable.Rows[i].Cells[1].Value.ToString()
+
"'            AND            middle_name            =            '"            +
valuationProtectionTable.Rows[i].Cells[2].Value.ToString() + "' AND `year`='" +
DateTime.Today.Year.ToString() + "' ); ";
          mySQLCommand.ExecuteNonQuery();
        }
        //внесення часу закінчення засідання по захисту дипломних робіт
        if (timeEndProtection.Text != "")
        {
          mySQLCommand.CommandText = "UPDATE meeting_present SET end_time = '" +
timeEndProtection.Text + "'" +
  "WHERE    id_meeting_present    =    (SELECT    MAX(id_meeting_present)    FROM
meeting_head);";
          mySQLCommand.ExecuteNonQuery();
        }
        else
        {
```

```
          MessageBox.Show("Перевірте час завершення засідання");
          statusStrip.Items[1].ForeColor = Color.Red;
          statusStrip.Items[1].Text   =   "Не   введено   значення   завершення
засідання";
          mySQLConnection.Close();
          return;
        }
        mySQLConnection.Close();
        statusStrip.Items[1].ForeColor = Color.Green;
        statusStrip.Items[1].Text = "Інформацію про оцінювання додано";
      }
      catch (MySqlException)
      {
        MessageBox.Show("Перевірте  всі  значення  у  таблиці.  Можливо  не  ввели
оцінку(-ок)");
        statusStrip.Items[1].ForeColor = Color.Red;
        statusStrip.Items[1].Text = "Перевірте всі значення у таблиці. Можливо
не ввели оцінку(-ок)";
        mySQLConnection.Close();
        return;
      }
      reportBox.Show();
    }



//……

    //створення протоколу про захист
    private void reportProtectionButton_Click(object sender, EventArgs e)
    {
      int i=0;
      string temp = "";


      //отримання номеру засідання
      mySQLConnection.Open();
      mySQLCommand.CommandText = "SELECT MAX(rate_number_of_protocol) " +
        "FROM meeting_head WHERE type_of_protocol='Протокол захисту';";
      MySqlDataReader mySQLReader = mySQLCommand.ExecuteReader();
      if (mySQLReader.HasRows)
      {
        if (mySQLReader.Read())
        {
          temp = mySQLReader.GetString(0);
        }
      }
      mySQLReader.Close();

      //заповнення номеру протоколу у шаблоні


      //створення документа за шаблоном
      WordApp = new Microsoft.Office.Interop.Word.Application();
      WordDoc      =      WordApp.Documents.Open(Application.StartupPath      +
"\\templates\\protection.dot"); //вказівка шляху до шаблону

      //виведення вікна для збереження готового звіту
      saveFileDialog1.ShowDialog();
      saveFileDialog1.AddExtension = true;
      saveFileDialog1.CheckPathExists = true;
      if (saveFileDialog1.FileName != "")
      {
```

```
        WordDoc.SaveAs(saveFileDialog1.FileName);
      }


      WordRange = WordApp.Selection.Range;
      WordFind = WordApp.Selection.Find;
      WordApp.Visible = true;


      replaceText(WordFind,                                          "[протокол]",
monthToString(dateProtection.Text.Split('-')[1]));

      //заповнення дати у шапці шаблону
      replaceText(WordFind, "[день1]", dateProtection.Text.Split('-')[2]);
      replaceText(WordFind,                                           "[місяць1]",
monthToString(dateProtection.Text.Split('-')[1]));
      replaceText(WordFind, "[рік1]", dateProtection.Text.Split('-')[0]);

      //тема студента
      mySQLCommand.CommandText = "SELECT topic " +
        "FROM student WHERE surname='" + reportList.Text.Split(' ')[1] + "' AND
`name`='"   +   reportList.Text.Split('   ')[2]   +   "'   AND   middle_name='"   +
reportList.Text.Split(' ')[3] + "';";
      mySQLReader = mySQLCommand.ExecuteReader();
      mySQLReader.Read();
      replaceText(WordFind, "[тема]", mySQLReader.GetString(0));
      replaceText(WordFind, "[студент]", reportList.Text.Split(' ')[1] + " " +
reportList.Text.Split(' ')[2] + " " + reportList.Text.Split(' ')[3]);
      mySQLReader.Close();

      //кваліфікація студента
      mySQLCommand.CommandText = "SELECT skill_level FROM qualification WHERE
id_qualification=(SELECT  id_qualification  FROM  student  WHERE  surname='"  +
reportList.Text.Split(' ')[1] + "' AND `name`='" + reportList.Text.Split(' ')[2]
+ "' AND middle_name='" + reportList.Text.Split(' ')[3] + "');";
      mySQLReader = mySQLCommand.ExecuteReader();
      mySQLReader.Read();
      replaceText(WordFind, "[кваліфікація]", mySQLReader.GetString(0));
      mySQLReader.Close();

      //інформація за спеціальністю
      mySQLCommand.CommandText  =  "SELECT  name_of_specialty,  code_specialty,
qualification_of_student FROM qualification WHERE skill_level='" + temp+ "';";
      mySQLReader = mySQLCommand.ExecuteReader();
      mySQLReader.Read();

      replaceText(WordFind, "[назва_шифру]", mySQLReader.GetString(0));
      replaceText(WordFind, "[шифр]", mySQLReader.GetString(1));
      replaceText(WordFind, "[наз_квал]", mySQLReader.GetString(2));
      mySQLReader.Close();

      //заповнення номеру комісії
      mySQLCommand.CommandText  =  "SELECT  number_of_DEK  FROM  commission_number
WHERE id_commission=(SELECT MAX(id_commission) FROM commission_number);";
      mySQLReader = mySQLCommand.ExecuteReader();
      mySQLReader.Read();
      replaceText(WordFind, "[№_ком]", mySQLReader.GetString(0));
      mySQLReader.Close();

      //внесення голови засідання
      mySQLCommand.CommandText   =   "SELECT   surname,   `name`,   middle_name,
science_degree, academic_title, post FROM members WHERE id_member = " +
        "(SELECT id_chief FROM meeting_present where id_meeting_present = " +
```

```
        "(SELECT MAX(id_meeting_present) FROM meeting_present));";
      mySQLReader = mySQLCommand.ExecuteReader();
      mySQLReader.Read();
      replaceText(WordFind, "[прізвище]", mySQLReader.GetString(0));
      replaceText(WordFind, "[ІГ]", mySQLReader.GetString(1)[0].ToString());
      replaceText(WordFind, "[ПГ]", mySQLReader.GetString(2)[0].ToString());
      replaceText(WordFind, "[наук_ступ]", mySQLReader.GetString(3));
      replaceText(WordFind, "[вчен_зван]", mySQLReader.GetString(4));
      replaceText(WordFind, "[посада]", mySQLReader.GetString(5));
      mySQLReader.Close();

      //заповнення присутніх на засіданні
      mySQLCommand.CommandText = "SELECT surname, name, middle_name, post " +
          "FROM members " +
          "WHERE  id_member=(SELECT  id_present_1  FROM  meeting_present  WHERE
id_meeting_present = (SELECT MAX(id_meeting_present) FROM meeting_present)) " +
          "or   id_member=(SELECT   id_present_2   FROM   meeting_present   WHERE
id_meeting_present = (SELECT MAX(id_meeting_present) FROM meeting_present)) " +
          "or   id_member=(SELECT   id_present_3   FROM   meeting_present   WHERE
id_meeting_present = (SELECT MAX(id_meeting_present) FROM meeting_present)) " +
          "or   id_member=(SELECT   id_present_4   FROM   meeting_present   WHERE
id_meeting_present = (SELECT MAX(id_meeting_present) FROM meeting_present)) " +
          "or   id_member=(SELECT   id_present_5   FROM   meeting_present   WHERE
id_meeting_present = (SELECT MAX(id_meeting_present) FROM meeting_present)) " +
          "or   id_member=(SELECT   id_present_6   FROM   meeting_present   WHERE
id_meeting_present = (SELECT MAX(id_meeting_present) FROM meeting_present)) " +
          "or   id_member=(SELECT   id_present_7   FROM   meeting_present   WHERE
id_meeting_present = (SELECT MAX(id_meeting_present) FROM meeting_present));";
      mySQLReader = mySQLCommand.ExecuteReader();
      i = 0;
      while (mySQLReader.Read())
      {
        i++;
        replaceText(WordFind,   "[ПР"   +   (i   +   1).ToString()   +   "]",
mySQLReader.GetString(0));
        replaceText(WordFind,   "[І"   +   (i   +   1).ToString()   +   "]",
mySQLReader.GetString(1)[0].ToString());
        replaceText(WordFind,   "[П"   +   (i   +   1).ToString()   +   "]",
mySQLReader.GetString(2)[0].ToString());
        replaceText(WordFind,  "[посада"  +  (i  +  1).ToString()  +  "]",
mySQLReader.GetString(3));
      }
      mySQLReader.Close();

      //вивід інформації про результати виступу студента
      mySQLCommand.CommandText   =   "SELECT   rating_vnz,   rating_national,
rating_ECTS,
insignia,question_text_1,question_text_2,question_text_3,question_text_4,questio
n_text_5 " +
        "FROM report_of_protect " +
        "WHERE  student_id_student=(SELECT  MAX(id_student)  FROM  student  WHERE
surname='"   +   reportList.Text.Split('   ')[1]   +   "'   AND   `name`='"   +
reportList.Text.Split(' ')[2] + "' AND middle_name='" + reportList.Text.Split('
')[3] + "');";
      mySQLReader = mySQLCommand.ExecuteReader();
      mySQLReader.Read();

      Microsoft.Office.Interop.Word.Table WordTable = WordDoc.Tables[1];

      if (mySQLReader.Read())
      {
        WordTable = WordDoc.Tables[1];
        i = 0;
```

```
        WordTable.Cell(i + 4, 1).Range.Text = (i + 1).ToString();
        WordTable.Cell(i + 4, 8).Range.Text = mySQLReader.GetString(3);
        WordTable.Cell(i + 4, 9).Range.Text = mySQLReader.GetString(4);
        WordTable.Cell(i          +          4,          10).Range.Text          =
(Int32.Parse(mySQLReader.GetString(3)) > 60) ? "так" : "ні";
        WordTable.Cell(i  +  4,  11).Range.Text  =  (mySQLReader.GetString(6)  ==
"True") ? "ЗВ" : "-";

        while (mySQLReader.Read())
        {
          i++;
          WordTable.Cell(WordTable.Rows.Count, 1).Range.Rows.Add();
          WordTable.Cell(i + 4, 1).Range.Text = (i + 1).ToString();
          WordTable.Cell(i + 4, 8).Range.Text = mySQLReader.GetString(3);
          WordTable.Cell(i + 4, 9).Range.Text = mySQLReader.GetString(4);
          WordTable.Cell(i          +          4,          10).Range.Text          =
(Int32.Parse(mySQLReader.GetString(3)) > 60) ? "так" : "ні";
          WordTable.Cell(i  +  4,  11).Range.Text  =  (mySQLReader.GetString(6)  ==
"True") ? "ЗВ" : "-";
        }
      }



      //внесення керівника
      mySQLCommand.CommandText   =   "SELECT   surname,   `name`,   middle_name,
science_degree, academic_title, post " +
        "FROM members " +
        "WHERE  id_member  =  (SELECT  id_head  FROM  student  where  surname='" +
reportList.Text.Split(' ')[1] + "' AND `name`='" + reportList.Text.Split(' ')[2]
+ "' AND middle_name='" + reportList.Text.Split(' ')[3] + "');";
      mySQLReader = mySQLCommand.ExecuteReader();
      mySQLReader.Read();

      if (mySQLReader.Read())
      {
        WordTable = WordDoc.Tables[1];
        i = 0;
        WordTable.Cell(i + 4, 1).Range.Text = (i + 1).ToString();
        WordTable.Cell(i + 4, 8).Range.Text = mySQLReader.GetString(3);
        WordTable.Cell(i + 4, 9).Range.Text = mySQLReader.GetString(4);
        WordTable.Cell(i          +          4,          10).Range.Text          =
(Int32.Parse(mySQLReader.GetString(3)) > 60) ? "так" : "ні";
        WordTable.Cell(i  +  4,  11).Range.Text  =  (mySQLReader.GetString(6)  ==
"True") ? "ЗВ" : "-";

        while (mySQLReader.Read())
        {
          i++;
          WordTable.Cell(WordTable.Rows.Count, 1).Range.Rows.Add();
          WordTable.Cell(i + 4, 1).Range.Text = (i + 1).ToString();
          WordTable.Cell(i + 4, 8).Range.Text = mySQLReader.GetString(3);
          WordTable.Cell(i + 4, 9).Range.Text = mySQLReader.GetString(4);
          WordTable.Cell(i          +          4,          10).Range.Text          =
(Int32.Parse(mySQLReader.GetString(3)) > 60) ? "так" : "ні";
          WordTable.Cell(i  +  4,  11).Range.Text  =  (mySQLReader.GetString(6)  ==
"True") ? "ЗВ" : "-";
        }
      }
//…….

    private void textBox16_Enter(object sender, EventArgs e)
    {
```

```
      if (sender is TextBox)
        (sender as TextBox).SelectAll();
      if (sender is ComboBox)
        (sender as ComboBox).SelectAll();
    }

    private void button5_Click(object sender, EventArgs e)
    {
      uint a, b, c, d;
      if (checkBox1.Checked == true) a = 1; else a = 0; //перетворює значення
Керівника для запису в БД
      if (checkBox2.Checked == true) b = 1; else b = 0; //значення рецинзента
      if (checkBox4.Checked == true) c = 1; else c = 0; // значення члену
комісії
      if (checkBox3.Checked == true) d = 1; else d = 0; // значення консультанта
      mySQLConnection.Open();
      try
      {
        mySQLCommand = new MySqlCommand(String.Format("INSERT INTO members
(surname, name, middle_name, job, post, science_degree, academic_title, head,
reviewer, DEK_member, consultant) values('{0}', '{1}', '{2}', '{3}', '{4}',
'{5}', '{6}', '{7}', '{8}', '{9}', '{10}');", textBox16.Text, textBox15.Text,
textBox14.Text,      textBox13.Text,      textBox12.Text,      comboBox19.Text,
comboBox20.Text, a, b, c, d), mySQLConnection);
        mySQLCommand.ExecuteNonQuery();
      }
      catch (MySqlException) {
        MessageBox.Show("Перевірте введені дані");
        mySQLConnection.Close();
        statusStrip.Items[1].ForeColor = Color.Red;
        statusStrip.Items[1].Text = "Помилка у введених даних";
        return;
      };
      mySQLConnection.Close();
      statusStrip.Items[1].ForeColor = Color.Green;
      statusStrip.Items[1].Text = "Запис про членів комісії додано";
    }

    private void comboBox24_Click(object sender, EventArgs e)
    {
      (sender as ComboBox).DataSource = null;
      (sender as ComboBox).Items.Clear();
      mySQLConnection.Open();
      mySQLCommand.CommandText = "SELECT CONCAT( surname, ' ', name) as pib FROM
dek.members WHERE head=1;";
      dataAdapter = new MySqlDataAdapter(mySQLCommand);
      dataTable = new DataTable();
      dataTable.Clear();
      dataAdapter.Fill(dataTable);
      (sender as ComboBox).DataSource = dataTable;
      (sender as ComboBox).DisplayMember = "pib";
      mySQLConnection.Close();
    }

//……
    private void comboBox23_Click(object sender, EventArgs e)
    {
      (sender as ComboBox).DataSource = null;
      (sender as ComboBox).Items.Clear();
      mySQLConnection.Open();
      mySQLCommand.CommandText = "SELECT CONCAT( surname, ' ', name) as pib FROM
dek.members WHERE reviewer=1;";
      dataAdapter = new MySqlDataAdapter(mySQLCommand);
```

```
      dataTable = new DataTable();
      dataTable.Clear();
      dataAdapter.Fill(dataTable);
      (sender as ComboBox).DisplayMember = "pib";
      (sender as ComboBox).DataSource = dataTable;
      (sender as ComboBox).ValueMember = "pib";
      mySQLConnection.Close();
    }

    private void comboBox25_Click(object sender, EventArgs e)
    {
      (sender as ComboBox).DataSource = null;
      (sender as ComboBox).Items.Clear();
      mySQLConnection.Open();
      mySQLCommand.CommandText = "SELECT CONCAT( surname, ' ', name) as pib FROM
dek.members WHERE consultant=1;";
      dataAdapter = new MySqlDataAdapter(mySQLCommand);
      dataTable = new DataTable();
      dataTable.Clear();
      dataAdapter.Fill(dataTable);
      (sender as ComboBox).DataSource = dataTable;
      (sender as ComboBox).DisplayMember = "pib";
      mySQLConnection.Close();
    }

    private void comboBox34_Click(object sender, EventArgs e)
    {
      (sender as ComboBox).DataSource = null;
      (sender as ComboBox).Items.Clear();
      mySQLConnection.Open();
      mySQLCommand.CommandText = "SELECT skill_level FROM qualification;";
      dataAdapter = new MySqlDataAdapter();
      dataAdapter.SelectCommand = mySQLCommand;
      dataTable = new DataTable();
      dataAdapter.Fill(dataTable);
      (sender as ComboBox).DisplayMember = "skill_level";
      (sender as ComboBox).DataSource = dataTable;
      (sender as ComboBox).ValueMember = "skill_level";
      mySQLConnection.Close();
    }
    //структура Прізвище та Ім'я
    struct hum
    {
      public string name;
      public string surname;
    };
    private void button13_Click(object sender, EventArgs e)
    {
      string reviewer, head, itproj, econom, safety, qualification;
      hum GetNumber;
      mySQLConnection.Open();
      // отримання номеру керівника із таблиці членів комісії
      GetNumber.surname = comboBox24.Text.Split(' ')[0];
      GetNumber.name = comboBox24.Text.Split()[1];
      mySQLCommand.CommandText = "SELECT dek.members.id_member FROM dek.members
WHERE dek.members.surname='" + GetNumber.surname + "' and dek.members.name='" +
GetNumber.name + "';";
      MySqlDataReader mySQLReader = mySQLCommand.ExecuteReader();
      mySQLReader.Read();
      head = mySQLReader.GetString(0);
      mySQLReader.Close();
//……
```

```
      //безпосередній запис до таблиці студент
      mySQLCommand = new MySqlCommand(String.Format("INSERT INTO dek.student
(dek.student.surname,         dek.student.name,         dek.student.middle_name,
dek.student.topic,         dek.student.group,         dek.student.form_education,
dek.student.id_head,     dek.student.id_reviewer,     dek.student.id_consultant,
id_qualification,  year)  values('{0}',  '{1}',  '{2}',  '{3}',  '{4}',  '{5}',
'{6}','{7}','{8}','{9}','{10}','{11}','{12}');", textBox26.Text, textBox25.Text,
textBox24.Text, textBox22.Text, textBox23.Text, comboBox18.Text, head, reviewer,
safety, qualification), mySQLConnection);
      mySQLCommand.ExecuteNonQuery();
      mySQLConnection.Close();

      statusStrip.Items[1].ForeColor = Color.Green;
      statusStrip.Items[1].Text = "Запис про студента додано";
    }
//……

    private void replaceText(Microsoft.Office.Interop.Word.Find WordFind, string
search, string replace)
    {
      WordFind.ClearFormatting();
      WordFind.Text = search;
      WordFind.Replacement.ClearFormatting();
      WordFind.Replacement.Text = replace;
      WordFind.Execute(Replace:
Microsoft.Office.Interop.Word.WdReplace.wdReplaceAll);
    }


  }
}
```