МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
імені І. І. МЕЧНИКОВА
ФАКУЛЬТЕТ МАТЕМАТИКИ, ФІЗИКИ
ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА МАТЕМАТИЧНОГО ЗАБЕЗПЕЧЕННЯ КОМП'ЮТЕРНИХ СИСТЕМ

ПРОЕКТУВАННЯ ІНФОРМАЦІЙНИХ СИСТЕМ

Методичні вказівки
до курсового проектування
для студентів факультету математики, фізики та
інформаційних технологій
першого (бакалаврського) рівня освіти,
спеціальності 126 «Інформаційні системи та технології»

Одеса
Олді+
2023

УДК 004.5/6(072)
П791

Укладачі:
Малахов Є. В., д.т.н., професор кафедри математичного забезпечення комп'ютерних систем;
Розновець О. І., старший викладач кафедри математичного забезпечення комп'ютерних систем.

Рецензенти:
Вербицький В. В., к.ф.-м.н, доцент кафедри оптимального управління та економічної кібернетики Одеського національного університету імені І. І. Мечникова;
Рачинська А. Л., к.ф.-м.н., доцент кафедри механіки, автоматизації та інформаційних технологій Одеського національного університету імені І. І. Мечникова.

Рекомендовано Вченою радою
факультету Математики, фізики та інформаційних технологій
ОНУ імені І. І. Мечникова.
Протокол № 2 від 30 жовтня 2023 р.

П791 **Проектування** інформаційних систем: метод. вказівки до курсового проектування студентів факультету математики, фізики та інформаційних технологій першого (бакалаврського) рівня освіти, спец. 126 «Інформаційні системи та технології» / уклад.: Є. В. Малахов, О. І. Розновець, – Одеса : Олді+, 2023. – 54 с.

Пропоновані методичні вказівки стануть у нагоді при виконанні курсових проектів з обов'язкової дисципліни «Проектування інформаційних систем», яка викладається студентам першого (бакалаврського) рівня вищої освіти спеціальності 126 «Інформаційні системи та технології» факультету математики, фізики та інформаційних технологій Одеського національного університету імені І. І. Мечникова. Методичні вказівки можуть бути корисні для студентів IT-спеціальностей закладів вищої освіти при опануванні ними дисциплін з проектування баз даних та інформаційних систем.

УДК 004.5/6(072)

ЗМІСТ

ПРЕАМБУЛА

Курсове проектування з дисципліни «Проектування інформаційних систем» здійснюється у відповідності до навчальної програми обов'язкової освітньої компоненти освітньо-професійної програми підготовки здобувачів вищої освіти першого (бакалаврського) рівня вищої освіти за спеціальністю 126 – «Інформаційні системи та технології».

Дисципліна викладається студентам третього курсу. Загальна кількість кредитів, відведених на вивчення дисципліни – 4. Загальна кількість годин – 120, з них лекцій – 16 годин, лабораторних робіт – 16 годин, самостійна робота – 88 годин, з яких 60 годин відводиться на виконання курсового проекту.

Метою викладання дисципліни є вивчення студентами архітектур інформаціних систем, ефективних технологій організації безпечного доступу до баз даних інформаціних систем, придбання практичних навичок щодо використання шаблонів проектування користувальницьких інтерфейсів з використанням існуючих CASE-систем для маніпулювання даними та організації взаємодії програмних застосунків з відповідними серверами.

Завдання дисципліни:

- ознайомлення з технологіями побудови информаційних систем;
- вивчення архітектур інформаційих систем;
- ознайомлення з технологіями організації безпечного доступу до баз даних;
- підготовка до виконання дипломних проектів та кваліфікаційних робіт, тематика яких пов'язана з дослідженням та проектуванням інформаційних систем та систем підтримки прийняття рішень.

У результаті вивчення дисципліни та виконання курсового проекту студент повинен

знати: архітектури інформаційних систем; технології організації взаємодії клієнт/сервер; технології забезпечення доступу до баз даних та захисту даних в інформаційних системах; основні операції реляційної алгебри, їхню реалізацію і використання в базах даних; команди і оператори мови SQL;

вміти: проектувати інформаційні системи; створювати та налагоджувати розподілені інформаційні системи; будувати користувацькі інтерфейси з використанням мов високого рівня, існуючих CASE-систем та сучасних фреймворків для організації маніпулювання даними шляхом SQL-запитів; забезпечувати безпеку зберігання даних та доступу до них.

Виконання курсового проекту з дисципліни «Проектування інформаційних систем» вимагає знань і вмінь, отриманих студентами при опануванні дисциплін «Операційні системи і середовища», «Алгоритмізація та програмування», «Структури даних та алгоритми», «Веб-технології та веб-дизайн», «Організація баз даних та знань».

Практичні навички, отриманні при виконанні студентами курсового проекту є базою для опанування дисципліни «Управління ІТ-проектами» та виконання кваліфікаційних робіт.

Завдання на курсовий проект видається на початку 6 семестру. Після виконання та оформлення курсового проекту студент захищає його за 100 бальною системою. При цьому береться до уваги компетентність студента, оригінальність та творчість мислення, обґрунтованість прийнятих рішень, ритмічність у роботі (дотримання строків здачі роботи або її складових частин).

Етапи курсового проектування відповідають розділам пояснювальної записки, що розглядається як приклад у даних методичних вказівках.

СТРУКТУРА ПОЯСНЮВАЛЬНОЇ ЗАПИСКИ ДО КУРСОВОГО ПРОЕКТУ З ДИСЦИПЛІНИ «ПРОЕКТУВАННЯ ІНФОРМАЦІЙНИХ СИСТЕМ»

Пояснювальна записка до курсового проекту з дисципліни «Проектування інформаційних систем» повинна містити наступні структурні елементи (в порядку їх розташування):

- титульний лист;
- анотація;
- зміст;
- перелік скорочень, умовних позначень і термінів (при необхідності);
- вступ;
- розділи основної частини пояснювальної записки;
- висновки;
- список використаних джерел;
- додатки (при необхідності).

Початок кожного структурного елементу пояснювальної записки повинен розміщуватись на новій сторінці.

ТИТУЛЬНИЙ ЛИСТ

Титульний лист є першою сторінкою пояснювальної записки і оформлюється відповідно до наведеного в додатку А зразку. Електронну версію бланка титульного листа можна отримати на кафедрі.

АНОТАЦІЯ

Анотація являє собою коротку характеристику проекту і містить формулювання теми проекту, мету, основні положення проекту, відмінні риси та переваги. Анотація повинна займати не більше однієї сторінки.

Примітка. У даних методичних вказівках в якості прикладу розглядається пояснювальна записка до курсового проекту на тему «Облік обладнання навчальних лабораторій кафедри».

Приклад тексту анотації наведений нижче.

Мета даного курсового проекту – проектування і реалізація інформаційної системи обліку обладнання навчальних лабораторій кафедри університету. Користувачами даної системи є студенти і співробітники кафедри. Реалізація виконана з використанням мови C # і СУБД PostgreSQL. Архітектура системи відповідає шаблону MVP.

У розробленій інформаційній системі передбачений облік аудиторій, викладачів і закріплених за ними дисциплін. Особливістю системи є організація обліку обладнання і програмного забезпечення в аудиторіях, а також можливість складання заявок на ремонт обладнання, комплектуючих або установку програмного забезпечення. В інформаційній системі реалізований захист від несанкціонованого доступу та здійснено розмежування повноважень різних категорій користувачів.

Результатом курсового проектування є інформаційна система управління навчальними лабораторіями кафедри зі зручним користувацьким інтерфейсом, що легко сприймається.

ЗМІСТ ПОЯСНЮВАЛЬНОЇ ЗАПИСКИ

Зміст розташовується після титульного листа і анотації. Він повинен включати в себе назви структурних частин пояснювальної записки: «ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ І ТЕРМІНІВ», «ВСТУП», назви всіх РОЗДІЛІВ і, при необхідності, підрозділів, «ВИСНОВКИ», «СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ», «ДОДАТКИ» із зазначенням номерів сторінок, на яких розміщується початок викладу відповідних частин роботи.

Нижче наведений приклад змісту пояснювальної записки до курсового проекту з дисципліни «Проектування інформаційних систем».

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ І ТЕРМІНІВ

Перелік скорочень, умовних позначень і термінів розташовується після змісту. У ньому наводяться (у алфавітному порядку) всі використовувані в пояснювальній записці малопоширені скорочення, абревіатури, умовні позначення і терміни.

У пояснювальній записці до курсового проекту з дисципліни «Проектування інформаційних систем» не слід приводити в переліку наступні загальновідомі скорочення, умовні позначення і терміни: БД – база даних, СУБД – система управління базами даних, ІС – інформаційна система, ПЗ – програмне забезпечення, MVP – Model-View-Presenter, MVC – Model-View-Controller, UI – User Interface.

Нижче наведені приклади скорочень, умовних позначень і термінів, які доцільно привести в переліку. Зазвичай подібні поняття є або специфічними для даної предметної області, або для використовуваних технологій і засобів розробки.

ДВО – додаткові види обслуговування

ОУ – освітня установа

CSRF (Cross Site Request Forgery) – міжсайтова підробка запиту. Являє собою тип шкідливих атак, що використовує недоліки протоколу HTTP, при якому неавторизовані команди виконуються від імені авторизованого користувача.

У разі повторення в тексті записки спеціальних термінів, скорочень, абревіатур, умовних позначень менше трьох разів, їх розшифровка наводиться в тексті при першому згадуванні, наприклад:

При виборі інформаційної системи для обліку обладнання в університеті необхідно враховувати питання фінансування освітніх установ (ОУ).

ВСТУП

Вступ розташовується після переліку скорочень, умовних позначень і термінів (якщо такий є) і, як правило, має обсяг до трьох сторінок. У вступі необхідно:

1) проаналізувати сучасний стан проблем і потреб користувачів розглянутої предметної області, а також оцінити існуючі рішення;

2) з урахуванням проведеної оцінки обґрунтувати актуальність розробки інформаційної системи;

3) коротко сформулювати мету проекту та задачі, які необхідно розв’язати для досягнення цієї мети.

Нижче наведений приклад тексту вступу.

Ведення обліку обладнання на будь-якому підприємстві або в будь-якій установі являє собою трудомісткий, тривалий і ресурсозатратний процес, при якому необхідно постійно стежити за зміною вимог і норм, а також враховувати фактичний стан обладнання і складу. Цей процес обтяжується людським фактором, оскільки неможливо відстежити вручну всі зміни комплектації обладнання. Для автоматизації обліку обладнання застосовуються спеціалізовані інформаційні системи.

При виборі інформаційної системи для обліку обладнання в університеті необхідно враховувати питання фінансування освітніх установ (ОУ), так як в першу чергу їхні витрати йдуть на обладнання та комплектуючі, програмне забезпечення, необхідне для проведення занять. З цієї причини програмні рішення автоматизації процесів управління й обліку обладнання є для ОУ дорогим придбанням не першочергової важливості.

На жаль, на ринку існує мало вільних або відкритих систем подібного призначення, вони або є комерційно невигідними, або мають низьку якість, або не покривають запити споживача в повній мірі.

Окрім вищесказаного, автоматизована система обліку обладнання в університеті, крім своєї основної функції, повинна відображати специфіку установи, беручи до уваги інформацію про викладачів і дисципліни, які вони викладають, а також легко інтегруватися з уже існуючими системами, наприклад, з системою бухгалтерського обліку.

Мета даного курсового проекту – проектування і реалізація інформаційної системи обліку обладнання навчальних лабораторій кафедри університету. Створювана інформаційна система покликана допомогти співробітникам кафедри і бухгалтерії мати доступ до облікової інформації, не вдаючись до допомоги адміністратора. При цьому:

– співробітники і адміністратор отримають універсальний інструмент для швидкого відстеження неполадок і складання заявок на ремонт обладнання і установку ПЗ;

– працівники бухгалтерії отримають інструмент для спрощення ведення обліку обладнання;

– адміністрація університету зможе отримувати більш прозоре уявлення про стан справ в даній області.

Для досягнення зазначеної мети необхідно розв’язати наступні задачі:

1) виконати аналіз предметної області;

2) визначити категорії користувачів і сформулювати їхні вимоги до створюваної інформаційної системи;

3) обрати архітектуру і шаблон проектування створюваної інформаційної системи;

4) спроектувати базу даних;

5) обрати технології та засоби реалізації інформаційної системи;

6) з урахуванням вибраних засобів створити БД, а також клієнтську програму, яка дасть можливість різним категоріям користувачів ефективно маніпулювати даними предметної області відповідно до їхніх повноважень;

7) забезпечити цілісність і безпеку даних як на рівні БД, так і на рівні застосунку;

8) забезпечити захист системи від несанкціонованого доступу і розмежування повноважень з боку різних категорій користувачів.

ОСНОВНА ЧАСТИНА

Основна частина пояснювальної записки до курсового проекту з дисципліни «Проектування інформаційних систем» включає розділи, пов'язані з проектуванням і розробкою ІС обраної предметної області:

- постановка задачі;
- проектування ІС;
- програмна модель застосунка;
- інформаційне моделювання предметної області;
- вибір програмного забезпечення для створення ІС;
- створення бази даних;
- запити до бази даних для розв’язання поставлених задач;
- програмна реалізація інтерфейсу;
- безпека ІС;
- інструкція користувача.

ПОСТАНОВКА ЗАДАЧІ

В процесі проектування ІС даний етап є найбільш важливим. Він передбачає збір і аналіз вимог, що пред'являються до змісту даних і процесу їх обробки користувачами різних категорій. На основі аналізу вимог користувачів визначаються вимоги до БД.

Розділ «Постановка задачі» повинен включати:

- формулювання задач, що розв’язуються інформаційною системою в цілому;
- визначення категорій користувачів, що працюють з ІС;
- формулювання задач користувачів, що розв’язуються за допомогою ІС.

Конкретні задачі, які покликана розв’язувати ІС, залежать від предметної області, для якої призначена ІС. Очевидно, що спеціалізовані задачі медичних систем відрізняються від спеціалізованих задач систем підтримки продажів. Не слід приводити занадто загальні описи задач ІС, такі як «пошук, обробка, зберігання інформації», «підвищення надійності зберігання даних» тощо. Формулювання задач ІС повинно виконуватися з урахуванням специфіки розглянутої предметної області. Наприклад:

1) Перегляд інформації про викладацький склад кафедри, закріплені за викладачами дисципліни і програмне забезпечення, необхідне для проведення занять.

2) Облік обладнання (комп'ютерів в аудиторіях) і встановленого на ньому програмного забезпечення.

3) Складання заявок на ремонт або установку ПЗ.

Користувачами ІС є фахівці в предметній області, для задоволення інформаційних потреб яких створюється ІС. ІС надає кожній категорії користувачів певний набір функцій, що дозволяє проводити маніпуляції з даними відповідно до політики доступу.

Визначення категорій користувачів, як і формулювання їхніх задач, повинно ґрунтуватися на специфіці аналізованої предметної області. В ІС зазвичай виділяються наступні основні категорії користувачів:

– адміністратор – виконує зміну даних предметної області, управляє списками користувачів ІС та їхніми привілеями на доступ до ІС;

– «звичайний» користувач – особа (або група осіб), яка виконує свої посадові обов'язки з використанням ІС.

Взагалі, назви категорій користувачів і їх кількість залежать від специфіки предметної області. Наприклад, у предметній області «Облік обладнання навчальних лабораторій кафедри», яка розглядається як приклад, виділяються наступні категорії користувачів:

1) Гість (студент) – має право на перегляд інформації про викладачів, навчальні дисципліни, аудиторії, комп'ютери і встановлене на них ПЗ (за винятком серійних номерів).

2) Співробітник кафедри – може переглядати всю інформацію, що зберігається в БД, крім серійних номерів ПЗ і ОС, змінювати стан обладнання (зламане або працює), створювати заявки на ремонт обладнання і установку ПЗ, а також додавати і змінювати інформацію про викладачів, дисципліни і ПЗ, необхідне для проведення навчальних занять з кожної дисципліни.

3) Адміністратор – може переглядати, додавати, редагувати і видаляти дані про викладачів і дисципліни, які вони ведуть, аудиторії, обладнання, програмне забезпечення, змінювати заявки на ремонт обладнання і установку ОС і ПЗ, а також управляє обліковими записами користувачів ІС.

Формулювання задач, що розв’язуються окремими категоріями користувачів за допомогою проектованої ІС, має виконуватися з метою визначення функціоналу, який надає ІС. Для кожної задачі повинні визначатися вхідні і вихідні дані. Вхідні дані – це інформація, яку користувач повинен ввести для виконання тої чи іншої задачі. Вихідні дані – це інформація, яку отримає користувач в результаті виконання задачі.

Список задач (з описом вхідних та вихідних даних) користувачів ІС «Облік обладнання навчальних лабораторій кафедри» наведений в наступному фрагменті прикладу розділу 1 пояснювальної записки з таблицею 1.1.

Таблиця 1.1 – Список задач користувачів ІС «Облік обладнання навчальних лабораторій кафедри»

Номер	Задача	Вхідні дані	Вихідні дані
	Адміністратор		
A1	Створити користувача	Логін, пароль, категорія	Новий користувач
A2	Редагувати користувача	Логін, нові пароль, категорія	Оновлена інформація про користувача
A3	Видалити користувача	Логін	Відсутні
A4	Додати співробітника	ПІБ співробітника, кафедра	Новий співробітник з присвоєним йому табельним номером
A5	Редагувати співробітника	Табельний номер, нові ПІБ, кафедра	Оновлена інформація про співробітника
A6	Видалити співробітника	Табельний номер	Відсутні
A7	Додати інформацію про обладнання	Назва обладнання, аудиторія, стан, параметри обладнання і його компонентів	Нове обладнання з присвоєним йому інвентарним номером
A8	Змінити інформацію про обладнання	Інвентарний номер, нові назва, аудиторія, стан, параметри обладнання і його компонентів	Оновлена інформація про обладнання
A9	Видалити інформацію про обладнання	Інвентарний номер	Відсутні
A10	Змінити заявку на установку ПЗ	Номер заявки, нові дата, ПІБ замовника, коментарі, пристрої, ПЗ, ОС, аудиторія	Оновлена інформація про заявку на установку ПЗ
A11	Змінити заявку на ремонт обладнання	Номер заявки, нові дата, ПІБ замовника, коментарі, пристрої, аудиторія	Оновлена інформація про заявку на ремонт обладнання
A12	Видалити заявку на ремонт обладнання або на установку ПЗ	Номер заявки	Відсутні
A13	Додати аудиторію	Назва	Нова аудиторія
A14	Редагувати аудиторію	Аудиторія, дані про яку змінюються, нова назва	Оновлена інформація про аудиторію
A15	Видалити аудиторію	Аудиторія, що видаляється	Відсутні
A16	Додати інформацію про ОС та ПЗ, встановлене на конкретному комп'ютері	Комп'ютер, ПЗ, ОС, шлях до ПЗ	Нова інформація про ОС та ПЗ, встановлені на конкретному комп'ютері
A17	Оновити інформацію про ПЗ, встановлене на конкретному комп'ютері	Комп'ютер, ПЗ, ОС, шлях до ПЗ	Оновлена інформація про ОС та ПЗ, встановлені на конкретному комп'ютері

Продовження таблиці 1.1

Номер	Задача	Вхідні дані	Вихідні дані
	Співробітник кафедри		
С1	Додати співробітника	ПІБ співробітника, кафедра	Новий співробітник з присвоєним йому табельним номером
С2	Переглянути інформацію про обладнання та його компоненти	Відсутні	Назва обладнання, аудиторія, стан, параметри обладнання і його компонентів, встановлені ПЗ і ОС
С3	Створити заявку на ремонт обладнання	Дата, ПІБ замовника, коментарі, пристрої, аудиторія	Нова заявка на ремонт обладнання з присвоєним їй номером
С4	Створити заявку на установку ПЗ	Дата, ПІБ замовника, коментарі, комп'ютер, ПЗ, ОС, аудиторія	Нова заявка на установку ПЗ з присвоєним їй номером
С5	Додати інформацію про дисципліну і необхідне для неї ПЗ	Викладач, дисципліна, ПЗ, кафедра	Нова інформація про дисципліну і необхідне для неї ПЗ
С6	Змінити інформацію про дисципліну і необхідне для неї ПЗ	Дисципліна, дані про яку змінюються, нові викладач, кафедра, дані про ПЗ	Оновлена інформація про дисципліну і необхідне для неї ПЗ
С7	Змінити стан обладнання	Інвентарний номер, новий стан обладнання (зламане/працює)	Оновлений стан обладнання
	Гість		
Г1	Переглянути інформацію про викладачів, навчальні дисципліни	Відсутні	Викладач (ПІБ), кафедра, дисципліна, ПЗ для проведення занять
Г2	Переглянути інформацію про обладнання та його компоненти	Відсутні	Назва обладнання, аудиторія, стан обладнання, параметри обладнання і його компонентів, встановлені ПЗ і ОС

ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

До даного розділу відносяться обґрунтування вибору архітектури ІС і шаблону проектування ПЗ.

В ІС, заснованих на застосуванні баз даних, логічно виділяються 3 основних шари:

1) шар представлення (призначеного для користувача інтерфейсу) – містить програмні засоби, за допомогою яких користувач може взаємодіяти з даними;

2) шар бізнес-логіки – містить програмні засоби, в яких відображаються правила, принципи і залежність поведінки об'єктів предметної області системи;

3) шар даних – містить програмні засоби, які надають дані застосункам, що їх обробляють.

Архітектура ІС визначає модель, структуру, виконувані функції і взаємозв'язок компонентів ІС. Найбільш часто використовується архітектура клієнт-сервер, згідно з концепцією якої ПЗ-споживач інформаційних послуг називається клієнтом, а ПЗ-постачальник інформаційних послуг – сервером. Як ПЗ серверної частини ІС, як правило, використовується СУБД, а в якості ПЗ клієнтської частини використовується клієнтська програма.

Сучасні ІС зазвичай будуються відповідно до N-рівневої архітектури клієнт-сервер, яка дозволяє створювати гнучкі і повторно використовувані застосунки: при поділі програми на рівні абстракції можливо замість переробки всього застосунку цілком вносити зміни лише в певний рівень. Найбільшого поширення набули дво- (рис. 1) і трирівнева (рис. 2) клієнт-серверні архітектури ІС.

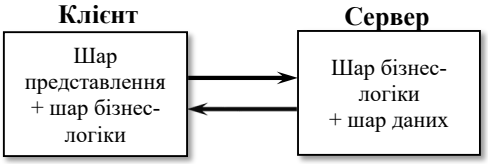


Рисунок 1 – Дворівнева архітектура клієнт-сервер

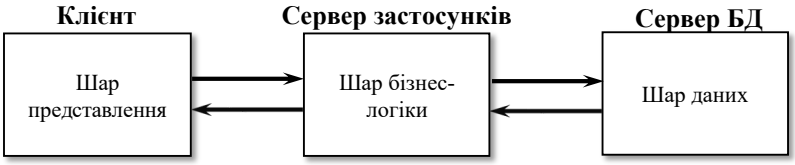


Рисунок 2 – Трирівнева архітектура клієнт-сервер

ПЗ всіх рівнів ІС, побудованої відповідно до N-рівневої архітектури, може бути встановлене на одному і тому ж комп'ютері або на різних комп'ютерах мережі.

При обґрунтуванні вибору архітектури ІС необхідно враховувати призначення проектованої ІС, програмно-апаратну інфраструктуру підприємства, установи або організації, для якої створюється ІС, кількість користувачів ІС, вимоги до безпеки даних тощо. Прийняте рішення слід аргументувати, вказавши переваги, які матиме обраний підхід в порівнянні з альтернативними.

Сучасний підхід до створення ІС передбачає використання **шаблонів проектування**, які надають ефективні рішення проблем, що виникають при проектуванні програмного забезпечення, припускають багаторазове використання коду, забезпечують надійність, дозволяють зменшити кількість помилок і прискорюють процес проектування. Найбільш часто використовуються шаблони MVC (Model-View-Controller – Модель-Представлення-Контролер) і MVP (Model-View-Presenter – Модель-Представлення-Представник).

При використанні шаблону MVC ІС ділиться на три окремі блоки (рис. 3):

1) Модель, яка здійснює маніпулювання даними застосунка, надання даних Представленню і реагування на команди Контролера шляхом зміни свого стану;

2) Представлення (призначений для користувача інтерфейс), яке відповідає за отримання необхідних даних з Моделі і відображення їх користувачеві, а також за отримання від користувача команд для роботи з даними і відправку цих команд Контролеру;

3) Контролер, який відповідає за перетворення команд користувача, одержуваних від Представлення, в набір дій над Моделлю з метою її зміни.

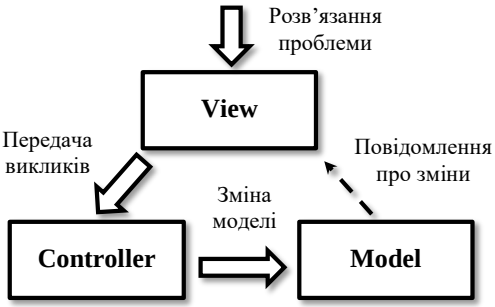


Рисунок 3 – Шаблон MVC

При використанні шаблону MVP, розробленого для поліпшення відділення логіки від відображення, ІС також ділиться на три окремі блоки (рис. 4):

1) Модель, яка здійснює маніпулювання даними застосунка, надання даних Представнику і реагування на команди Представника шляхом зміни свого стану;

2) Представлення, яке відповідає за відображення даних користувачеві (звертаючись при цьому за цими даними до Представника), а також за отримання від користувача команд для роботи з даними і відправку цих команд Представнику;

3) Представник, який відповідає за перетворення команд користувача, одержуваних від Представлення, в набір дій над Моделлю з метою її зміни, а

також виконує функції зміни Представлення при зміні стану Моделі, будучи своєрідним «посередником» між Моделлю і Представленням.

Ухвалення рішення про використання конкретного шаблону проектування необхідно обґрунтувати, зазначивши переваги, які надає використання вибраного шаблону в порівнянні з іншими.

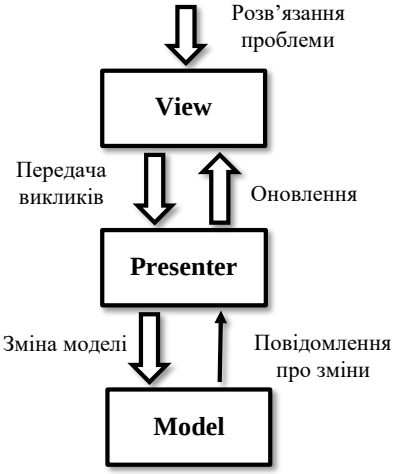


Рисунок 4 – Шаблон MVP

ІНФОРМАЦІЙНЕ МОДЕЛЮВАННЯ ПРЕДМЕТНОЇ ОБЛАСТІ

Даний розділ повинен містити опис концептуальної моделі предметної області, що включає:

- опис сутностей, їхніх атрибутів і обмежень цілісності;
- формалізацію зв'язків між сутностями;
- концептуальну модель предметної області у вигляді ER-діаграми.

Концептуальне моделювання предметної області виконується на основі функціональних вимог користувачів і абсолютно не залежить від будь-яких особливостей фізичної реалізації БД, таких як тип обраної СУБД або мови програмування.

Сутність – це реальний або уявлений об'єкт предметної області, інформація про який повинна зберігатися в БД. Кожна сутність характеризується певним наборів атрибутів, що відображає її властивості. Кожен атрибут має тип, який є абстракцією конкретного типу даних, що підтримується СУБД. У реляційній моделі даних сутності представляються відношеннями.

Сутність повинна мати один або кілька потенційних ключів, кожен з яких представляє собою ненадлишковий унікальний набір атрибутів. Один з потенційних ключів вибирається в якості первинного ключа (ідентифікатора).

Зв'язки являють собою асоціацію між сутностями і дозволяють по одній сутності знаходити інші сутності, пов'язані з нею. Зв'язки бувають безумовними, коли екземпляри обох сутностей беруть участь в зв'язку, і умовними, коли деякі екземпляри однієї або обох сутностей не приймають участь в зв'язку.

Кожен зв'язок може мати один з наступних типів:

– один-до-одного (1:1) (БУ – безумовний, У – умовний (необов'язковий, опціональний), 2У – біумовний) – один екземпляр першої сутності зв'язаний з одним екземпляром другої сутності;

– один-до-багатьох (1:N) (БУ, У, 2У) – один екземпляр першої сутності зв'язаний з декількома екземплярами другої сутності;

– багато-до-багатьох (M:N) (БУ, У, 2У) – один екземпляр першої сутності зв'язаний з декількома екземплярами другої сутності, і один екземпляр другої сутності зв'язаний з декількома екземплярами першої сутності.

Формалізація зв'язків відбувається наступним чином.

Якщо між двома сутностями існує безумовний зв'язок 1:1 (всі екземпляри обох сутностей беруть участь в зв'язку), то для їх представлення досить одного відношення, первинним ключем якого може бути ідентифікатор будь-якої з сутностей. Якщо кожну сутність необхідно представити у вигляді окремого відношення, то для формалізації зв'язку необхідно атрибути первинного ключа одного з відношень додати в схему іншого відношення у ролі зовнішнього ключа.

Якщо між двома сутностями існує умовний з одного боку зв'язок 1:1, то кожна сутність представляється окремим відношенням, і ідентифікатор сутності стає первинним ключем відповідного відношення, а ідентифікатор сутності, відповідної умовності зв'язку, додається у ролі зовнішнього ключа в схему відношення іншої сутності.

Якщо між двома сутностями існує зв'язок 1:1, умовний з двох сторін, то для формалізації зв'язку цього типу використовується три відношення: по одному для кожної сутності і одне для зв'язку (асоціативне), і в число атрибутів останнього відношення будуть входити первинні ключі двох інших. При цьому обидва ці ключі для асоціативного відношення є потенційними ключами, і, якщо один з них обирається у ролі первинного ключа асоціативного відношення, то другий автоматично вважається зовнішнім.

Для формалізації зв'язків типу 1:N не має значення, чи є зв'язок з боку однозв'язної сутності умовною або безумовною. Кожна сутність представляється окремим відношенням, при цьому первинний ключ відношення, що відповідає однозв'язній сутності, додається у ролі зовнішнього ключа в схему відношення N-зв'язкової сутності.

Для формалізації зв'язку 1:N, умовного з боку N-зв'язкової сутності,

формується по одному відношенню для кожної сутності і одне відношення для зв'язку, і в схему останнього відношення включаються в якості зовнішніх ключів первинні ключі двох інших відношення сутностей. Потенційними ключами асоціативного відношення можуть бути як ключ відношення N-зв'язкової сутності, так і множина, що складається з ключів обох відношень.

Зв'язок M:N будь-якої умовності також вимагає трьох відношень для формалізації: по одному для кожної сутності і одне для зв'язку, причому останнє повинно мати в схемі відношення первинні ключі двох інших (вони ж є зовнішніми ключами в асоціативному відношенні). Якщо при цьому асоціативне відношення не має власного атрибуту-ідентифікатора, то первинним ключем може бути лише множина, що містить обидва зовнішніх ключа.

Для формалізації n-арного зв'язку необхідно n+1 відношення: по одному для кожної сутності, яка бере участь в зв'язку, і одне для самого зв'язку. Причому схема асоціативного відношення повинна включати первинні ключі всіх інших n відношень, а множина всіх цих ключів буде первинним ключем даного відношення.

Цілісністю даних називається механізм підтримки відповідності бази даних предметної області, а обмеження цілісності покликані запобігати попаданню в БД неприпустимих даних. Приклади обмежень цілісності: потенційний ключ, зовнішній ключ, обмеження на значення, які може приймати конкретний атрибут.

Опис сутностей розглянутої предметної області, їх атрибутів і обмежень цілісності зручно представити у вигляді таблиці (див. табл. 1):

Таблиця 1 – Опис сутностей

Ім'я атрибута	Призначення атрибута	Обмеження
Ім'я сутності		
Ім'я атрибута 1	Призначення атрибута 1	Обмеження атрибута 1
Ім'я атрибута 2	Призначення атрибута 2	Обмеження атрибута 2
		Обмеження сутності

Інформацію про складені потенційні ключі можна розмістити в осередку «Обмеження сутності».

Приклад опису сутностей предметної області «Облік обладнання навчальних лабораторій кафедри» наведений в наступному фрагменті розділу 3 пояснювальної записки з таблицею 3.1.

Таблиця 3.1 – Опис сутностей предметної області «Облік обладнання навчальних лабораторій кафедри»

Ім'я атрибута	Призначення атрибута	Обмеження
Room (Аудиторія)		
id	ідентифікатор аудиторії	первинний ключ
number	Номер (назва) аудиторії	унікальне, не порожнє

Продовження таблиці 3.1

Ім'я атрибута	Призначення атрибута	Обмеження
Computer (Комп'ютер)		
computer_name	ім'я комп'ютера	первинний ключ
serial_number	серійний номер комп'ютера	унікальне, не порожнє
audit_id	ідентифікатор аудиторії	зовнішній ключ для зв'язку з сутністю Room (id), не порожнє
ip	IP-адреса комп'ютера	не порожнє
mac	MAC адреса комп'ютера	унікальне, не порожнє
status	стан комп'ютера	значення з множини ('істина' (працює), 'хиба' (не працює)), за замовчуванням 'істина', не порожнє
		унікальна комбінація (audit_id, ip)
Monitor (Монітор)		
id	ідентифікатор монітора	первинний ключ
model	модель монітора	не порожнє, унікальне
matrix	матриця монітора	значення з множини ('tft tn', 'tft va', 'tft ips', 'oled'), не порожнє
diagonal	діагональ монітора	значення> 0, не порожнє
resolution	роздільна здатність	не порожнє
status	стан монітора	значення з множини ('істина' (працює), хиба (не працює)), за замовчуванням 'істина', не порожнє
Monitor_And_Computer (Монітор і комп'ютер)		
serial_number	серійний номер комплекту	первинний ключ
computer_name	ім'я комп'ютера	зовнішній ключ для зв'язку з сутністю Computer (computer_name), не порожнє
monitor_id	ідентифікатор монітора	зовнішній ключ для зв'язку з сутністю Monitor (id), не порожнє
status	стан комплекту	значення з множини ('істина' (працює), 'хиба' (не працює)), за замовчуванням 'істина', не порожнє
price	вартість комплекту	значення> = 0, не порожнє
		унікальна комбінація (computer_name, monitor_id)
Component_Parts (Додаткове обладнання)		
id	ідентифікатор комплектуючого	первинний ключ
model	модель комплектуючого	не порожнє, унікальне
type	тип комплектуючого	значення з множини ('cpu', 'gpu', 'motherboard', 'ram', 'soundcard', 'hdd'), не порожнє
capacity	об'єм пам'яті комплектуючого	значення може бути порожнім
Component_In_Computer (Комплектуючі в комп'ютері)		
computer_name	ім'я комп'ютера	зовнішній ключ для зв'язку з сутністю Computer (computer_name), не порожнє

Продовження таблиці 3.1

Ім'я атрибута	Призначення атрибута	Обмеження
component_id	ідентифікатор комплектуючого	зовнішній ключ для зв'язку з сутністю Component_Parts (id), не порожнє
count	кількість комплектуючих (одного типу)	значення> = 0, за замовчуванням 1, не порожнє
locked	заборона на видалення /зміну даних про комплектуюче комп'ютера	значення з множини ('істина' (заборонено), 'хиба' (заборони немає)), за замовчуванням 'істина', не порожнє
		первинний ключ (computer_name, component_id)
OS (Операційна система (ОС))		
id	ідентифікатор ОС	первинний ключ
name	назва ОС	не порожнє
serial_os	серійний номер ОС	не порожнє
type	тип ОС	значення з множини ('Windows', 'Linux'), не порожнє
Software (Програмне забезпечення)		
id	ідентифікатор ПЗ	первинний ключ
name	назва ПЗ	унікальне, не порожнє
cost	вартість ПЗ	значення> = 0, за замовчуванням 0, не порожнє
Request_For_Software_Installation (Заявка на установку ПЗ)		
id	ідентифікатор заявки	первинний ключ
fio	ПІБ замовника	не порожнє
commentary	коментарі до заявки	не порожнє
computer_name	ім'я комп'ютера	зовнішній ключ для зв'язку з сутністю Computer (computer_name), не порожнє
os_id	ідентифікатор ОС	зовнішній ключ для зв'язку з сутністю OS (id), не порожнє
program_id	ідентифікатор ПЗ	зовнішній ключ для зв'язку з сутністю Software (id), не порожнє
date	дата заявки	не порожнє
Cathedra (Кафедра)		
id	ідентифікатор кафедри	первинний ключ
name	назва кафедри	не порожнє
Teacher (Викладач)		
id	ідентифікатор викладача	первинний ключ
first_name	ім'я викладача	не порожнє
second_name	прізвище викладача	не порожнє
patronymic	по батькові викладача	не порожнє
cathedra_id	ідентифікатор кафедри	зовнішній ключ для зв'язку з сутністю Cathedra (id), не порожнє
Course (Дисципліна)		
id	ідентифікатор дисципліни	первинний ключ
course_name	назва дисципліни	не порожнє

Продовження таблиці 3.1

Ім'я атрибута	Призначення атрибута	Обмеження
Course_And_Software (Дисципліна і необхідне для неї ПЗ)		
id	ідентифікатор запису	первинний ключ
audit_id	ідентифікатор аудиторії	зовнішній ключ для зв'язку з сутністю Room (id), не порожнє
course_id	ідентифікатор дисципліни	зовнішній ключ для зв'язку з сутністю Course (id), не порожнє
program_id	ідентифікатор ПЗ	зовнішній ключ для зв'язку з сутністю Software (id), не порожнє
teacher_id	ідентифікатор викладача	зовнішній ключ для зв'язку з сутністю Teacher (id), не порожнє
		унікальна комбінація (course_id, program_id, teacher_id, audit_id)
Network_Device (Мережевий пристрій)		
id	ідентифікатор мережевого пристрою	первинний ключ
model	модель мережевого пристрою	не порожнє
total_ports	кількість LAN портів мережевого пристрою	значення> 0, не порожнє
type	тип мережевого пристрою	значення з множини ('router', 'switch', 'hub', 'repeater'), не порожнє
NetDevice_In_Room (Мережевий пристрій в аудиторії)		
id	ідентифікатор запису	первинний ключ
netdevice_id	ідентифікатор мережевого пристрою	зовнішній ключ для зв'язку з сутністю Network_Device (id), не порожнє
audit_id	ідентифікатор аудиторії	зовнішній ключ для зв'язку з сутністю Room (id), не порожнє
busy_ports	всього зайнято портів	значення> = 0, не порожнє
free_ports	всього вільно портів	значення> = 0, не порожнє
ip	IP-адреса мережевого пристрою	не порожнє, унікальне
mac	MAC-адреса мережевого пристрою	не порожнє, унікальне
status	стан мережевого пристрою	значення з множини ('істина' (працює), 'хиба' (не працює)), не порожнє
price	закупівельна вартість	значення> = 0, не порожнє
		унікальна комбінація (netdevice_id, audit_id)
Printer_Scanner (Принтер/сканер)		
id	ідентифікатор принтера/сканера	первинний ключ
model	модель принтера/сканера	не порожнє
type	тип	значення з множини ('printer', 'scanner'), не порожнє

Продовження таблиці 3.1

Ім'я атрибута	Призначення атрибута	Обмеження
RrSc_er_In_Room (Принтер/сканер в аудиторії)		
id	ідентифікатор запису	первинний ключ
prsc_er_id	ідентифікатор принтера/сканера	зовнішній ключ для зв'язку з сутністю Printer_Scanner (id), не порожнє
audit_id	ідентифікатор аудиторії	зовнішній ключ для зв'язку з сутністю Room (id), не порожнє
status	стан принтера/сканера	значення з множини ('істина' (працює), 'хиба' (не працює)), не порожнє
price	закупівельна вартість	значення> = 0, не порожнє
		унікальна комбінація (prsc_er_id, audit_id)
Projector (Проектор)		
id	ідентифікатор проектора	первинний ключ
model	модель проектора	не порожнє
Projector_In_Room (Проектор в аудиторії)		
id	ідентифікатор запису	первинний ключ
projector_id	ідентифікатор проектора	зовнішній ключ для зв'язку з сутністю Projector (id), не порожнє
audit_id	ідентифікатор аудиторії	зовнішній ключ для зв'язку з сутністю Room (id), не порожнє
status	стан проектора	значення з множини ('істина' (працює), 'хиба' (не працює)), не порожнє
price	закупівельна вартість	значення> = 0, не порожнє
		унікальна комбінація (projector_id, audit_id)
Appliances (Побутова техніка)		
id	ідентифікатор побутової техніки	первинний ключ
model	модель побутової техніки	не порожнє
Appliances_In_Room (Побутова техніка в аудиторії)		
id	ідентифікатор запису	первинний ключ
appliances_id	ідентифікатор побутової техніки	зовнішній ключ для зв'язку з сутністю Appliances (id), не порожнє
audit_id	ідентифікатор аудиторії	зовнішній ключ для зв'язку з сутністю Room (id), не порожнє
status	стан побутової техніки	значення з множини ('істина' (працює), 'хиба' (не працює)), не порожнє
price	закупівельна вартість	значення> = 0, не порожнє
		унікальна комбінація (appliances_id, audit_id)
Keyboard_Mouse (Клавіатура/миша)		
id	ідентифікатор клавіатури/миші	первинний ключ
model	модель клавіатури/миші	не порожнє
type	тип клавіатури/миші	значення з множини ('keyboard', 'mouse'), не порожнє

Продовження таблиці 3.1

Ім'я атрибута	Призначення атрибута	Обмеження
KeyMse_In_Room (Клавіатура/миша в аудиторії)		
id	ідентифікатор запису	первинний ключ
keymse_id	ідентифікатор клавіатури/миші	зовнішній ключ для зв'язку з сутністю Keyboard_Mouse (id), не порожнє
audit_id	ідентифікатор аудиторії	зовнішній ключ для зв'язку з сутністю Room (id), не порожнє
status	стан клавіатури/миші	значення з множини ('істина' (працює), 'хиба' (не працює)), не порожнє
price	закупівельна вартість	значення> = 0, не порожнє
		унікальна комбінація (keymse_id, audit_id)
Cable (Кабелі)		
id	ідентифікатор кабелю	первинний ключ
type	тип кабелю	не порожнє
Cable_In_Room (Кабелі в аудиторії)		
id	ідентифікатор запису	первинний ключ
cable_id	ідентифікатор кабелю	зовнішній ключ для зв'язку з сутністю Cable (id), не порожнє
audit_id	ідентифікатор аудиторії	зовнішній ключ для зв'язку з сутністю Room (id), не порожнє
length	довжина кабелю	значення> = 0, не порожнє
price	закупівельна вартість	значення> = 0, не порожнє
status	стан кабелю	значення з множини ('істина' (працює), 'хиба' (не працює)), не порожнє
		унікальна комбінація (cable_id, audit_id)
Request_For_Repair (Заявка на ремонт)		
id	ідентифікатор заявки	первинний ключ
date	дата заявки	не порожнє
fio	ПІБ замовника	не порожнє
commentary	коментарі до заявки	не порожнє
computer_name	ім'я комп'ютера	зовнішній ключ для зв'язку з сутністю Computer (computer_name)
component_id	ідентифікатор комплектуючого	зовнішній ключ для зв'язку з сутністю Component_In_Computer (id)
monitor_id	ідентифікатор монітора	зовнішній ключ для зв'язку з сутністю Monitor (id)
netdevice_id	ідентифікатор мережевого пристрою	зовнішній ключ для зв'язку з сутністю Network_Device (id)
keymse_id	ідентифікатор клавіатури/миші	зовнішній ключ для зв'язку з сутністю KeyMse_In_Room (id)
prsc_er_id	ідентифікатор принтера/сканера	зовнішній ключ для зв'язку з сутністю RrSc_er_In_Room (id)
projector_id	ідентифікатор проектора	зовнішній ключ для зв'язку з сутністю Projector_In_Room (id)

Продовження таблиці 3.1

Ім'я атрибута	Призначення атрибута	Обмеження
appliances_id	ідентифікатор побутової техніки	зовнішній ключ для зв'язку з сутністю Appliances_In_Room (id)
cable_id	ідентифікатор кабелю	зовнішній ключ для зв'язку з сутністю Cable_In_Room (id)
OS_and_Software (Програмне забезпечення в ОС)		
computer_name	ім'я комп'ютера	зовнішній ключ для зв'язку з сутністю Computer (computer_name)
program_id	ідентифікатор ПЗ	зовнішній ключ для зв'язку з сутністю Software (id)
os_id	ідентифікатор ОС	зовнішній ключ для зв'язку з сутністю OS (id)
path_exe	розташування (шлях до програми на комп'ютері)	не порожнє
locked	заборона на видалення/зміни даних про ПЗ комп'ютера	значення з множини ('істина' (заборонено), 'хиба' (заборони немає)), за замовчуванням 'істина', не порожнє
serial_key	серійний номер ПЗ	не порожнє
		первинний ключ (computer_name, program_id, os_id)
users (Користувачі)		
login	логін користувача	первинний ключ, не порожнє
password	пароль користувача	не порожнє
category	роль користувача при підключенні до БД	значення з множини ('administrator', 'employee', 'guest'), не порожнє

Далі наведені приклади опису формалізації зв’язків між сутностями, описаними в вищенаведеному прикладі у табл. 3.1.

1) Зв'язок між кафедрами та викладачами. На одній кафедрі працює кілька викладачів. З іншого боку, один і той же викладач працює лише на одній кафедрі. Отже, між кафедрами та викладачами існує зв'язок один-до-багатьох, безумовний з обох боків. Для формалізації цього зв'язку ідентифікатор кафедри доданий в таблицю викладачів у вигляді зовнішнього ключа.

2) Зв'язок між комп'ютерами і моніторами. До одного комп'ютера в комплекті повинен обов'язково йти один або кілька моніторів. З іншого боку, один і той же монітор може йти в комплекті з одним комп'ютером. Отже, між комп'ютерами і моніторами існує зв'язок один-до-багатьох, безумовний з обох боків. Для формалізації цього зв'язку додана проміжна таблиця Monitor_and_Computer, в яку поміщені первинні ключі таблиць Monitor і Computer, що є в проміжній таблиці зовнішніми ключами.

3) Зв'язок між комп'ютерами і комплектуючими. Один комп'ютер складається з декількох комплектуючих. З іншого боку, однакові комплектуючі можуть бути встановлені на різних комп'ютерах. Виходячи з того, що «вільні» комплектуючі на кафедрі не значаться (всі вони знаходяться в розпорядженні інформаційного центру), кожне комплектуюче має обов'язково бути встановлено. Отже, між комплектуючими і комп'ютерами існує зв'язок

багато-до-багатьох, який є безумовним з обох боків. Для формалізації цього зв'язку додана проміжна таблиця Component_in_Computer, в яку поміщені первинні ключі таблиць Component_Parts і Computer, що є в проміжній таблиці зовнішніми ключами, а їх комбінація – первинним ключем.

4) Зв'язок між аудиторіями, викладачами, дисциплінами і ПЗ. В одній і тій же аудиторії можуть проводитися заняття по одній і тій же дисципліні або з різних дисциплін, заняття можуть вести як один викладач, так і різні, ПЗ також може бути як одне і те ж для однієї дисципліни, так і різне (наприклад, на різних спеціальностях для однієї дисципліни різні робочі програми). З іншого боку, одна і та ж дисципліна може викладатися в різних аудиторіях одним і тим же викладачем або різними викладачами з використанням одного і того ж або різного ПЗ. Отже, між аудиторіями, викладачами, дисциплінами і ПЗ існує n-арний зв'язок багато-до-багатьох (n=4). Для формалізації цього зв'язку додана проміжна таблиця Course_And_Software, в яку поміщені первинні ключі таблиць Teacher, Course, Room і Software, що є в проміжній таблиці зовнішніми ключами, а їх комбінація – потенційним ключем.

5) Зв'язок між комп'ютерами, ОС і ПЗ. Для однієї і тієї ж ОС може існувати безліч програм. З іншого боку, одна і та ж програма може бути призначена для використання в різних ОС. Крім цього, на одному комп'ютері може бути встановлено кілька ОС і, відповідно, кілька програм. З іншого боку, одна і та ж програма може бути встановлена на декількох ОС і на декількох комп'ютерах. Отже, між ОС і ПЗ існує безумовний n-арний зв'язок багато-до-багатьох (n=3). Для формалізації цього зв'язку додана проміжна таблиця OS_and_Software, в яку поміщені первинні ключі таблиць Computer, OS і Software, що є в проміжній таблиці зовнішніми ключами, а їх комбінація – первинним ключем.

6) Зв'язок між ПЗ і заявками на установку ПЗ на комп'ютери. На установку одного і того ж ПЗ на різні комп'ютери можна оформити декілька заявок в різний час. З іншого боку, одна і та ж заявка може бути оформлена на установку лише одного ПЗ. Отже, між ПЗ і заявками на установку ПЗ існує безумовний зв'язок один-до-багатьох. Для формалізації цього зв'язку в таблицю заявок на установку ПЗ Request_For_Software_Installation поміщений первинний ключ таблиці Software, що є в таблиці заявок зовнішнім ключем для зв'язку з таблицею ПЗ. Крім цього, в таблицю заявок в якості зовнішніх ключів також поміщені первинні ключі таблиць Computer і OS для зазначення того, на який комп'ютер і на яку ОС необхідно встановити відповідне ПЗ.

Одним із засобів графічного представлення предметної області є діаграма «сутність-зв'язок» (ER-діаграма, ERD). Для візуалізації ER-діаграм існує ряд нотацій, таких як нотація Чена, нотація Баркера, нотація «воронячі лапки» («Crow's feet»), нотація Microsoft Access (використовувана в однойменній СУБД), UML і т. ін. Для побудови ER-діаграми можна скористатися різними програмними засобами, наприклад, Valentina Studio, Navicat, Microsoft Visual Studio, Microsoft Access, StarUML і багатьма іншими. Ці продукти використовують різні нотації, які іноді дещо відрізняються від загальноприйнятих.

Фрагмент ER-діаграми для предметної області «Облік обладнання навчальних лабораторій кафедри» наведений на рис. 5. Для побудови ER-діаграми використаний програмний продукт Navicat.

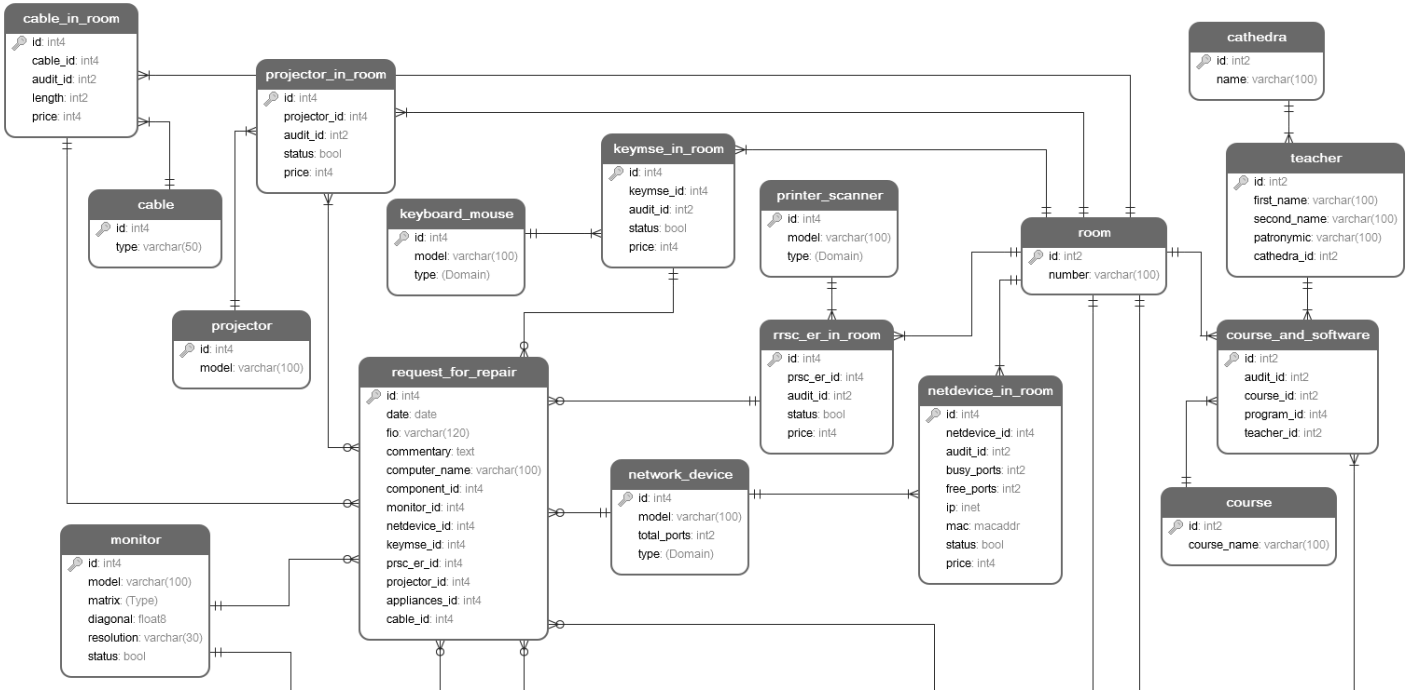


Рисунок 5 – Фрагмент ER-діаграми для предметної області «Облік обладнання навчальних лабораторій кафедри»

ПРОГРАМНА МОДЕЛЬ ЗАСТОСУНКА

Даний розділ присвячений побудові моделі програми, створюваної в рамках курсового проектування.

Під побудовою моделі мається на увазі декомпозиція застосунку на підсистеми (функціональні модулі, сервіси, класи, підпрограми) і організація їх взаємодії один з одним і з зовнішнім світом. При цьому застосунок являє собою конструктор, що складається з набору модулів/сервісів/класів/підпрограм, які взаємодіють один з одним по добре визначеним і простим правилам. Подібна організація застосунку забезпечує його гнучкість, розширюваність і масштабованість.

Наприклад, при використанні об'єктно-орієнтованого підходу до моделювання для представлення моделі рівня прикладного компонента і рівня управління ресурсами програми використовуються діаграми класів UML, які описують внутрішню структуру об'єктів предметної області (атрибути, методи), типи зв'язків та інтерфейси між ними.

У пояснювальній записці необхідно привести діаграми класів рівня прикладного компонента (Controller, Presenter) і рівня управління ресурсами (Model) для підсистем всіх користувачів створюваної ІС. Діаграми класів необхідно супроводити поясненнями, що містять короткий опис класів, їхніх атрибутів і методів, а також зв'язків між класами і інтерфейсів.

Також в даному розділі необхідно привести модель рівня представлення (View) даних у вигляді ієрархії екранних форм (у разі створення настільного застосунку) або веб-сторінок (в разі створення веб-застосунку), яка відображає послідовність переходу користувачів за формами (веб-сторінками) застосунку для виконання відповідних задач. Ієрархії форм (веб-сторінок) повинні бути наведені для підсистем всіх користувачів.

На рис. 6 наведений приклад ієрархії форм для підсистеми користувача Адміністратор інформаційної системи «Облік обладнання навчальних лабораторій кафедри».

Рисунок 6 – Ієрархія форм підсистеми користувача Адміністратор ІС «Облік обладнання навчальних лабораторій кафедри»

ВИБІР ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ СТВОРЕННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

Даний розділ присвячений обґрунтуванню вибору СУБД, технологій, середовищ розробки, мов програмування, допоміжних бібліотек, фреймворків для створення ІС.

При обґрунтуванні вибору засобів програмування необхідно враховувати такі фактори, як фінансові витрати на їх придбання, вимоги до апаратно-програмної платформи, підтримувані можливості, продуктивність, надійність, масштабованість, підтримка обраного шаблону проектування ПЗ, швидкість розробки ПЗ, наявність документації, особливості взаємодії з іншими використовуваними засобами.

Отже, у цьому розділі пояснювальної записки потрібно навести аргументи на користь використання обраних засобів для створення ІС в обраній предметній області. Приклад фрагменту розділу 5 пояснювальної записки наведений нижче.

Як засоби реалізації ІС для предметної області «Облік обладнання навчальних лабораторій кафедри» обрані наступні інструменти:

- СУБД PostgreSQL;
- мова програмування C#;
- середовище розробки Visual Studio 2019 Professional Edition;
- модулі інтерфейсу Windows Forms;
- бібліотека класів Npgsql для роботи з СУБД PostgreSQL.

Крім того, для отримання інформації про комплектуючі комп'ютерів використана утиліта CPU-Z.

СТВОРЕННЯ БАЗИ ДАНИХ

Даний розділ повинен містити тексти SQL-запитів для створення доменів, типів даних, послідовностей, таблиць, уявлень, тригерів, збережених процедур, функцій і т. ін., супроводжувані докладними коментарями. При цьому немає необхідності розглядати код створення всіх об'єктів БД, досить навести кілька прикладів, які наочно демонструють можливості мови запитів SQL для розв'язання задачі створення БД в предметної області, що розглядається.

Забезпечені коментарями тексти SQL-запитів, використані для створення всіх об'єктів БД, доцільно оформити у вигляді одного або декількох окремих додатків, прикладами яких є додатки Б і В.

Нижче наведений приклад тексту розділу 6 пояснювальної записки для предметної області «Облік обладнання навчальних лабораторій кафедри».

Детально розглянемо запит на створення таблиці Monitor. Для визначення одного з її атрибутів знадобиться створити перелічувальний тип даних:

```
CREATE TYPE matrix_m AS ENUM ( 'TFT TN', 'TFT VA', 'TFT IPS', 'OLED');
```

Створення типу відбувається за допомогою команди `CREATE TYPE`, далі вказується ім'я типу. За допомогою `ENUM (...)` перераховані допустимі значення для атрибутів даного типу.

Для створення наборів значень не обов'язково створювати типи даних, замість них можна використовувати домени:

```
CREATE DOMAIN type_cp VARCHAR (50)
DEFAULT 'CPU'
CONSTRAINT type_check
CHECK (VALUE IN ( 'CPU', 'GPU', 'Motherboard', 'RAM', 'Soundcard', 'HDD'));
```

Створення домену відбувається за допомогою команди `CREATE DOMAIN`, далі вказується ім'я домену. За замовчуванням (`DEFAULT`) поле даного типу буде мати значення 'CPU', допустимі значення задаються за допомогою `CHECK` серед запропонованих варіантів: 'CPU', 'GPU', 'Motherboard', 'RAM', 'Soundcard' і 'HDD'.

Для створення таблиці Monitor використовується наступний SQL-запит:

```
CREATE TABLE Monitor (
      id serial PRIMARY KEY,
      model varchar (100) UNIQUE NOT NULL,
      matrix matrix_m NOT NULL,
      diagonal double precision CHECK (diagonal>0) NOT NULL,
      resolution varchar (30) NOT NULL
      status boolean DEFAULT true NOT NULL);
```

Створення таблиці відбувається за допомогою команди `CREATE TABLE`, далі вказується ім'я таблиці. Кожному полю таблиці зіставляються тип даних обраної СУБД (PostgreSQL) і обмеження цілісності. Поле `id` є первинним ключем (`PRIMARY KEY`) і має тип `serial` (цей тип не є справжнім типом даних, а являє собою зручний спосіб створення унікального ідентифікатора шляхом збільшення попереднього значення в даному стовпці на вказане значення, за замовчуванням на 1). Поле `matrix` має раніше визначений тип `matrix_m`. `NOT NULL` при визначенні полів таблиці означає, що вони не можуть містити порожні значення. Запис `DEFAULT true` при визначенні поля `status` типу `boolean` означає, що дані у полі `status` за замовчуванням приймають значення `true`, тим самим позначаючи стан монітора як робочий. `CHECK (diagonal>0)` при визначенні поля `diagonal` вказує, що значення у цьому полі має бути більше нуля. `UNIQUE` при визначенні поля `model` вказує на те, що значення у даному полі не повинні повторюватися.

Розглянемо ще два запити на створення таблиць (лістинг 6.1).

```
CREATE TABLE Cathedra (
      id smallserial PRIMARY KEY,
      name varchar (100) NOT NULL);

CREATE TABLE Teacher (
      id smallserial PRIMARY KEY,
      first_name varchar (100) NOT NULL,
      second_name varchar (100) NOT NULL,
      patronymic varchar (100) NOT NULL,
      cathedra_id smallint REFERENCES Cathedra (id) NOT NULL);
```

Лістинг 6.1 – Запити на створення таблиць `Cathedra` та `Teacher`

Перший з них демонструє створення батьківської таблиці `Cathedra`, а другий – створення дочірньої таблиці `Teacher`, в якій поле `cathedra_id` визначене як зовнішній ключ або посилання (`REFERENCES`) з метою забезпечення зв'язку з полем `id` таблиці `Cathedra`.

Тексти програм SQL-запитів, які використовуються для створення всіх об'єктів БД, наведені в додатках Б і В.

ЗАПИТИ ДО БАЗИ ДАНИХ ДЛЯ РОЗВ'ЯЗАННЯ ПОСТАВЛЕНИХ ЗАДАЧ

В даному розділі повинні міститися тексти SQL-запитів, необхідних для розв'язання задач користувачів інформаційної системи, або представлення цих запитів за допомогою засобів обраної мови програмування (із зазначенням тексту SQL-запитів, отримуваних при трансляції).

При описі рішень задач користувачів слід використовувати такі шаблони:

1) Для розв'язання задач A*x*-A*y* і C*m* необхідні запити типу `SELECT стовпець1, стовпець2, стовпець3 FROM таблиця [WHERE умова]` з підстановкою відповідних імен таблиць і стовпців (зазначити, яких саме) і, у разі потреби, умов (вказати, яких саме).

2) Для розв'язання задачі B*k* і B*p* необхідний запит `UPDATE таблиця SET стовпець = вираз | значення [WHERE умова]` з підстановкою відповідних імен таблиць і стовпців (зазначити, яких саме), виразів або безпосередніх значень (зазначити, яких саме) і, у разі потреби, умов (вказати, яких саме).

3) Для розв'язання задачі C*z* створена збережена процедура (функція) `ім'я_процедури_(функції)` (див. додаток В), для звернення до якої необхідний запит `SELECT * FROM ім'я_процедури_(функції) (параметри)`. Необхідно вказати ім'я процедури (функції) і перелічити параметри, з якими вона викликається.

4) Для розв'язання задачі C*q* необхідний тригер `імя_тригера` (див. додаток В), що спрацьовує при генерації запиту `DELETE FROM таблиця [WHERE умова]`. Необхідно вказати як ім'я таблиці, так і умови виконання запиту.

Нижче наведені приклади опису розв'язань задач користувачів інформаційної системи «Облік обладнання навчальних лабораторій кафедри» у розділі 7 пояснювальної записки.

1) Для розв'язання задач А6, А9, А12, А15 використовується запит типу `DELETE FROM таблиця where id = значення;` При цьому для розв'язання задачі А6 замість параметра *`таблиця`* підставляється ім'я таблиці `Teachers`, для розв'язання задачі А9 – ім'я таблиці `Monitor` або `Network_Device` або `Printer_Scanner` або `Projector` або `Appliances` або `Keyboard_Mouse` або `Cable`, для розв'язання задачі А12 – ім'я таблиці `Request_For_Repair` або `Request_For_Software_Installation`, а для розв'язання задачі А15 – ім'я таблиці `Room`. Значення ідентифікатора примірника об'єкта, що видаляється, передається з відповідного елемента управління користувацького інтерфейсу.

2) Для розв'язання задачі Г1 необхідний наступний SQL-запит:

```
SELECT course_name AS 'Дисципліна',
      concat (first_name, '', second_name, '', patronymic) AS 'Викладач',
      number AS 'Аудиторія', software.name as 'ПО'
FROM teacher, course, cathedra, course_and_software, room, software
WHERE room.id = course_and_software.audit_id
      AND teacher.id = course_and_software.teacher_id
      AND course.id = course_and_software.course_id
      AND teacher.cathedra_id = cathedra.id
      AND course_and_software.program_id = software.id;
```

3) Для розв'язання задачі А16 створена функція `AddSoftwareToPc` (див. додаток В), яка викликається при виконанні запиту

```
SELECT * FROM AddSoftwareToPc ('ws40-01', 'Mozilla Firefox', 'Windows 10',
      'C:\Program Files\Mozilla');
```

4) Для розв'язання задач А1 і А2 додатково створений тригер `make_password_hash` (див. додаток В), який при оновленні або додаванні записів в таблицю `users` викликає тригерну функцію `hasg_password ()`, що здійснює шифрування пароля користувача. Тригер спрацьовує при виконанні запитів `INSERT INTO users VALUES ( 'ivanov', 'Qwerty1', 'administrator');` або `UPDATE users SET password = 'Qwerty1' WHERE user = 'ivanov';`

Тексти створених представлень, тригерів і збережених процедур (функцій), супроводжувані коментарями, доцільно оформити у вигляді окремого додатка.

У додатку В наведені приклади опису тригера і збереженої процедури.

ПРОГРАМНА РЕАЛІЗАЦІЯ ІНТЕРФЕЙСУ

В даному розділі необхідно навести вихідний код функцій, написаних обраною мовою програмування, що виконують типові операції в створеному застосунку, наприклад, підключення до БД, виведення вмісту таблиці на екран, виконання зміни даних в БД (додавання, редагування, видалення), обробка пошукових запитів з використанням різних параметрів пошуку тощо. Програмний код повинен супроводжуватися детальними коментарями.

При описі типових функцій можна використовувати шаблони, подібні до тих, які використовуються при описі рішень задач користувачів ІС (див. попередній розділ).

Вихідний код основних класів, також супроводжуваний коментарями, доцільно оформити як окремий додаток.

БЕЗПЕКА ІНФОРМАЦІЙНОЇ СИСТЕМИ

Даний розділ повинен містити опис засобів забезпечення безпеки на рівні сервера БД і застосунку, таких як:

- ролі користувачів та їхні привілеї на доступ до об'єктів БД (таблиць, представлень, функцій тощо);
- механізм зберігання паролів;
- спосіб визначення ролі користувача при введенні логіна і пароля.

Нижче наведений приклад тексту розділу 9 пояснювальної записки для предметної області «Облік обладнання навчальних лабораторій кафедри».

Безпека на рівні БД забезпечується шляхом створення ролей і наділення їх відповідними привілеями. В ІС «Облік обладнання навчальних лабораторій кафедри» реалізовано три ролі: Адміністратор, Співробітник і Гість. У таблиці 9.1 наведені привілеї ролей на таблиці БД. При цьому використані такі позначення: C (Create) – створення (додавання) даних, R (Read) – читання даних, U (Update) – оновлення даних, D (Delete) – видалення даних, E (Execute) – виконання процедури (функції).

Таблиця 9.1 – Привілеї ролей на об'єкти БД

Об'єкти БД	Ролі		
	Адміністратор (administrator)	Співробітник (employee)	Гість (guest)
Таблиці			
Room	CRUD	R	R
Computer	CRUD	R	R
Monitor	CRUD	R	
Monitor_And_Computer	CRUD	RU	
Component_Parts	CRUD	R	
Component_In_Computer	CRUD	RU	
OS	CRUD	R	R
Software	CRUD	R	R
Request_For_Software_Installation	CRUD	CR	
Cathedra	CRUD	R	R
Teacher	CRUD	CRU	R
Course	CRUD	CRU	R
Course_And_Software	CRUD	CRU	R
Network_Device	CRUD	R	
NetDevice_In_Room	CRUD	RU	
Printer_Scanner	CRUD	R	
RrSc_er_In_Room	CRUD	RU	
Projector	CRUD	R	
Projector_In_Room	CRUD	RU	
Appliances	CRUD	R	
Appliances_In_Room	CRUD	RU	
Keyboard_Mouse	CRUD	R	
KeyMse_In_Room	CRUD	RU	
Cable	CRUD	R	
Cable_In_Room	CRUD	RU	
Request_For_Repair	CRUD	CR	
OS_and_Software	CRUD	R	R
users	CRUD	R	

Продовження таблиці 9.1

Об'єкти БД	Ролі		
	Адміністратор (administrator)	Співробітник (employee)	Гість (guest)
Функції			
AddSoftwareToPc	E		
hasg_password	E		

Створення ролей і наділення їх привілеями відповідно до наведеної вище таблиці здійснюється за допомогою наступних SQL-запитів (лістинги 9.1-9.3):

```
CREATE ROLE guest WITH LOGIN PASSWORD 'guest12345';
GRANT SELECT ON Room, Computer, OS, Software, Cathedra, Teacher, Course,
Course_And_Software, OS_and_Software, users IN SCHEMA lab_equipment TO guest;
```

Лістинг 9.1 – Створення і наділення привілеями ролі Гість

```
CREATE ROLE employee WITH LOGIN PASSWORD 'employee54321';
GRANT SELECT ON ALL TABLES IN SCHEMA lab_equipment TO employee;
GRANT INSERT ON Request_For_Software_Installation, Teacher, Course,
Course_And_Software, Request_For_Repair IN SCHEMA lab_equipment TO employee;
GRANT UPDATE ON Monitor_And_Computer, Component_In_Computer, Course_And_Software,
NetDevice_In_Room, RrSc_er_In_Room, Teacher, Course, Projector_In_Room,
Appliances_In_Room, KeyMse_In_Room, Cable_In_Room IN SCHEMA lab_equipment TO
employee;
```

Лістинг 9.2 – Створення і наділення привілеями ролі Співробітник

```
CREATE ROLE administrator WITH LOGIN PASSWORD 'admin975312468';
GRANT INSERT, UPDATE, SELECT, DELETE ON ALL TABLES IN SCHEMA lab_equipment TO
administrator;
GRANT EXECUTE ON ALL FUNCTIONS IN SCHEMA lab_equipment TO administrator;
```

Лістинг 9.3 – Створення і наділення привілеями ролі Адміністратор

Всім створеним ролям необхідно надати доступ до схеми `lab_equipment`, щоб користувачі мали змогу використовувати об'єкти в ній:

```
GRANT USAGE ON SCHEMA lab_equipment TO guest, employee, administrator;
```

Також створена службова роль `connect_user`, від імені якої виконується підключення до БД для виконання наступних дій:

- перевірки існування в таблиці `users` облікового запису користувача, що проходить процедуру автентифікації в застосунку;
- визначення того, яка роль з перерахованих вище відповідає цьому обліковому запису.

Роль `connect_user` не володіє ніякими привілеями, крім доступу до схеми `lab_equipment` і можливості здійснювати вибірку даних з таблиці `users` (лістинг 9.4):

```
CREATE ROLE connect_user WITH LOGIN PASSWORD 'c0nneCT';
GRANT USAGE ON SCHEMA lab_equipment TO connect_user;
GRANT SELECT ON users IN SCHEMA lab_equipment TO connect_user;
```

Лістинг 9.4 – Створення і наділення привілеями ролі connect_user

Паролі користувачів зберігаються в полі `password` таблиці `users` в зашифрованому вигляді. Шифрування паролів при створенні і редагуванні користувачів здійснюється за допомогою тригерної функції `hasg_password ()`, що викликається при спрацьовуванні тригера `make_password_hash` (див. додаток В).

Визначення ролі користувача відбувається наступним чином. Після того як користувач в застосунку ввів логін і пароль, для перевірки існування користувача з введеним логіном і паролем надсилається запит до таблиці `users`; підключення до БД для перегляду цієї таблиці виконується під службовою роллю `connect_user`. Перед відправкою запиту відбувається шифрування введеного користувачем пароля. Якщо в таблиці `users` запис з даними логіном і паролем існує, то відбувається від'єднання від БД, а наступне за цим нове підключення до БД виконується від імені ролі, зазначеної в стовпці `category` для знайденого користувача. Користувачеві, який успішно пройшов процедуру автентифікації, надається інтерфейс, який забезпечує засоби для виконання задач відповідної категорії користувачів.

ІНСТРУКЦІЯ КОРИСТУВАЧА

Мета створення інструкції користувача полягає в тому, щоб забезпечити користувача необхідною інформацією для самостійної роботи з ІС. Послідовність викладу інформації в інструкції повинна збігатися з послідовністю дій користувача.

Для демонстрації інтерфейсу створеної ІС в текст інструкції повинні бути включені копії основних екранних форм (скріншоти) і опис способу і порядку використання цих форм. При цьому необхідно вказувати, які задачі користувач може виконати за допомогою конкретної форми, які дані необхідно ввести за допомогою форми, щоб виконати певну задачу, яким буде результат її виконання, які повідомлення можуть бути видані по ходу роботи. Ця інформація може бути наведена як в текстовому вигляді (опис рис. 10.1 і 10.2 в наведеному нижче прикладі), так і у вигляді підказок на формі (рис. 10.3).

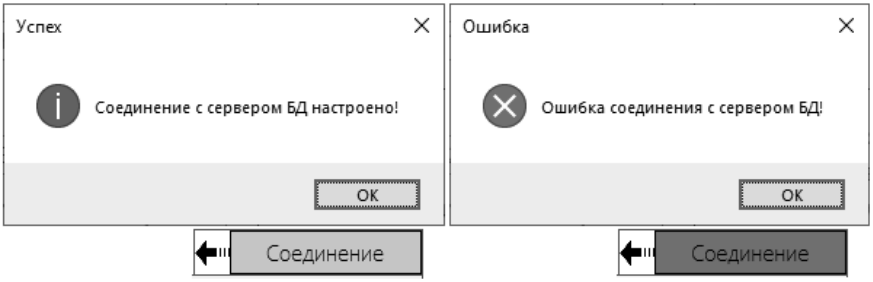
Нижче наведений приклад фрагменту розділу 10 пояснювальної записки, що містить інструкцію користувача для застосунка «Облік обладнання навчальних лабораторій кафедри».

При запуску програми користувач-адміністратор бачить вікно (рис. 10.1, а), що містить два поля для введення логіна і пароля, а також кнопку «Войти» для підтвердження введення даних. Користувач також може увійти як гість, натиснувши на кнопку «Войти в приложение-клиент», але при цьому йому буде доступний обмежений функціонал програми. Вгорі над полем «логін» знаходиться кнопка «Авторизация» для налаштування з'єднання з сервером БД. При її

натисканні відкривається форма, в якій містяться поля для введення IP-адреси сервера і номера порту (рис. 10.1, б). Після натискання кнопки «Проверить соединение» програма встановлює з'єднання з сервером БД (рис. 10.2, а), або видає помилку про неможливість підключення (рис. 10.2, б), при цьому також змінюється колір кнопки переходу.

а) б)

Рисунок 10.1 – Форма авторизації (а) і форма з'єднання з сервером БД (б)



а) б)

Рисунок 10.2 – Успішний (а) і неуспішний (б) результати з'єднання з сервером БД

Структура інтерфейсу застосунка адміністратора показані на рис. 10.3.

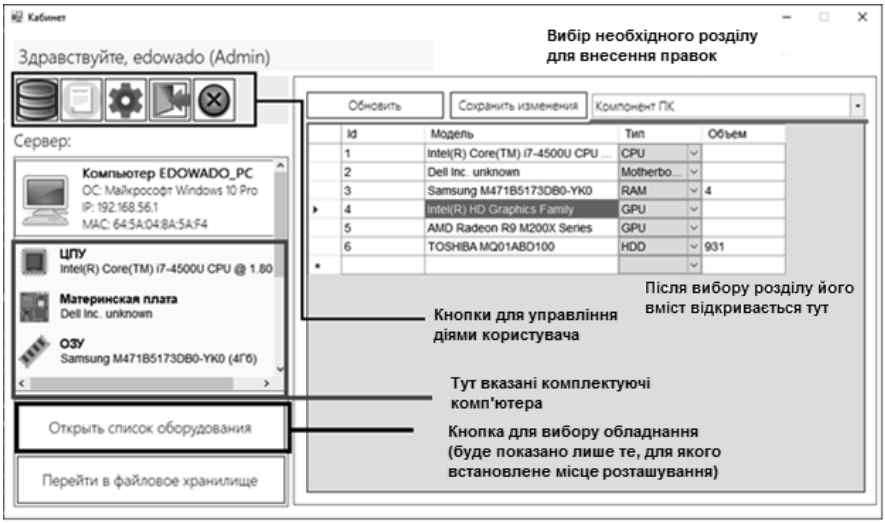


Рисунок 10.3 – Структура інтерфейсу застосунка адміністратора

На рис. 10.4 показане призначення кнопок користувацького інтерфейсу.

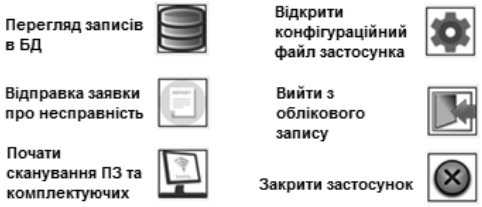


Рисунок 10.4 – Призначення кнопок користувацького інтерфейсу

ВИСНОВКИ

Даний структурний елемент пояснювальної записки розміщується після викладу основної частини, починаючи з нової сторінки, і його обсяг не повинен перевищувати двох сторінок.

Висновки повинні містити повні й чіткі висновки за результатами виконаної роботи, які повинні відповідати поставленим раніше задачам, можливі області використання результатів роботи, рекомендації щодо можливого практичного застосування роботи, основні напрямки подальшого її розвитку.

Приклад тексту висновків:

В результаті аналізу предметної області сформульована мета створення ІС проектування і реалізація інформаційної системи обліку обладнання навчальних лабораторій кафедри університету, визначений список категорій її користувачів (Адміністратор, Співробітник кафедри, Гість) і задач, які користувачі повинні розв'язувати за допомогою ІС. Сформовані вимоги до даних і спроектована база даних з 28 таблиць для зберігання і маніпулювання даними предметної області.

Для створення ІС обрані дворівнева архітектура клієнт-сервер, шаблон проектування MVP, СУБД PostgreSQL, мова програмування C# і WinForms для розробки користувацького інтерфейсу.

Інтерфейс клієнтської програми розроблений з урахуванням вимог кожної категорії користувачів і надає необхідний функціонал для розв'язання відповідних задач. Доступ до даних з боку різних категорій користувачів розмежований за допомогою механізму ролей і привілеїв. Захист від несанкціонованого доступу реалізований шляхом використання механізму аутентифікації і авторизації.

За рахунок використання в створеній інформаційній системі дворівневої архітектури клієнт-сервер досягнута стабільність і надійність системи, а завдяки використанню шаблону проектування MVP – її висока масштабованість.

Створена ІС є добре структурованою, в ній можливо як нарощування функціоналу користувачів вже наявних категорій, так і додавання функціоналу нових категорій користувачів, наприклад, співробітників центру інформаційних технологій, які виконують ремонт комп'ютерів і їхніх комплектуючих відповідно до заявок співробітників кафедри.

ПРАВИЛА ОФОРМЛЕННЯ ПОЯСНЮВАЛЬНОЇ ЗАПИСКИ ДО КУРСОВОГО ПРОЕКТУ

Правила оформлення сформульовані згідно державного стандарту [8].

При оформленні тексту пояснювальної записки для опису виконаних автором етапів проектування ІС **не дозволяється** використовувати вирази типу «я створив», «ми обрали», «розглядалося», «було описано» тощо. Натомість слід використовувати безособові дієслова доконаного виду, яким не повинно передувати дієслово «було». Наприклад: «створено», «обрано», «розглянуто», «описано».

Орієнтовний **обсяг пояснювальної записки** повинен складати 30-40 сторінок формату А4 (210х297 мм). Сторінки повинні бути заповнені хоча б на 50%. **Розміри полів сторінок**: верхнє і нижнє – 20 мм, праве – 15 мм, ліве – 25 мм. Основний текст повинен бути надрукований без переносів **шрифтом** Times New Roman Cyr розміром 14 пунктів, **вирівнювання** – по ширині сторінки, **міжрядковий інтервал** – 1,5. **Фрагменти вихідного коду програм** наводяться шрифтом Courier New розміром 12 пунктів, вирівнювання – по лівому краю, міжрядковий інтервал – 1-1,15. **Абзацний відступ** повинен бути однаковим впродовж усього тексту і приблизно дорівнювати п'яти знакам (12,5 мм).

Помилки, описки і графічні неточності (у друкованому документі) можуть бути виправлені зафарбовуванням коректором і нанесенням на те ж

місце (або між рядками) виправленого тексту чорного кольору від руки. На сторінці повинно бути не більше двох таких виправлень.

Титульний лист оформлюється українською мовою, інші структурні елементи записки – українською або англійською мовою (за бажанням автора).

Сторінки слід нумерувати арабськими цифрами, дотримуючись наскрізної нумерації впродовж усього тексту. Номер проставляють у верхній частині сторінки справа без крапки в кінці, починаючи з листа, наступного за першою сторінкою змісту. У загальну нумерацію сторінок включають всі листи, починаючи з титульного.

Розділи і підрозділи, пункти і підпункти основної частини повинні мати **заголовки**. Заголовки СТРУКТУРНИХ ЕЛЕМЕНТІВ та РОЗДІЛІВ слід розташовувати в середині рядка нової сторінки, друкувати великими літерами **напівжирним шрифтом** без крапки в кінці, не підкреслюючи. Заголовки підрозділів, пунктів і підпунктів слід починати з абзацного відступу і друкувати **напівжирним шрифтом** малими літерами, крім першої великої, не підкреслюючи, без крапки в кінці. Відстань між заголовком і подальшим чи попереднім текстом має дорівнювати одному рядку. Не допускається розміщувати назву підрозділу в нижній частині сторінки, якщо після нього розташовано менше двох рядків тексту.

Розділи і підрозділи слід нумерувати арабськими цифрами. Номер підрозділу складається з номера розділу і порядкового номера підрозділу, розділених крапкою. Після номера крапку не ставлять. При посиланнях на розділи, підрозділи, пункти, підпункти вказують їхні номери, наприклад: ... в розділі 4 ..., ... див. підрозділ 2.1.

Структурні елементи «ЗМІСТ», «ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ І ТЕРМІНІВ», «ВСТУП», «ВИСНОВКИ», «СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ», «ДОДАТОК» не нумерують.

Додатки нумеруються прописними буквами алфавіту, які розташовуються після слова «ДОДАТОК». При нумерації додатків не припустимо використання наступних букв відповідних алфавітів: в українській мові: Ґ, Є, З, І, Ї, Й, О, Ч, Ь; в англійській мові: I, O.

Ілюстрації (схеми, діаграми, знімки екранних форм) слід розташовувати безпосередньо після тексту, в якому вони згадуються вперше, або на наступній сторінці. На всі ілюстрації мають бути посилання в тексті.

Ілюстрації розміщуються з вирівнюванням по центру і повинні мати назву, яку розміщують під ілюстрацією через один рядок після неї також з вирівнюванням по центру. Відстань між текстом і ілюстрацією становить один рядок.

Ілюстрації слід нумерувати арабськими цифрами порядковою нумерацією в межах розділу, за винятком ілюстрацій, наведених у додатках. Номер

ілюстрації складається з номера розділу і порядкового номера ілюстрації, розділених крапкою. Номер ілюстрації від її назви відділяється тире, крапка в кінці назви не ставиться. Наприклад, другий малюнок третього розділу слід зазначити наступним чином:

Рисунок 3.2 – Структурна схема алгоритму методу

Якщо ілюстрація не вміщується на одній сторінці, можна переносити її на інші сторінки, при цьому назву ілюстрації розміщують на першій сторінці і на кожній сторінці під рисунком вказують: «Рисунок _, лист _».

Посилання на ілюстрації в тексті скорочують і пишуть з малої літери, наприклад, «див. рис. 3.2» або, при неявному посиланні, «(рис. 3.2)».

Таблиці оформляються відповідно до схеми:

Таблиця номер – Назва таблиці

Заголовки рядків	Заголовок графи таблиці		Заголовок графи таблиці	
	Підзаголовок графи	Підзаголовок графи	Підзаголовок графи	Підзаголовок графи

Головка

Боковик

Таблицю слід розташовувати безпосередньо після тексту, в якому вона згадується вперше, або на наступній сторінці. Таблиця має назву, яку друкують малими літерами (крім першої великої) і вміщують над таблицею з вирівнюванням вліво. Нумеруються таблиці арабськими цифрами порядковою нумерацією в межах розділу, за винятком таблиць, що наводяться в додатках. Номер таблиці складається з номера розділу і порядкового номера таблиці, між якими ставиться крапка.

Заголовки граф таблиці пишуть з великої літери, а підзаголовки – з малої, якщо вони складають одне речення з заголовком. Підзаголовки, що мають самостійне значення, пишуть з великої літери. В кінці заголовків і підзаголовків таблиць крапки не ставлять. Заголовки і підзаголовки граф вказують в однині.

На всі таблиці повинні бути посилання в тексті. Посилання на таблиці в тексті скорочують і пишуть з малої літери, наприклад, «див. табл. 4.2» або, при неявному посиланні, «(табл. 4.2)».

Якщо рядки або графи таблиці виходять за формат сторінки, таблицю поділяють на частини, розміщуючи одну частину під іншою або поруч, переносячи частину таблиці на наступну сторінку. При цьому в кожній частині таблиці повторюють її головку і боковик. При поділі таблиці на частини

допускається її головку або боковик заміняти відповідно номерами граф чи рядків. При цьому нумерують арабськими цифрами графи та/або рядки першої частини таблиці. Слово «Таблиця _» вказують один раз зліва над першою частиною таблиці, над іншими частинами пишуть: «Продовження таблиці _» із зазначенням її номера.

Тексти програм (лістинги) слід розташовувати безпосередньо після тексту, в якому вони згадуються вперше, або на наступній сторінці. Відстань між текстом і лістингом становить один рядок. Кожен лістинг повинен мати назву, яку розміщують під лістингом через один рядок після нього з вирівнюванням по центру.

Лістинги слід нумерувати арабськими цифрами порядковою нумерацією в межах розділу. Номер лістинга складається з номера розділу і порядкового номера лістинга, розділених крапкою. Номер лістинга від його назви відділяється тире, крапка в кінці назви не ставиться. Наприклад, перший лістинг п'ятого розділу слід позначити наступним чином:

Лістинг 5.1 – SQL-запит на виведення даних

Якщо лістинг не вміщується на одній сторінці, можна переносити його на інші сторінки, при цьому назву лістинга розміщують на першій сторінці і на кожній сторінці під лістингом вказують: «Лістинг _, лист _».

На всі лістинги повинні бути дані посилання в тексті, наприклад, «див. лістинг 4.2» або, при неявному посиланні, «(лістинг 4.2)».

Перед **переліком** ставлять двокрапку. Перед кожною позицією переліку слід ставити малу літеру або цифру з дужкою або, не нумеруючи, коротке тире (перший рівень деталізації). Для подальшої деталізації переліку слід використовувати літери з дужкою (другий рівень деталізації). Переліки першого рівня деталізації друкують малими літерами з абзацного відступу, другого рівня – з відступом щодо місця розташування переліків першого рівня. наприклад:

1) інтегроване середовище для розробки програмного забезпечення Microsoft Visual Studio;

2) засоби баз даних:

а) система управління базами даних з відкритим вихідним кодом PostgreSQL;

б) інструмент для корпоративного моделювання PowerDesigner;

– ...;

– ...;

3) компілятори.

У всіх розділах роботи необхідно вказувати посилання на літературні джерела, з яких запозичуються матеріали. Не допускається переказ тексту інших авторів без посилань на них, а також пряме цитування без використання лапок.

Відповідальність за компіляцію і достовірність відомостей, що містяться в кваліфікаційної роботі, несе її виконавець.

Категорично не допускається використання оригінальних текстів (зокрема, з Інтернет), рисунків, таблиць і т.п. інших авторів за винятком випадків, коли необхідне цитування. У разі цитування кожна цитата повинна мати обов'язкове посилання на першоджерело.

Робота з обсягом прямого цитування (включаючи не зазначені як цитати оригінальні тексти інших авторів) більше 10% від всієї роботи може бути прирівняна до реферативної. Оцінка за таку роботу автоматично знижується незалежно від отриманих автором результатів або взагалі не допускається до захисту.

Список використаних джерел приводять в кінці пояснювальної записки, починаючи з нової сторінки. Він включає описи лише тих джерел, які були використані при виконанні роботи і на які є посилання в тексті. Джерела нумеруються і друкуються з абзацного відступу. Відомості про джерела слід наводити і розташовувати в порядку появи посилань на них у тексті. Неприпустимо вказувати джерела, на які в тексті немає жодного посилання. У тексті записки посилання на використані літературні джерела слід позначати порядковим номером за переліком посилань, виділеним двома квадратними дужками, наприклад: «... в роботі [2] ...» або «... в роботах [1-7,10] ...».

Вимоги до оформлення бібліографічних описів сформульовані у відповідності до державного стандарту [9].

Приклади оформлення бібліографічних посилань:

1. Таненбаум Э., Архитектура компьютера. 4-е изд. – СПб.: Питер, 2003. – 697 с.: ил.
2. Самофалов К.Г., Цифровые ЭВМ: Теория и проектирование: Учебник. / К.Г. Самофалов, В.Н. Корнейчук, В.И. Тарасенко / Под ред. К.Г.Самофалова. – К.: Вища шк., 1989. – 424 с.
3. Bacon DF, Compiler transformations for high performance computing / DF Bacon, SL Grahem, OJ Sharp // ASM Computing Surveys. 1994. V. 26. № 4. – P. 217-225.
4. MPI: A Message-Passing Interface Standard. June 12, 1995 [Електронний ресурс] – Режим доступу: http://www.mcs.anl.gov/mpi.
5. MPI - The Complete Reference: Vol.1, The MPI Core. Second edition / Snir M., Otto S., Huss-Lederman S. etc. – published by MIT Press, 1998. – 426 p.

Додаткові матеріали (вихідні тексти програм, графічні матеріали, таблиці великого формату і т.п.) слід розміщувати в **додатках**, які оформлюються на наступних після списку літератури сторінках. Кожен додаток слід починати з нової сторінки. У першому рядку з вирівнюванням по центру друкують слово «ДОДАТОК» і букву, яка вказує його номер. З наступного рядка друкують заголовок додатка малими літерами з першої великої, наприклад:

ДОДАТОК А
Сценарій створення бази даних

Якщо в додаток винесені документи, на яких неможливо надрукувати заголовок (наприклад, зовнішні документи формату А4), то заголовок такого додатка оформляється в центрі окремого листа, що передує такому додатку. Цей лист також входить в нумерацію сторінок і на ньому ставиться номер. Номери сторінок на самих документах такого додатка можуть не ставиться, але в нумерації вони враховуються.

При необхідності текст додатків може поділятися на розділи, підрозділи, пункти, які слід нумерувати в межах кожного додатка. При цьому ставлять номер додатка – букву і цифру через крапку, наприклад, А.2 – другий розділ додатка А. Ілюстрації, таблиці, формули і лістинги, які містяться в тексті додатка, слід нумерувати в межах кожного додатка, аналогічно нумерації в розділах основної частини, наприклад, Рисунок Г.3 – третій рисунок додатка Г.

Посилання на додатки в тексті пишуть з малої літери, наприклад, див. додаток А чи, при неявному посиланні, (додаток А).

КРИТЕРІЇ ОЦІНЮВАННЯ ЗАХИСТУ КУРСОВОГО ПРОЕКТУ

№	Критерій	Бали
1	Наявність працездатної системи відповідно до завдання (в тому числі архітектура системи, повнота розв’язуваних задач)	до 30
2	Коректність і оптимальність SQL-запитів на маніпулювання даними, наявність збережених процедур (функцій) та тригерів	до 20
3	«Дружність» користувацького інтерфейсу	до 10
4	Пояснювальна записка	до 10
5	Захист курсового проекту (презентація та відповіді на запитання по темі роботи)	до 30
	Разом	до 100

РЕКОМЕНДОВАНА ЛІТЕРАТУРА

Основна література

1. Малахов Є.В., Проектування баз даних та їх реалізація засобами стандартного SQL та PostgreSQL: Навч. посіб. для студ. вищих навч. закладів / Є.В. Малахов, О.А. Блажко, М.Г. Глава // Одеса: ВМВ, 2012. – 248 с.
2. Г. Гайна, Основи проектування баз даних. Навчальний посібник / Вид. «Кондор», 2018. – 204 с.
3. T. Connolly, C. Begg, Database Systems: A Practical Approach to Design, Implementation, and Management, 6th Edition / Pearson, 2014. – 1440 p.
4. T C.J. Date, An Introduction to Database Systems, 8th Edition / Pearson, 2003. – 1040 p.

Допоміжна література

5. H.-J. Schönig, Mastering PostgreSQL 13: Build, administer, and maintain database applications efficiently with PostgreSQL 13, 4th Edition / Packt Publishing, 2020 – 476 p.
6. R. Elmasri, S. B. Navathe, Fundamentals of Database Systems, 7th edition / Pearson, 2015. – 1280 p.
7. R.O. Obe, L.S. Hsu, PostgreSQL: Up and Running: A Practical Guide to the Advanced Open Source Database, 3rd edition / O'Reilly Media, 2017. – 314 p.
8. ДСТУ 3008:2015. Звіти у сфері науки і техніки: Структура та правила оформлювання.
9. ДСТУ ГОСТ 7.1:2006. Бібліографічний запис, бібліографічний опис. Загальні вимоги та правила складання.

ІНФОРМАЦІЙНІ РЕСУРСИ

1. Робочі матеріали за курсом та електронний варіант конспекту лекцій на порталі «Публікації Одеського національного університету імені І.І. Мечникова» – Режим доступу: http://pub.onu.edu.ua/fakultet-matematyky-fizyky-ta-informatsiinykh-tekhnolohii/63-osvitnii-riven-bakalavr/spetsialnist-126-informatsiini-systemy-ta-tekhnolohii.
2. PostgreSQL: The world's most advanced open source database – Режим доступу: http://www.postgresql.org/
3. PostgreSQL | Найбільш просунута відкрита СУБД в світі – Режим доступу: http://postgresql.ru.net/
4. Васильєв А.Ю. Робота з PostgreSQL: настройка і масштабування – Режим доступу: http://postgresql.leopard.in.ua/

ДОДАТОК А
Титульний лист

МІНІСТЕРСТВО ОСВІТИ И НАУКИ УКРАЇНИ
ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені І.І. МЕЧНИКОВА
Факультет математики, фізики та інформаційних технологій
Кафедра математичного забезпечення комп'ютерних систем

КУРСОВИЙ ПРОЕКТ

з дисципліни

«Проектування інформаційних систем»

на тему:

_____

_____

студента (студентки) III курсу
групи _____
спеціальності _____
_____
(Прізвище, ім'я та по батькові)
Керівник: _____
(Прізвище, ім'я та по батькові)
Захищено «___» _____ 202__ р.
з оцінкою _____
комісія:

_____
(ПІБ) (Підпис)
_____
(ПІБ) (Підпис)
_____
(ПІБ) (Підпис)

Одеса – 202__

ДОДАТОК Б
Сценарій створення таблиць бази даних

```
CREATE TABLE Room (
    id smallserial PRIMARY KEY,
    number varchar (100) UNIQUE NOT NULL);

CREATE TABLE Computer (
    computer_name varchar (100) PRIMARY KEY,
    serial_number varchar (100) UNIQUE NOT NULL,
    audit_id smallint REFERENCES Room (id) NOT NULL,
    ip inet NOT NULL,
    mac macaddr UNIQUE NOT NULL,
    status boolean DEFAULT true NOT NULL,
    UNIQUE (audit_id, ip));

CREATE TYPE matrix_m AS ENUM ('TFT TN', 'TFT VA', 'TFT IPS', 'OLED');

CREATE TABLE Monitor (
    id serial PRIMARY KEY,
    model varchar (100) UNIQUE NOT NULL,
    matrix matrix_m NOT NULL,
    diagonal double precision CHECK (diagonal > 0) NOT NULL,
    resolution varchar (30) NOT NULL,
    status boolean DEFAULT true NOT NULL);

CREATE TABLE Monitor_And_Computer (
    serial_number varchar (100) PRIMARY KEY,
    computer_name varchar (100) REFERENCES Computer (computer_name)
         NOT NULL,
    monitor_id integer REFERENCES Monitor (id) NOT NULL,
    status boolean DEFAULT true NOT NULL,
    price integer CHECK (price >= 0) NOT NULL,
UNIQUE (computer_name, monitor_id));

CREATE DOMAIN type_cp VARCHAR(50)
DEFAULT 'CPU'
CONSTRAINT type_check
CHECK (VALUE IN ('CPU', 'GPU', 'Motherboard', 'RAM', 'Soundcard',
    'HDD'));

CREATE TABLE Component_Parts (
    id serial PRIMARY KEY,
    model varchar (100) UNIQUE NOT NULL,
    type type_cp NOT NULL,
    capacity smallint NULL);

CREATE TABLE Component_In_Computer (
    computer_name varchar (100) NOT NULL REFERENCES Computer
         (computer_name),
    component_id int NOT NULL REFERENCES Component_Parts (id),
```

```
    count smallint CHECK (count >= 0) DEFAULT 1 NOT NULL,
    locked boolean DEFAULT false NOT NULL,
    PRIMARY KEY (computer_name, component_id));

CREATE DOMAIN type_os VARCHAR(50)
DEFAULT 'Windows'
CONSTRAINT type_check
CHECK (VALUE IN ('Windows', 'Linux'));

CREATE TABLE OS (
    id smallserial PRIMARY KEY,
    name varchar (100) NOT NULL,
    serial_os varchar (100) NOT NULL,
    type type_os NOT NULL);

CREATE TABLE Software (
    id serial PRIMARY KEY,
    name varchar (300) UNIQUE NOT NULL,
    cost integer CHECK (cost >= 0) DEFAULT 0 NOT NULL);

CREATE TABLE Request_For_Software_Installation (
    id serial PRIMARY KEY,
    fio varchar(120) NOT NULL,
    commentary Text NOT NULL,
    computer_name varchar (100) REFERENCES Computer (computer_name)
         NOT NULL,
    os_id smallint REFERENCES OS (id) NOT NULL,
    program_id integer REFERENCES Software (id) NOT NULL,
    date date NOT NULL);

CREATE TABLE Cathedra (
    id smallserial PRIMARY KEY,
    name varchar (100) NOT NULL);

CREATE TABLE Teacher (
    id smallserial PRIMARY KEY,
    first_name varchar (100) NOT NULL,
    second_name varchar (100) NOT NULL,
    patronymic varchar (100) NOT NULL,
    cathedra_id smallint REFERENCES Cathedra (id) NOT NULL);

CREATE TABLE Course (
    id smallserial PRIMARY KEY,
    course_name varchar (100) NOT NULL);

CREATE TABLE Course_And_Software (
    id smallserial PRIMARY KEY,
    audit_id smallint REFERENCES Room (id) NOT NULL,
    course_id smallint REFERENCES Course (id) NOT NULL,
    program_id integer REFERENCES Software (id) NOT NULL,
    teacher_id smallint REFERENCES Teacher (id) NOT NULL,
    UNIQUE (course_id, program_id, teacher_id, audit_id));
```

```
CREATE DOMAIN type_netd VARCHAR(50)
DEFAULT 'Router'
CONSTRAINT type_check
CHECK (VALUE IN ('Router', 'Switch', 'Hub', 'Repeater'));

CREATE TABLE Network_Device (
    id serial PRIMARY KEY,
    model varchar (100) NOT NULL,
    total_ports smallint NOT NULL,
    type type_netd NOT NULL);

CREATE TABLE NetDevice_In_Room (
    id serial PRIMARY KEY,
    netdevice_id integer REFERENCES Network_Device (id) NOT NULL,
    audit_id smallint REFERENCES Room (id) NOT NULL,
    busy_ports smallint CHECK (busy_ports >= 0) NOT NULL,
    free_ports smallint CHECK (free_ports >= 0) NOT NULL,
    ip inet NOT NULL UNIQUE,
    mac macaddr NOT NULL UNIQUE,
    status boolean NOT NULL,
    price integer CHECK (price >= 0) NOT NULL,
    UNIQUE (netdevice_id, audit_id));

CREATE DOMAIN type_prcs VARCHAR(50)
DEFAULT 'Printer'
CONSTRAINT type_check
CHECK (VALUE IN ('Printer', 'Scanner'));

CREATE TABLE Printer_Scanner (
    id serial PRIMARY KEY,
    model varchar (100) NOT NULL,
    type type_prcs NOT NULL);

CREATE TABLE RrSc_er_In_Room (
    id serial PRIMARY KEY,
    prsc_er_id integer REFERENCES Printer_Scanner (id) NOT NULL,
    audit_id smallint REFERENCES Room (id) NOT NULL,
    status boolean NOT NULL,
    price integer CHECK (price >= 0) NOT NULL,
    UNIQUE (prsc_er_id, audit_id));

CREATE TABLE Projector (
    id serial PRIMARY KEY,
    model varchar (100) NOT NULL);

CREATE TABLE Projector_In_Room (
    id serial PRIMARY KEY,
    projector_id integer REFERENCES Projector (id) NOT NULL,
    audit_id smallint REFERENCES Room (id) NOT NULL,
    status boolean NOT NULL,
    price integer CHECK (price >= 0) NOT NULL,
    UNIQUE (projector_id, audit_id));
```

```
CREATE TABLE Appliances (
    id serial PRIMARY KEY,
    model varchar (100) NOT NULL);

CREATE TABLE Appliances_In_Room (
    id serial PRIMARY KEY,
    appliances_id integer REFERENCES Appliances (id) NOT NULL,
    audit_id smallint REFERENCES Room (id) NOT NULL,
    status boolean,
    price integer CHECK (price >= 0) NOT NULL,
    UNIQUE (appliances_id, audit_id));

CREATE DOMAIN type_keymse VARCHAR(50)
DEFAULT 'Keyboard'
CONSTRAINT type_check
CHECK (VALUE IN ('Keyboard', 'Mouse'));

CREATE TABLE Keyboard_Mouse (
    id serial PRIMARY KEY,
    model varchar (100) NOT NULL,
    type type_keymse NOT NULL);

CREATE TABLE KeyMse_In_Room (
    id serial PRIMARY KEY,
    keymse_id integer REFERENCES Keyboard_Mouse (id) NOT NULL,
    audit_id smallint REFERENCES Room (id) NOT NULL,
    status boolean NOT NULL,
    price integer CHECK (price >= 0) NOT NULL,
    UNIQUE (keymse_id, audit_id));

CREATE DOMAIN type_cab VARCHAR(50)
DEFAULT 'Schuko'
CONSTRAINT type_check
CHECK (VALUE IN ('Schuko', 'VGA', 'HDMI', 'DVI', 'USB'));

CREATE TABLE Cable (
    id serial PRIMARY KEY,
    type varchar(50) NOT NULL);

CREATE TABLE Cable_In_Room (
    id serial PRIMARY KEY,
    cable_id integer REFERENCES Cable (id) NOT NULL,
    audit_id smallint REFERENCES Room (id) NOT NULL,
    length smallint CHECK (length >= 0) NOT NULL,
    status boolean NOT NULL,
    price integer CHECK (price >= 0) NOT NULL,
    UNIQUE (cable_id, audit_id));

CREATE TABLE Request_For_Repair (
    id serial PRIMARY KEY,
    date date NOT NULL,
    fio varchar(120) NOT NULL,
```

```
    commentary Text NOT NULL,
    computer_name varchar (100) REFERENCES Computer (computer_name),
    component_id integer REFERENCES Component_In_Computer (id),
    monitor_id integer REFERENCES Monitor (id),
    netdevice_id integer REFERENCES Network_Device (id),
    keymse_id integer REFERENCES KeyMse_In_Room (id),
    prsc_er_id integer REFERENCES RrSc_er_In_Room (id),
    projector_id integer REFERENCES Projector_In_Room (id),
    appliances_id integer REFERENCES Appliances_In_Room (id),
    cable_id integer REFERENCES Cable_In_Room (id));

CREATE TABLE OS_and_Software (
    computer_name varchar (50) REFERENCES Computer (computer_name),
    program_id integer REFERENCES Software (id),
    os_id smallint REFERENCES OS (id),
    path_exe varchar NOT NULL,
    locked boolean DEFAULT false NOT NULL,
    serial_key varchar(100) NOT NULL,
    PRIMARY KEY (computer_name, program_id, os_id));

CREATE TABLE users (
login varchar (50) not null PRIMARY KEY,
password varchar (255) NOT NULL,
category varchar (50) NOT NULL
    CHECK (category in 'administrator', 'employee', 'guest'));
```

ДОДАТОК В
Запити на створення тригерів і функцій

Допоміжний тригер для шифрування пароля при додаванні або зміні користувача: визначення тригерної функції міститься у лістингу В.1, а визначення тригера – у лістингу В.2.

```
CREATE OR REPLACE FUNCTION hasg_password()
RETURNS trigger AS
$ BODY $
BEGIN
-- якщо операція INSERT або якщо новий пароль відрізняється від старого
 IF (TG_OP = 'INSERT' OR NEW.password NOT LIKE OLD.password) THEN
-- тоді новий пароль хешується
 NEW.password = encode (digest (NEW.password, 'sha256'), 'hex');
 END IF;
-- повернути новий запис
 RETURN NEW;
END;
$BODY$ language plpgsql;
```

Лістинг В.1 – Код тригерної функції для шифрування пароля при додаванні або зміні користувача

```
CREATE TRIGGER make_password_hash
-- спрацьовує перед додаванням або оновленням даних в таблиці users
 BEFORE INSERT OR UPDATE ON users
-- для кожного доданого запису буде виконуватися тригерна функція
-- hasg_password
 FOR EACH ROW
 EXECUTE PROCEDURE hasg_password();
```

Лістинг В.2 – Визначення тригера для шифрування пароля при додаванні або зміні користувача

Функція для додавання інформації про ПЗ, встановлене на конкретному комп'ютері (лістинг В.3). Інформацію про ПЗ, якого немає в БД, потрібно додати в відповідну таблицю. Ця функція необхідна, щоб при скануванні комп'ютера автоматично додавати або оновлювати дані про ПЗ. Вхідні параметри: ім'я комп'ютера, масив з назв програм, назва ОС, масив із шляхів до цих програм (ця інформація формується спеціальним ПЗ при скануванні комп'ютера і передається в дану функцію).

```
CREATE OR REPLACE FUNCTION AddSoftwareToPc (comp_name varchar (100), soft
varchar [], os_name varchar (100), soft_path varchar [])
returns void as $$
DECLARE
-- локальні змінні функції
os_id smallint;
BEGIN
-- вибирається ідентифікатор ОС, назва якої збігається з тою,
-- що передає комп'ютер (комп'ютер передає назву ОС у вигляді
-- «Windows 10 Pro» і не знає її ідентифікатора в таблиці БД).
-- Можна вказувати як імена переданих параметрів, так і їхні
-- порядкові номери у вигляді конструкції $номер.
SELECT id INTO osid FROM os WHERE name ILIKE $3;
-- якщо така ОС не знайдена, то необхідно викликати виключення,
-- видавши повідомлення про помилку
IF (osid IS NULL)
 THEN RAISE Exception 'Wrong OS NAME!';
END IF;
-- З таблиці OS_and_Software видаляються всі незаблоковані програми,
-- встановлені на заданій ОС заданого комп'ютера, яких немає в новому
-- списку. Оскільки функції передаються не ідентифікатори програм, а їхні
-- назви (як у випадку з ОС), то використовується підзапит. Тим самим
-- ідентифікатор програми в таблиці os_and_software порівнюється зі
-- списком ідентифікаторів програм, отриманих по їхній назві
DELETE FROM OS_and_Software WHERE computer_name = $1 AND os_id = osid AND
locked = false AND program_id IN (SELECT id FROM software WHERE name = ANY
($2));
-- В таблицю software додаються ті програми, які там відсутні.
-- Така ситуація може виникнути, наприклад, коли хтось встановив
-- нову версію Visual Studio, а в БД її ще немає; значить, інформацію
-- про нову версію потрібно додати
INSERT INTO software (name) SELECT * FROM unnest ($2) AS names WHERE NOT
EXISTS (SELECT * FROM software WHERE name = ANY ($2));
-- І нарешті все передані в якості параметра програми пов'язуються
-- з переданим в якості параметра комп'ютером
INSERT INTO OS_and_Software (computer_name, program_id, os_id, path_exe)
SELECT $1, s.id, osid, unnest ($4) FROM software s WHERE s.name = ANY ($2)
ON CONFLICT DO NOTHING;
END;
$$ language plpgsql;
```

Лістинг В.3 – Код функції для додавання інформації про ПЗ, встановлене на конкретному комп'ютері

Параметр `ON CONFLICT` задає дію, що замінює виникнення помилки при порушенні обмеження унікальності або обмеження-виключення. В даному випадку значення `DO NOTHING` скасовує додавання рядка.

Навчальне видання

ПРОЕКТУВАННЯ ІНФОРМАЦІЙНИХ СИСТЕМ

Методичні вказівки
до курсового проектування
для студентів факультету математики, фізики та
інформаційних технологій
першого (бакалаврського) рівня освіти,
спеціальності 126 «Інформаційні системи та технології»

Укладачі
Малахов Євгеній Валерійович
Розновець Ольга Ігорівна

В авторській редакції

Підписано до друку 30.10.2023 р. Формат 60x84/16.
Папір офсетний. Гарнітура Times. Цифровий друк.
Ум. друк. арк. 3,25. Наклад 30. Зам. № 1123-0904.
Віддруковано з готового оригінал-макета.

Видавництво та друк: ОЛДІ+
65101, Україна, м. Одеса, вул. Інглезі, 6/1
Свідоцтво ДК № 7642 від 29.07.2022 р.

Тел.: +38 (098) 559-45-45,
+38 (095) 559-45-45, +38 (093) 559-45-45
Для листування: 65101, Україна, м. Одеса, вул. Інглезі, 6/1
E-mail: office@oldiplus.ua