

Міністерство освіти і науки України
Фаховий коледж ракетно-космічного машинобудування
Дніпровського національного університету імені Олеся Гончара

Предметна комісія програмної інженерії

ДИПЛОМНИЙ ПРОЄКТ

Тема: Експертна система формування інтегрованих навчальних планів в системі безперервної підготовки фахівців.

Виконав студент Вайчекаускас Станіслав Костянтинович
(прізвище, ім'я та по батькові)

Відділення комп'ютерної і програмної інженерії КПІ

Спеціальність 121 «Інженерія програмного забезпечення»
(код і назва спеціальності)

Курс IV № групи (шифр) ПЗ-16-1

Керівник дипломного проєкту _____

(прізвище, ім'я та по батькові, посада, учений ступінь, учене звання)

Консультанти (з окремих розділів проєкту):

з економічної частини: Л.В. Болваненко
(назва розділу, оцінка, підпис, ініціали, прізвище)

з охорони праці: Д.С. Пустовий
(назва розділу, оцінка, підпис, ініціали, прізвище)

Старший консультант В.О. Михайлова
(підпис, ініціали, прізвище)

Голова предметної комісії програмної інженерії С.С. Ланська
(назва комісії, підпис, ініціали, прізвище)

Дата захисту: “_____” червня 2020 р.

Оцінка _____

Підписи членів ЕК: Голова ЕК О.П. Луценко

Заст. голови ЕК В.В. Сітарчук

Член комісії В.О. Михайлова

Секретар С.С. Ланська

м. Дніпро

2020 р.

РЕФЕРАТ

Темою дипломного проекту є «Експертна система. Формування інтегрованих навчальних планів в системі безперервної підготовки фахівців».

Дипломний проект: 78 сторінок, 26 рисунків, 18 таблиць, 10 джерел, додаток.

Об'єкт дослідження: експертні системи.

Мета роботи: — дослідити існуючі експертні системи та розробити програму для формування інтегрованих навчальних планів на основі введеної інформації.

Постановка задачі 10 сторінок. Інформація про вимоги до використовуваних технічних засобів, опис інструментальних засобів, що використовувались при розробці програми.

Опис етапів реалізації 9 сторінок. Докладна інформація про етапи проектування та розробки програмного додатку.

Опис програмного продукту 9 сторінок. Докладна інструкція користувача для роботи з програмою.

Економічна частина 4 сторінки. Економічний розрахунок вартості програмного продукту.

В данному проекті засобами QtCreator Community 5.12.4 розроблено експертну систему, що призначена для формування інтегрованих навчальних планів здобувачів освіти в системі «неперервної освіти» та переліку питань для організації вступного фахового випробування.

Ключові слова: *освіта, експертна система, QtCreator.*

ЗМІСТ

ЗМІСТ.....	3
ВСТУП.....	4
1 ПОСТАНОВКА ЗАДАЧІ.....	6
1.1 Технічне завдання на розробку програмного продукту.....	6
1.2 Огляд існуючих рішень.....	9
1.3 Обґрунтування середовища розробки та вибору мови програмування....	12
2 ОПИС ЕТАПІВ РЕАЛІЗАЦІЇ.....	16
2.1 Опис проектування бази.....	16
2.2 Опис проектування інтерфейсу програми.....	21
3 ОПИС ПРОГРАМНОГО ПРОДУКТУ	25
4 АНАЛІЗ ДОСЛІДНОЇ ІНФОРМАЦІЇ.....	34
5 ОХОРОНА ПРАЦІ.....	38
1.1 Аналіз небезпечних та шкідливих виробничих чинників проектного технологічного процесу, об'єкту, система або пристрою.....	39
1.2 Інженерно-технічні заходи з охорони праці.....	40
1.3 Пожежна профілактика.....	42
1.4 Заходи з ергономіки.....	43
6 ЕКОНОМІЧНИЙ РОЗДІЛ.....	45
ВИСНОВКИ.....	50
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	51
Додаток.....	52

					ДП.ПЗ.161.03.ПЗ			
Змн.	Лист	№ докум.	Підпис	Дата				
Розроб.		Вайчекаускас С.К.			Програмна оболонка створення бази знань для обліку зарахованих дисциплін в системі безперервної освіти	Літ.	Арк.	Архів
Перевір.		Ланська С.С.					2	73
Реценз.		Білодородько О.І.				ФКРКМ ДНУ		
Н. контр.		Михайлова В.О.						
Затверд.								

ВСТУП

В умовах сучасного інформаційного суспільства все актуальнішою стає потреба використання новітніх технологій у навчанні, як для покращення методів та технології навчання, так і для підвищення його ефективності. Також використання новітніх інформаційних технологій дає можливість набагато збільшити стандартизацію знань та методи покращення сприймання їх студентами навчального курсу, та значно зменшити часові та фінансові затрати у різних галузях.

Одним з шляхів розв'язання цієї проблеми є система безперервної освіти, яка складається з базової і подальшої освіти та передбачає на другому етапі послідовне чергування навчання в системі спеціально створених освітніх закладів з професійною діяльністю.

Завдяки своїй змістовій наповненості й необмеженості в часі безперервна професійна освіта має можливості для виконання важливих функцій, а саме:

- загальноосвітньої, компенсуючої (усунення недоліків у базовій освіті, її доповнення новою інформацією, що з'являється в умовах інформаційно-технологічної революції);
- адаптивної (гнучка професійна підготовка, перепідготовка й підвищення кваліфікації з метою оновлення професійного досвіду, здобуття іншого фаху в умовах постійних змін на виробництві, розвитку теле- та радіокомунікацій, комп'ютерного доступу до інформаційних банків даних тощо);
- економічної (задоволення потреб держави, регіонів, різних галузей промисловості, сільського господарства і сфери послуг у конкурентоспроможних фахівцях, підготовлених до впровадження новітніх технологій, техніки та ін.).

Сьогодні процес складання інтегрованих навчальних планів, заснований на досвіді і інтуїції працівників вищої школи потребує серйозного удосконалення та наукового підґрунтя прийнятих рішень. Процес конструювання індивідуального навчального плану студента або групи студентів може являти собою пе-

		Вайчекаускас С.К.			ДП.ПЗ.161.03.ПЗ	Арк.
		Ланська С.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		43

дагогічну технологію, орієнтовану на реалізацію інформаційних технологій за допомогою експертної системи. В умовах скорочених строків навчання застосування експертних систем забезпечує можливість побудови індивідуального навчального плану, що підвищує ефективність процесів навчання, викладання і самоосвіти.

Програмна оболонка для створення бази знань— це складова експертної системи яка забезпечує формування індивідуальних навчальних планів здобувачів в системі «неперервної освіти» та перелік питань для організації вступного фахового випробування.

Використання експертної системи при складанні інтегрованих індивідуальних навчальних планів має ряд переваг. По-перше, з часом система буде розширюватися і накопичувати відповідності компетенцій як з суміжних спеціальностей, так і з інших галузей знань. По-друге, можна складати робочі навчальні плани напрямів підготовки без повторного залучення експертів з різних предметних областей. По-третє, систему легко можна буде перенавчати відповідно до нових вимог складання інтегрованих планів.

Виходячи з вище написаного, можна зробити висновок, що тема дипломного проекту є актуальною, а поставлене завдання — своєчасним.

		Вайчекаускас С.К.			ДП.ПЗ.161.03.ПЗ	Арк.
		Ланська С.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		43

1 ПОСТАНОВКА ЗАДАЧІ

1.1 Технічне завдання на розробку програмного продукту

1.1.1 Найменування розробки

Темою дипломного проекту є «Експертна система. Формування інтегрованих навчальних планів в системі безперервної підготовки фахівців».

1.1.2 Підстава для розробки

Підставою для даного проекту слугує завдання для дипломного проекту, яке видане Коледжем ракетно-космічного машинобудування Дніпровського національного університету ім. О. Гончара.

1.1.3 Призначення розробки

Програма призначена для обробки наявної інформації про освітні програми вищих навчальних закладів, складання на їх основі тестових завдань для вступу і формування інтегрованих навчальних планів для вступників.

1.1.4 Вимоги до програмного продукту, що розробляється

В межах даного дипломного проекту потрібно організувати базу даних для збереження інформації про дисципліни, модулі та компетенції освітніх рівнів, між якими формується інтегрований план, а також коефіцієнт відповідності між рівнями. Також необхідно передбачити збереження питань, які в подальшому можуть бути використані для формування тестів вступного випробування.

		Вайчекаускас С.К.			ДП.ПЗ.161.03.ПЗ	Арк.
		Ланська С.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		43

Програмну необхідно спроектувати на базі клієнт-серверної технології з виконанням наступних функцій:

- розмежуванням доступу до даних користувачів;
- додавання, редагування та видалення дисциплін;
- додавання, редагування та видалення модулів;
- додавання, редагування та видалення компетенцій;
- додавання, редагування та видалення коефіцієнтів відповідності;
- додавання, редагування та видалення питань;
- можливість створення, завантаження та редагування різних проектів;
- додавання та видалення користувачів певного проекту;
- формування звітності про зараховані модулі та предмети;
- формування питань для вступних екзаменів.

Програмна оболонка мати контроль введення даних на коректність та повторюваність.

Для функціонування програмного додатку необхідно визначити три рівні доступу до системи: Адміністратор, Оператор та Експерт. Дані групи необхідні для розмежування доступу до засобів системи з метою підвищення її безпеки.

Адміністратор є найбільш привілейованою і нечисленною групою користувачів. Користувачам даної групи дозволено створювати, видаляти і редагувати всі, без винятку групи користувачів, робити налаштування системи, робити резервні копії та відновлення бази даних.

Оператор є другою за можливостями доступу групою користувачів. Учасникам цієї групи дозволено створювати і редагувати дані по дисциплінам, модулям, компетентностям та тестовим запитанням, також саме оператор формує звітність по перезаліченим предметам та створеним питанням до вступних іспитів.

Експерти — це група користувачів-фахівців в тій галузі знань, завдання якої повинна вирішувати експертна система, тобто вносити, редагувати та видаляти коефіцієнти відповідності між компетенціями різних освітніх рівнів.

		Вайчекаускас С.К.			ДП.ПЗ.161.03.ПЗ	Арк.
		Ланська С.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		43

1.1.5 Вимоги до програмного та апаратного забезпечення на етапі експлуатації

Рекомендовані вимоги до програмного та апаратного забезпечення серверу:

- підтримка СУБД на базі MySQL;
- операційні системи Windows 7, 8, 10, Linux;
- оперативної пам'яті не менше 1ГБ;
- процесор з частотою не менше 1ГГц;
- вільний дисковий простір не менше 1ГБ.

Рекомендовані вимоги до програмного та апаратного забезпечення клієнта:

- підтримка СУБД MySQL;
- операційні системи Windows 7, 8, 10, Linux;
- процесор з частотою не менше 1ГГц;
- вільної оперативної пам'яті не менше 100МБ;
- вільного дискового простору не менше 100МБ;
- підтримка і наявність інструментів Qt.

1.1.6 Вимоги до програмної документації

Основними документами, що регламентують розробку майбутніх програм, повинні бути документи Єдиної системи програмної документації (ЄСПД): постановка завдання, опис етапів реалізації, опис застосування.

		Вайчекаускас С.К.			ДП.ПЗ.161.03.ПЗ	Арк.
		Ланська С.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		43

1.1.7 Календарний план робіт

Таблиця 1.1 – Календарний план розробки проекту

п/п	Назва етапів розробки проекту	Строк виконання етапів проекту
1	Аналіз предметної області	05.04.2020
2	Аналіз існуючих подібних розробок	15.04.2020
3	Вибір технічних засобів для реалізації завдання	30.04.2020
4	Виділення основного функціоналу експертної системи	02.05.2020
5	Проектування бази даних	04.05.2020
6	Проектування основних компонентів	07.05.2020
7	Проектування інтерфейсу	10.05.2020
8	Проектування інструментів	14.05.2020
9	Реалізація інтерфейсу	16.05.2020
10	Реалізація інструментів	18.05.2020
11	Тестування та усунення помилок	22.05.2020

1.2 Огляд існуючих рішень

У зв'язку з розробкою дипломного проекту було прийнято рішення аналізувати існуючі програмні експертні системи схожого призначення для співставлення перспектив проекту та для кращого розуміння того, які можливості та функціонал повинен бути втілений у дипломній роботі.

Найцікавішою програмою стала MYCIN. MYCIN — це експертна система розроблена на початку 1970-х років в Стенфордському університеті. Вона була спроектована для діагностування бактерій, що викликають важкі інфекції, такі як бактеріємія і менінгіт, а також для рекомендації необхідної кількості антибіотиків в залежності від маси тіла пацієнта.

MYCIN працювала за допомогою дуже простої машини виведення, та бази знань з ~ 600 правил. Після запуску, програма ставила користувачеві (лікаря) довгий ряд простих «так / ні» або текстових питань. В результаті, система нада-

вала список підозрюваних бактерій, відсортований за ймовірністю, вказувала довірчий інтервал для вірогідності діагнозів і їх обґрунтування (тобто MYCIN надавала список питань і правил, які привели її до саме такого ранжирування діагнозів), а також рекомендувала курс лікування.

Исследования, проведенные в Stanford Medical School, обнаружили, что MYCIN предлагает приемлемую терапию примерно в 69 % случаев, что лучше, чем у экспертов по инфекционным болезням, которых оценивали по тем же критериям.

1.3 Обґрунтування середовища розробки та вибору мови програмування

У якості мови програмування було обрано мову C++.

Мова програмування C++ найбільш поширена серед розробників програмного забезпечення. Вона є дуже зручною у розробці прикладних програм; драйверів пристроїв; розробка ОС; відео ігор. Реалізацією мови C++ займаються одночасно декілька проектів як безкоштовних, так і комерційних, а саме: GNU, Microsoft і Embarcadero (Borland).

Середовищем розробки було обрано Qt Creator.

Qt Creator — кросплатформенний інструментарій розробки програмного забезпечення (ПЗ) мовою програмування C++. Дозволяє запускати написане за його допомогою ПЗ на більшості сучасних операційних систем (ОС), просто компілюючи текст програми для кожної операційної системи без зміни серцевого коду. Містить всі основні класи, які можуть бути потрібні для розробки прикладного програмного забезпечення, починаючи з елементів графічного інтерфейсу й закінчуючи класами для роботи з мережею, базами даних, OpenGL, SVG і XML. Бібліотека дозволяє керувати потоками, працювати з мережею та забезпечує кросплатформенний доступ до файлів.

Qt Creator має вбудований редактор форм, що дає змогу власноруч прописувати стиль форми можливостями мови CSS. Підтримує CMake, що дає змогу

		Вайчекаукас С.К.			ДП.ПЗ.161.03.ПЗ	Арк.
		Ланська С.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		43

писати кросплатформенні додатки. З відомих додатків Qt використовує Google Earth, завдяки зручності використання цієї IDE .

Була обрана за можливість розробляти кросплатформенні проекти, легке перенесення проекту з однієї ОС у іншу, можливість написання мобільних додатків мовою C++, широкий вибір інструментів для розробки програмного забезпечення та зручний інтерфейс.

У якості СУБД обрано MySQL.

MySQL — це система управління базами даних.

База даних являє собою структуровану сукупність даних. Ці дані можуть бути будь-якими — від простого списку майбутніх покупок до переліку експонатів картинної галереї або величезної кількості інформації в корпоративній мережі. Для запису, вибірки й обробки даних, що зберігаються в комп'ютерній базі даних, необхідна система управління базою даних, якою і є ПО MySQL. Оскільки комп'ютери чудово справляються з обробкою великих обсягів даних, управління базами даних відіграє центральну роль в обчисленнях. Реалізовано таке управління може бути по-різному - як у вигляді окремих утиліт, так і у вигляді коду, що входить до складу інших додатків.

MySQL — це система управління реляційними базами даних.

В реляційній базі даних дані зберігаються не всі скопом, а в окремих таблицях, завдяки чому досягається вигреш в швидкості і гнучкості. Таблиці зв'язуються між собою за допомогою відносин, завдяки чому забезпечується можливість об'єднувати при виконанні запиту дані з декількох таблиць. SQL як частина системи MySQL можна охарактеризувати як мову структурованих запитів й одну з найвикористовуваних мов для доступу до баз даних.

Програмне забезпечення MySQL — це ПЗ з відкритим кодом.

ПЗ з відкритим кодом означає, що застосовувати і модифікувати його може будь-хто. Таке ПЗ можна отримувати по Internet і використовувати безкоштовно. При цьому кожен користувач може вивчити вихідний код і змінити його відповідно до своїх потреб. Використання програмного забезпечення MySQL

		Вайчекаукас С.К.			ДП.ПЗ.161.03.ПЗ	Арк.
		Ланська С.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		43

регламентується ліцензією GPL (GNU General Public License), в якій зазначено, що можна і чого не можна робити з цим програмним забезпеченням в різних ситуаціях. Якщо робота у рамках GPL вас не влаштовує або планується вбудовування MySQL-коду в комерційний додаток, є можливість купити комерційну ліцензовану версію у компанії MySQL AB. See section 1.6.3 ліцензії на ПЗ MySQL.

MySQL є дуже швидким, надійним і легким у використанні. MySQL має також ряд зручних можливостей, розроблених в тісному контакті з користувачами. Спочатку сервер MySQL розроблявся для управління великими базами даних з метою забезпечити більш високу швидкість роботи в порівнянні з існуючими на той момент аналогами. І ось вже протягом декількох років даний сервер успішно використовується в умовах промислової експлуатації з високими вимогами. Незважаючи на те що MySQL постійно вдосконалюється, він вже сьогодні забезпечує широкий спектр корисних функцій. Завдяки своїй доступності, швидкості і безпеки MySQL дуже добре підходить для доступу до баз даних по Internet.[11]

		Вайчекаускас С.К.			ДП.ПЗ.161.03.ПЗ	Арк.
		Ланська С.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		43

Технічні можливості СУБД MySQL.

MySQL є системою клієнт-сервер, яка містить багатопоточний SQL-сервер, що забезпечує підтримку різних обчислювальних машин баз даних, а також кілька різних клієнтських програм і бібліотек, засоби адміністрування і широкий спектр програмних інтерфейсів (API). Також поставляється у вигляді багатопоточної бібліотеки, яку можна підключити до користувача додатком і отримати компактний, швидкий і легкий в управлінні продукт.

		Вайчекаускас С.К.			ДП.ПЗ.161.03.ПЗ	Арк.
		Ланська С.С.				43
Змн.	Арк.	№ докум.	Підпис	Дата		

2 ОПИС ЕТАПІВ РЕАЛІЗАЦІЇ

2.1 Опис проектування бази

Програма має можливість обробляти дані різних проектів.

Кожен проект є окремою базою даних створеною за одним шаблоном. ER-діаграму шаблону можна побачити на рисунку 2.1

Також, для збереження інформації по кожному проекту і користувачеві, таку як: додаткова інформація проекту, його назва, логіни, ініціали та проекти до яких отримав доступ користувач, створено головну базу даних, ER-діаграму якої зображено на рисунку 2.2.

Шаблон бази даних проекту складається з 8-ми пов'язаних між собою таблиць-моделей, які умовно поділені на два рівня, перелік основних таблиць: Discipline — «Дисципліна», Module — «Модуль», Mod_comp— проміжна таблиця, Competence — «Компетенція», Conformity — «Відповідності», Question — «Питання для тесу», Type_compet «Типи компетенцій». Перелік таблиць, їх поля та призначення полів наведено у таблицях 2.1 – 2.7.

Таблиця 2.1 – Таблиця Discipline — «Дисципліна»

Поле	Призначення
id_discipline	Ідентифікатор дисципліни
name_discipline	Назва дисципліни
control_discipline	Вид контролю

Таблиця 2.2 – Таблиця Module— «Модуль»

Поле	Призначення
id_module	Ідентифікатор модуля
name_module	Назва модуля
hours_module	Кількість годин для модуля
id_discipline	Ідентифікатор дисципліни, до якої відноситься модуль

Таблиця 2.3 – Таблиця Mod_comp — проміжна таблиця

Поле	Призначення
id_module	Ідентифікатор модуля
id_compet	Ідентифікатор компетенції

Таблиця 2.4 – Таблиця Competence — «Компетенція»

Поле	Призначення
id_compet	Ідентифікатор компетенції
name_compe	Назва компетенції
kind_compet	Значення компетенції

Таблиця 2.5 – Таблица Conformity — «Відповідності»

Поле	Призначення
id_coeff	Ідентифікатор коефіцієнту відповідності
id_compet	Ідентифікатор компетенції першого рівня
id_compet_2	Ідентифікатор компетенції другого рівня
value_coeff	Значення відповідності

Таблиця 2.6 – Таблица Question — «Питання для тесту»

Поле	Призначення
id_question	Ідентифікатор питання
id_compet_2	Ідентифікатор компетенції другого рівня
question	Значення питання
answer_a	Варіант відповіді А
answer_b	Варіант відповіді Б
answer_c	Варіант відповіді В
answer_d	Варіант відповіді Г
answer_tru	Значення правильної відповіді

Таблиця 2.7 – Таблиця Type_compet — типи компетенцій

Поле	Призначення
id_type	Ідентифікатор типу
name_type	Найменування типу

Між таблицями встановлено зв'язок «один-до-багатьох». Також є реалізованими функції каскадного оновлення та видалення даних, та забезпечення цілісності даних.

Головна база даних складається з 3-х пов'язаних між собою таблиць: Databases — «Бази даних», Users — «Користувачі» та User_Database — проміжна таблиця між проектами та користувачами. Перелік таблиць, їх поля та призначення будуть зображені у таблицях 2.8 — 2.10.

Таблиця 2.8 – Таблиця Databases — список існуючих проектів

Поле	Призначення
name_db	Назва проекту
info_db	Додаткова інформація о проекті

Таблиця 2.9 – Таблиця Users — таблиця користувачів

Поле	Призначення
name	Логін користувача
role	Роль яку виконує у проектах
fio	Прізвище та ініціали

Таблиця 2.10 – Таблиця User_Databases — проміжна таблиця

Поле	Призначення
user_name	Логін користувача
database_name	Назва проекту

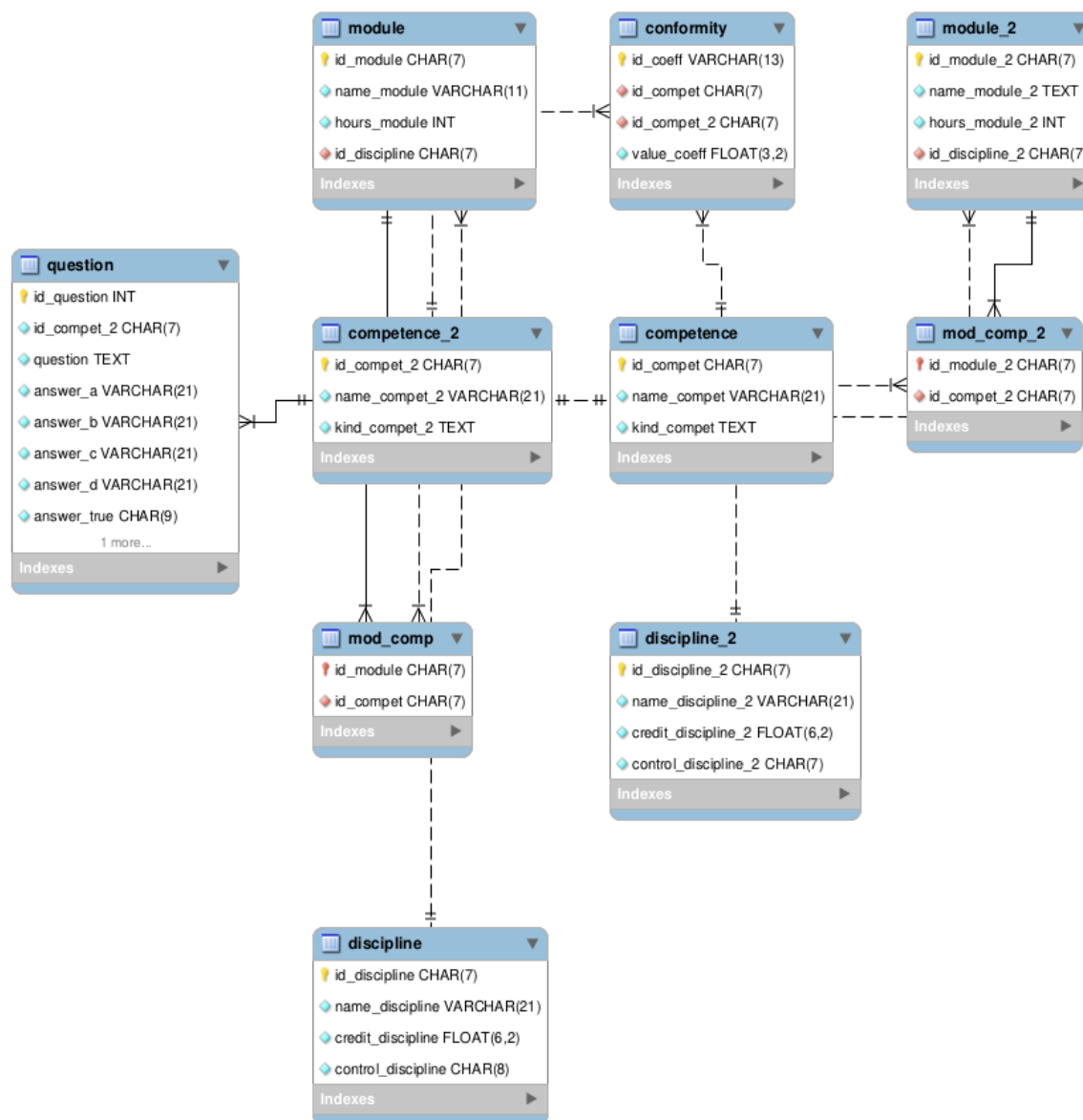


Рисунок 2.1 – ER-діаграма бази даних проекту

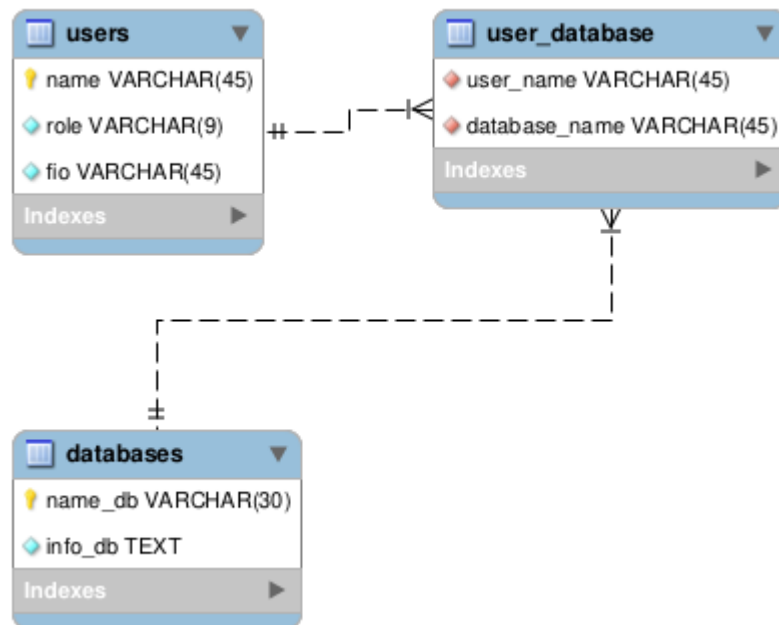


Рисунок 2.2 – ER-діаграма головної таблиці

2.2 Опис проектування інтерфейсу програми

Перед проектуванням інтерфейсу необхідно зрозуміти, який функціонал необхідний кожному типу користувачів. Для визначення ролей користувачів та розмежування їх функцій була розроблена модель сценаріїв, яка наведена на рисунку 2.3

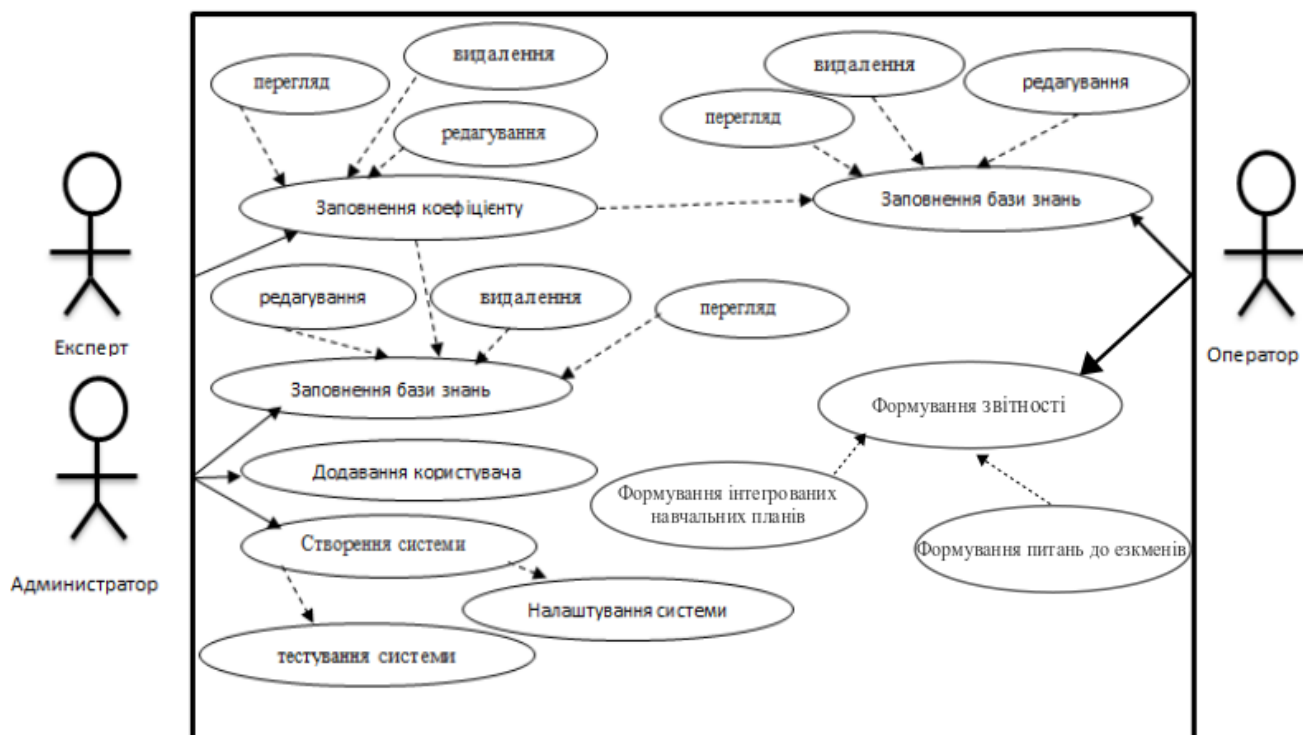


Рисунок 2.1 – Модель сценаріїв для програмної оболонки

Неформальний опис сценаріїв програмної системи.

Сценарій «Створення системи» передбачає:

- 1 Створення нового проекту.
- 2 Відновлення бази даних.

Сценарій «Заповнення бази знань» передбачає:

- 1 Додавання інформації про дисципліни, модулі, компетенції, питання для тестування.
- 2 Редагування інформації про дисципліни, модулі, компетенції, питання для тестування.
- 3 Перегляд інформації про дисципліни, модулі, компетенції, питання для тестування.
- 4 Видалення інформації про дисципліни, модулі, компетенції, питання для тестування.

Сценарій «Додавання користувача» передбачає:

- 1 Реєстрація нового експерта та закріплення його до бази.
- 2 Реєстрація нового оператора.

Сценарій «Заповнення коефіцієнту» передбачає:

- 1 Додавання значення коефіцієнту відповідності між компетенціями.
- 2 Редагування значення коефіцієнту.
- 3 Видалення інформації про значення коефіцієнту.

Сценарій «Формування звітності» передбачає:

- 1 Формування інтегрованих навчальних планів.
- 2 Формування питань до вступних іспитів.

Більш детально виділені функції кожної групи користувачів на діаграмі діяльності, яка представлена на рисунку 2.4.

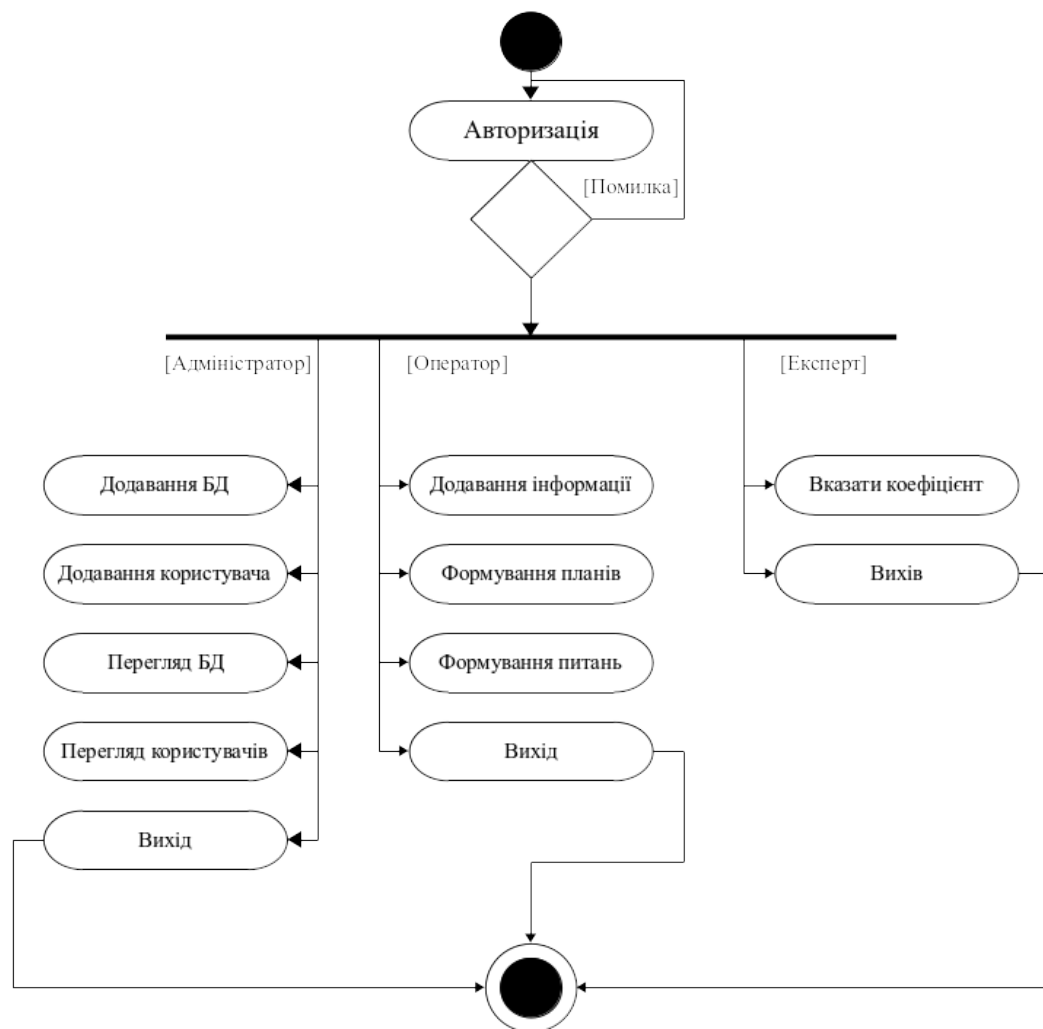


Рисунок 2.4 – Діаграма діяльності користувачів для програмної оболонки

Опис реалізації інтерфейсу програми входять наступні форми, призначення яких приведено в таблиці 2.11.

Таблиця 2.11 – Призначення форм

Назва	Призначення
Authorization	Форма авторизації користувачів
AdminMain	Робочий простір адміністратора
ExpertMain	Робочий простір експерта
OperMain	Робочий простір оператора
InformationDB_Dialog	Форма для перегляду і редагування БД
InformationUser_Dialog	Форма для перегляду і редагування користувача
DatabaseSelectionDialog	Форма для відбору необхідних БД
UsersSelectionDialog	Форма для відбору необхідних користувачів

3 ОПИС ПРОГРАМНОГО ПРОДУКТУ

При запуску програми, буде відкрито вікно авторизації, вікно зображено на рисунку 3.1. З його допомогою користувач може увійти у свій обліковий запис в системі. Також є можливість налаштувати параметри підключення, адресу та порт сервера, для цього треба натиснути на кнопку з трьома крапками у лівому кутку. Вікно параметрів можна побачити на рисунку 3.2.

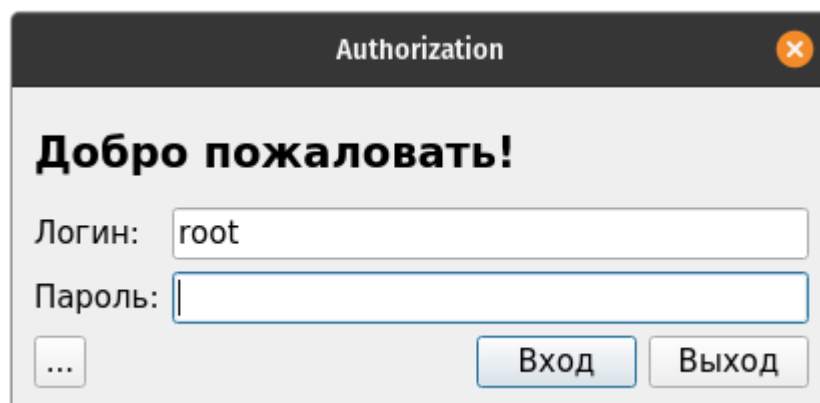


Рисунок 3.1 – Вікно авторизації

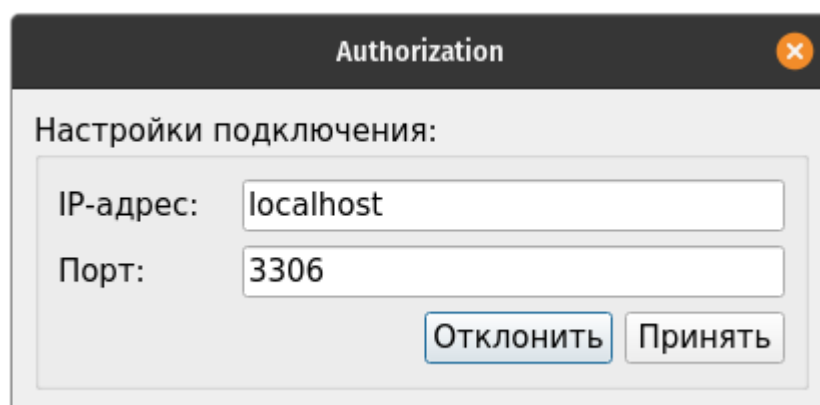


Рисунок 3.2 – Вікно параметрів у вікні авторизації

Після успішної авторизації буде відкрито вікно, відповідне ролі користувача, далі по черзі.

Для «Адміністратора» буде відкрито вікно, яке надає усі інструменти для додавання, редагування, перегляду та видалення баз даних і користувачів. Зображення можна побачити на рисунку 3.3.

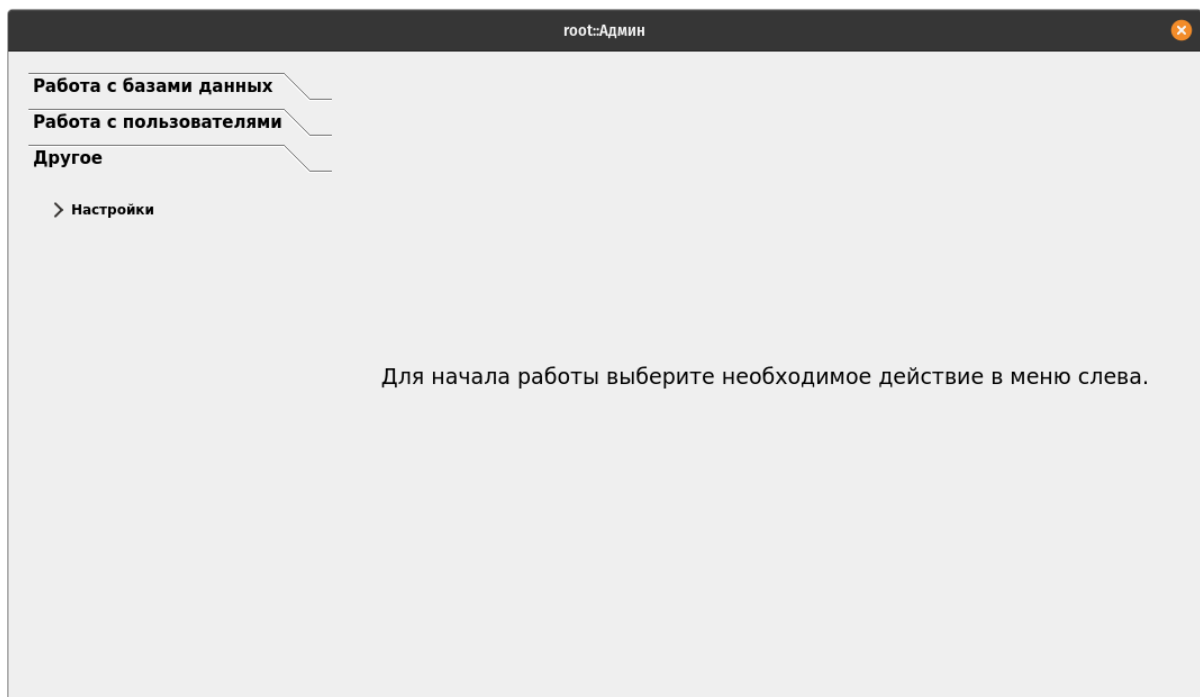


Рисунок 3.3 – Вікно авторизації

У меню ліворуч розташовано усі доступні для адміністратора дії, розбиті на розділи. При старті користувача зустрічає стартова сторінка із коротким повідомленням, про наступні дії.

Перший розділ – це «Робота з базами даних». Він включає у себе два пункти, додавання нової бази даних та перегляд існуючих. Обравши перший пункт, користувач попаде на відповідну сторінку. Її зображено на рисунку 3.4.

Рисунок 3.4 – Сторінка додавання бази даних

На сторінці знаходиться поле для вводу назви бази даних, додаткової інформації та розділ перегляду і вибору користувачів, які зможуть редагувати базу. Для переходу у меню вибору користувачів, необхідно натиснути на кнопку «Изменить». Після цього відкриється вікно з вибором користувачів, зображено на рисунку 3.5. Коли усі ключові поля будуть уведені, програма не виявить помилок, стане активної кнопка «Готово», яка у свою чергу, при натисканні, сповістить користувача про успішне створення бази даних, та запропонує перейти у головне меню або продовжити створення баз даних. Якщо буде знайдено помилки у введеній інформації — програма сповістить про це користувача.

Вікно являє собою два стовпчики, у першому знаходиться список користувачів, у другому всі обрані користувачі. Вікно надає можливості пошуку. Пошук можливий за введеною користувачем строкою, за роллю шуканого користувача або одразу всіма параметрами. Якщо всі поля пошуку пусті, програма виведе список усіх користувачів. Для додавання користувача у список редакторів цієї бази даних, необхідно натиснути на напис у стовпчику ліворуч, тоді він з'явиться у праворуч, з минулого буде видалений. Для збереження змін треба натиснути кнопку «Сохранить», для відхилення «Отменить», а для очищення стовбця обраних користувачів необхідно натиснути кнопку «Сбросить всё».

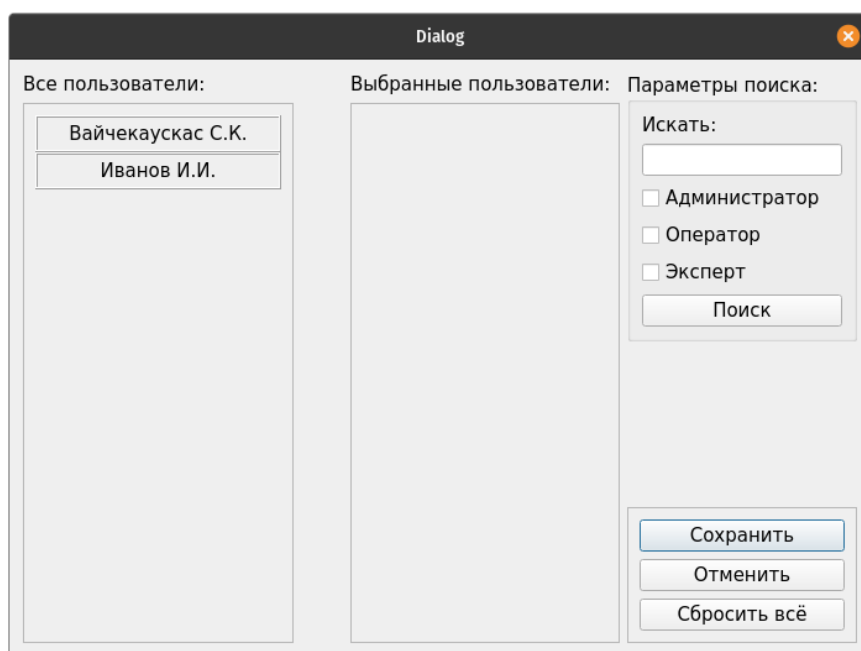


Рисунок 3.5 – Вікно вибору користувачів

Обравши другий пункт — користувач попаде на сторінку перегляду існуючих баз даних. Сторінку зображено на рисунку 3.6.

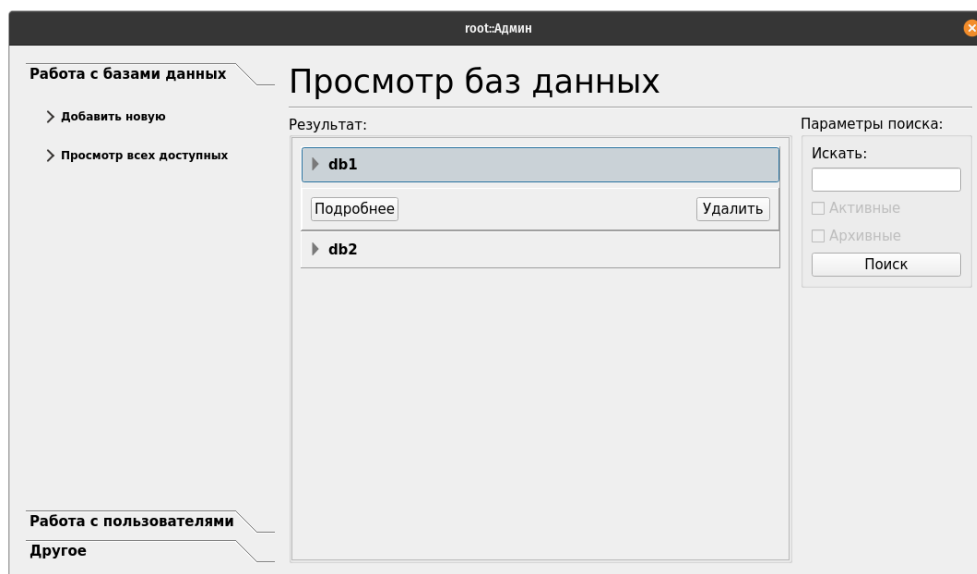


Рисунок 3.6 — Сторінка перегляду існуючих баз даних

На сторінці буде список існуючих баз даних, які представлені своєрідними кнопками. При натисканні на таку кнопку з неї буде випадати меню, з можливими діями. При повторному натисканні меню буде сховане. Також для користувача доступний пошук.

Меню дій дає можливість перейти до вікна перегляду інформації про обрану базу даних або видалити її. Для переходу до вікна перегляду інформації — необхідно натиснути на кнопку «Свойства». Вікно зображено на рисунку 3.7.

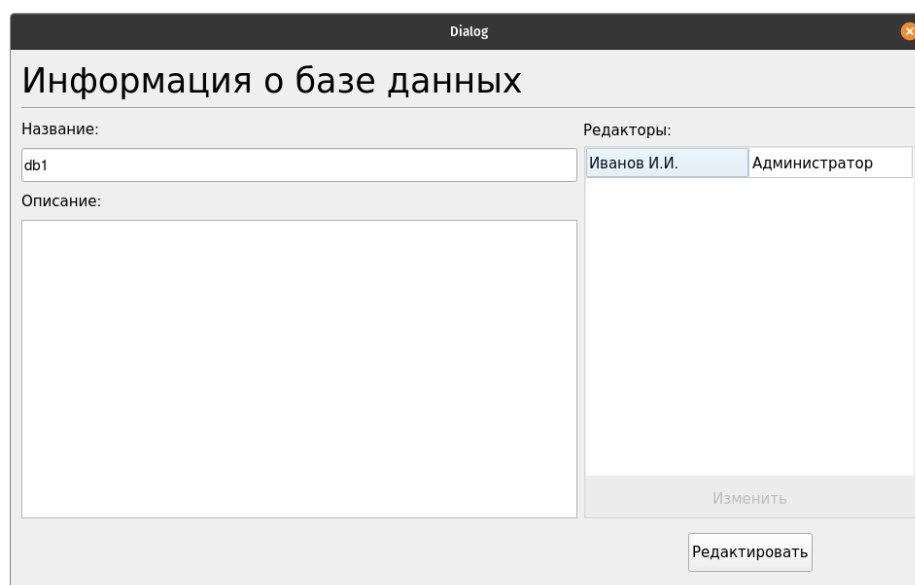


Рисунок 3.7 — Вікно перегляду інформації бази даних

Вікно показує усю необхідну для адміністратора інформацію про базу даних, а саме: назва, додаткова інформація та редактори. Для того щоб редагувати інформацію необхідно натиснути на кнопку «Редактировать», тоді поля для вводу стануть активними, і їх можна буде редагувати. Також з'явиться дві кнопки для збереження або відхилення змін.

Другий розділ — «Робота з користувачами». Також має два пункти, додавання та перегляд користувачів. Додавання зображено на рисунку 3.8.

Рисунок 3.8 — Сторінка додавання користувача

Принцип роботи сторінки додавання користувача відповідний сторінці додавання бази даних. При натисканні на кнопку «Изменить» відкриється вікно вибору баз даних, які будуть доступні для редагування цьому користувачеві. Вікно зображено на рисунку 3.9.

Рисунок 3.9 — Вікно вибору бази даних

Вікно вибору бази даних працює відповідно вікну вибору користувачів.

Перегляд користувачів такий самий як і перегляд баз даних, суттєвої різниці не має, лише розширений пошук. Зображено на рисунку 3.10. Випадаюче меню так само надає можливість видалити користувача чи переглянути інформацію про нього. Вікно перегляду інформації зображено на рисунку 3.11.

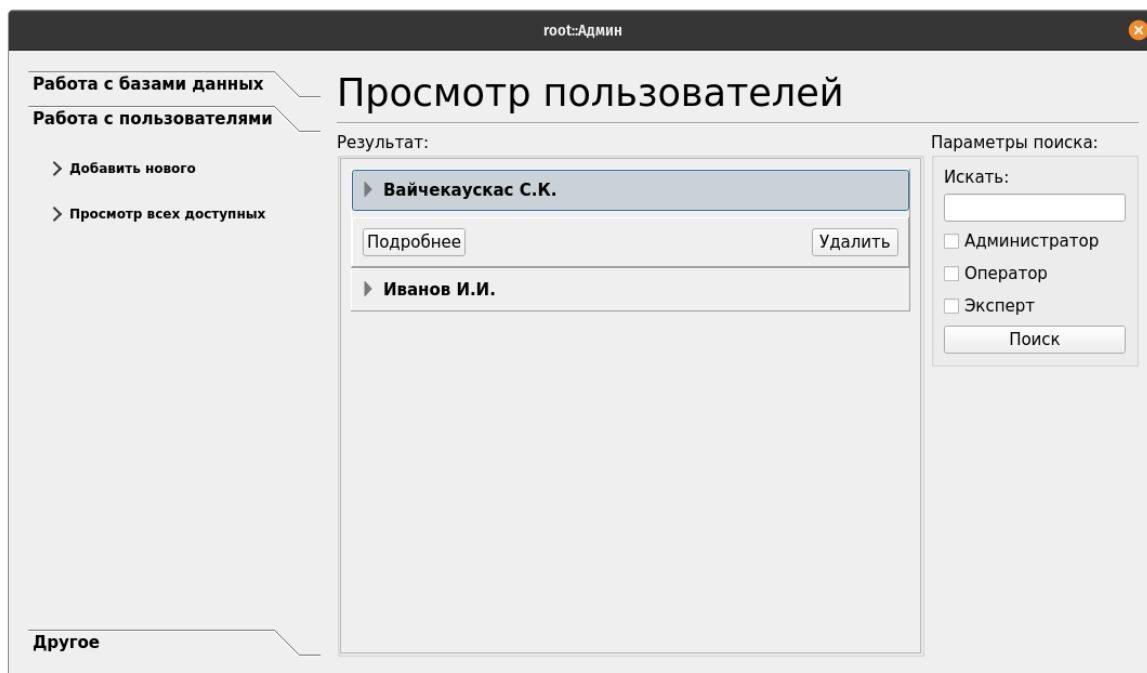


Рисунок 3.10 – Вікно перегляду користувачів

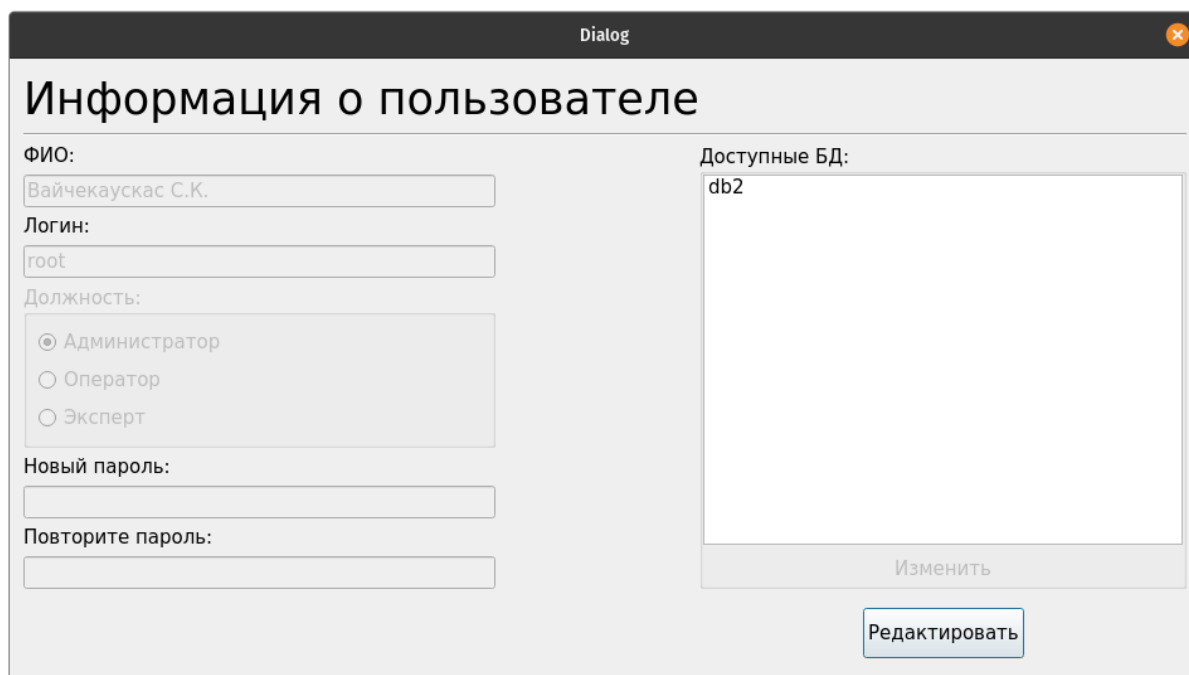


Рисунок 3.11 – Вікно перегляду інформації користувача

Третій розділ містить якісь додаткові функції для адміністратора. Передбачається, що там будуть розташовані тестуючі інструменти, інструменти для збору статистичної інформації, більш тонке налагодження системи. На даний момент у цьому пункті знаходяться лише налаштування середовища адміністратора.

Для «Експерта» буде відкрито вікно вибору бази даних. Вікно має поле вибору бази даних та додаткові інформаційні поля, наприклад повної назви бази даних та додаткової інформації про неї. Вікно зображено на рисунку 3.12.

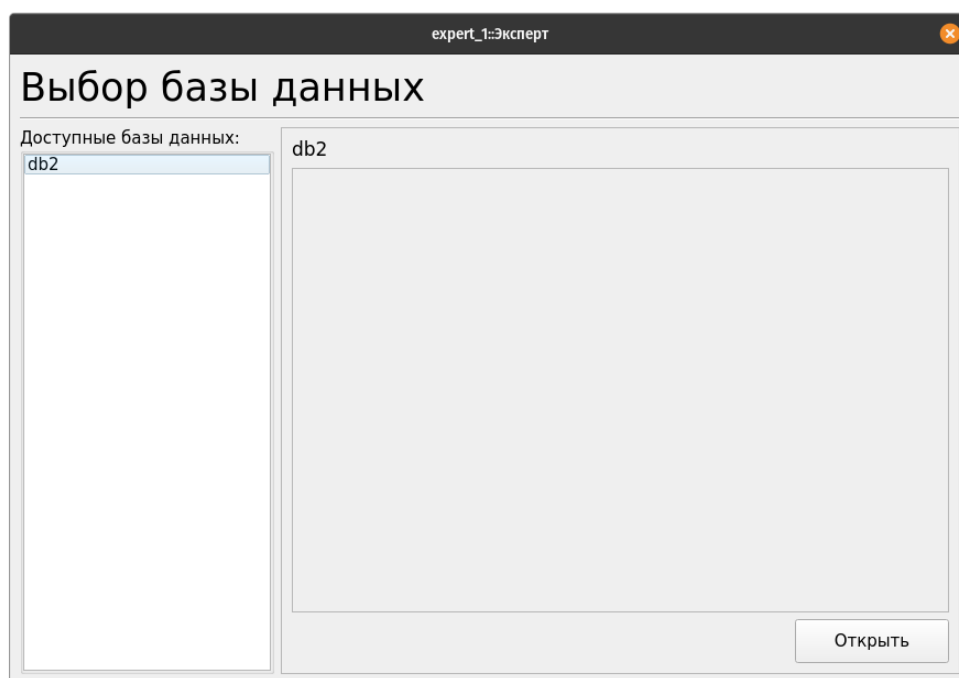


Рисунок 3.12 – Вікно вибору бази даних експертом

Для переходу у меню редагування коефіцієнтів, треба вибрати базу даних і натиснути кнопку «Открыть».

Вікно має дві колонки, ліворуч компетенції молодшого спеціаліста а праворуч спеціалісти, при виборі однієї із компетентностей у меню ліворуч — з'явиться опис цієї компетенції та відповідні їй компетенції у рядку праворуч. По середині знаходиться елемент для виставлення коефіцієнту, також присутні кнопки збереження та відхилення змін. Встановлювати коефіцієнт можна одразу декільком компетентностям, для цього треба обрати компетентність із

стовпичка ліворуч і декілька компетентностей із стовпичка праворуч. Вікно зображено на рисунку 3.13.

Рисунок 3.13 – Вікно редагування коефіцієнту

Для «Оператора» спочатку відкриється те ж саме вікно що і для експерта, де він повинен обрати базу даних для роботи.

Далі оператору буде відкрито вікно з основною інформацією про базу даних. За замовчуванням середовище відкривається у режимі перегляду, для переходу у режим редагування необхідно переключити кнопку у «інформаційному барі» в самому низу форми. Середовище оператора зображено на рисунку 3.14.

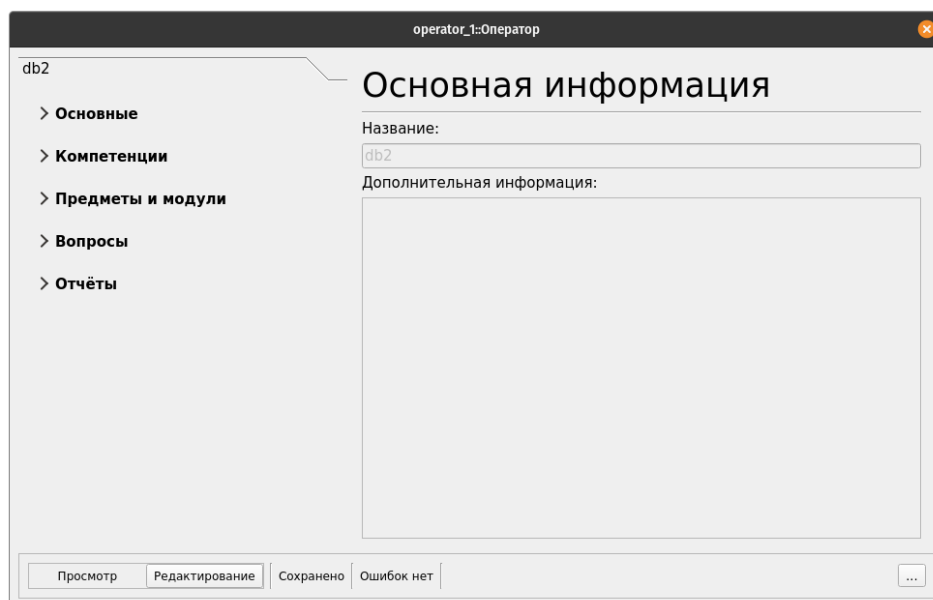


Рисунок 3.14 – Середовище роботи оператора
Ліворуч у порядку заповнення бази даних розташовані кнопки для переходу по пунктах перегляду або редагування бази даних. Меню редагування компетенцій зображено на рисунку 3.15.

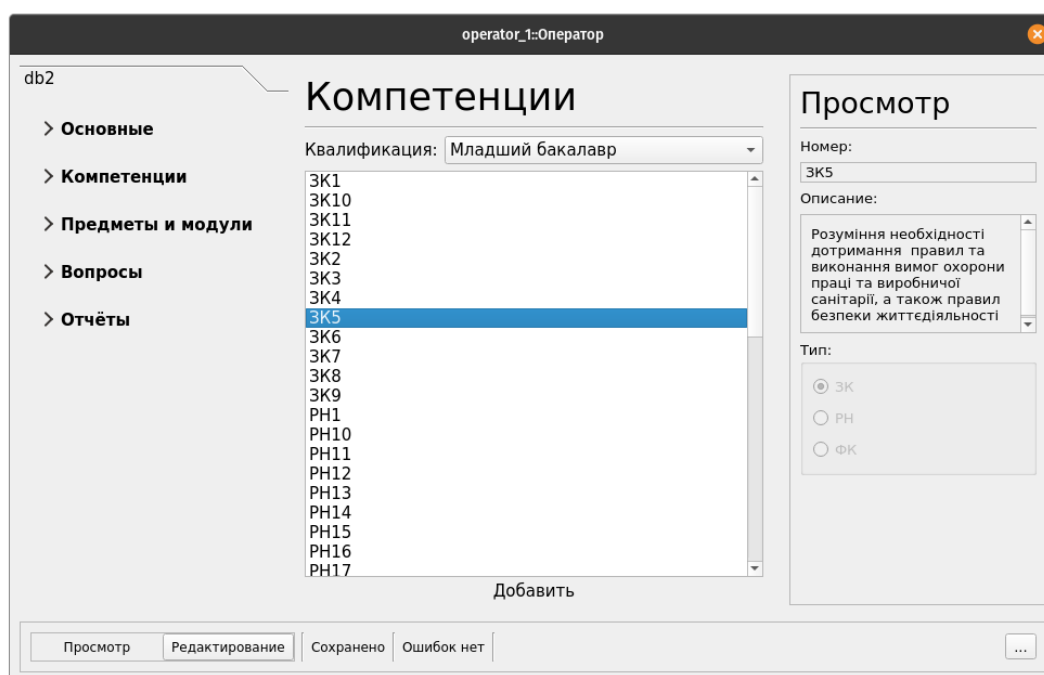


Рисунок 3.15 – Перегляд компетенцій

При перегляді або редагуванні компетенцій або предметів та модулів, по середині розміщено основну робочу область, з переглядом наявності об'єктів, можливістю їх додавання та видалення, а також вибором рівня компетенцій або модулів з якими бажає ознайомитися оператор. Праворуч знаходиться меню де-

тальної інформації обраного об'єкта. На рисунку 3.15 це меню детальної інформації компетенції. При виборі із стовпчика у робочій області бодь-якої компетенції, у меню праворуч з'являється детальна інформація до неї. У режимі перегляду меню грає роль довідкового вікна, у режимі редагування кожне поле буде активним і його можна буде відредагувати.

У нижньому «інформаційному барі» зображено стан середовища оператора на даний момент, а саме режим в якому зараз воно знаходиться, чи редагувалися дані з останнього збереження інформації, наявність помилок при заповненні бази даних, а також у крайньому кутку праворуч є кнопка, яка визиває «випадаюче» меню, де можна зберегти або відхилити зміни у базі даних.

		Вайчекаускас С.К.			ДП.ПЗ.161.03.ПЗ	Арк.
		Ланська С.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		43

4 АНАЛІЗ ДОСЛІДНОЇ ІНФОРМАЦІЇ

Коректність відображення даних, відсутність непрацюючих модулів, контроль коректності даних, що вводяться користувачем, контроль діяльності користувачів — всьому цьому необхідно приділяти чималу увагу, оскільки необроблені помилки можуть призвести до псування важливої інформації, викликати збої у роботі інших програм та можуть навіть призвести до псування обладнання.

У даному розділі описано аналіз можливих ситуацій, що були проаналізовані при проектуванні програмної оболонки, тобто ситуації, що можуть призвести до виникнення помилок.

Однією із можливих ситуацій виникнення помилки є додавання користувачем із роллю «Адміністратор» нових користувачів або баз даних. Проблема полягала у тому, що могли створюватися користувачі з однаковими логінами, що призводило до помилок при їх авторизації, або створення баз даних з однією й тією ж назвою, що хоч і не викликало критичних проблем у програмі, створювали дискомфорт у роботі. Проблему було вирішено шляхом перевірки введеної інформації і у випадку появи дублюючої інформації виведення відповідних повідомлень та вимкнення функцій зміни бази даних доки помилки не будуть вирішені. Рішення наведено на рисунках 4.1—4.2.

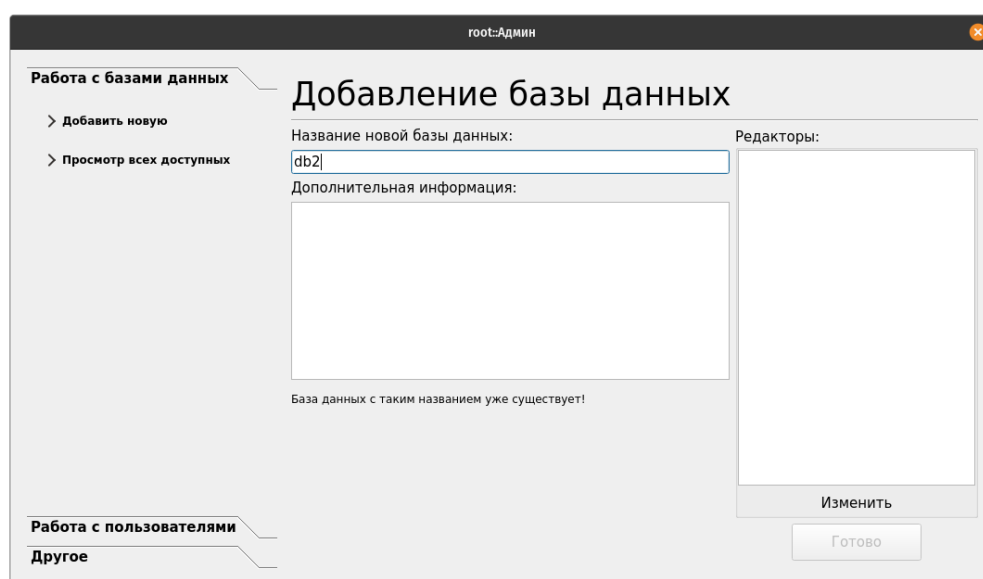


Рисунок 4.1 — Повідомлення про існуючу базу даних з такою назвою

		Вайчекаускас С.К.			ДП.ПЗ.161.03.ПЗ	Арк.
		Ланська С.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		43

root:Админ

Работа с базами данных

Работа с пользователями

> Добавить нового

> Просмотр всех доступных

Добавление нового пользователя

ФИО: Vaichekauskas S.K. Введено!

Роль:

☐ Администратор

☐ Оператор

☐ Эксперт

Обязательное поле!

Логин: admin Такой логин уже существует!

Пароль:

Повторите пароль:

Обязательное поле для ввода!

Доступные БД:

Изменить

Готово

Рекомендется сразу добавить доступные для редактирования базы данных, чтобы пользователь мог сразу приступить к работе. Этот параметр можно будет настроить и позже.

Другое

Рисунок 4.2 – Повідомлення про існуючого користувача з таким логіном

Ще одною проблемою було підключення до бази даних на новій машині, або перенесення серверу з однієї машини на іншу, оскільки ір-адреса серверу змінювалась, і клієнтська програма не могла підключитися. Проблема була вирішена додавання у вікно авторизації меню налаштувань, де користувач міг обрати за необхідністю новий порт або ір-адресу серверу та зміною алгоритму підключення до бази даних. Меню налаштувань зображено на рисунку 4.3, новий алгоритм приведено на лістингу 4.1.

Authorization

Настройки подключения:

IP-адрес: localhost

Порт: 3306

Отклонить Принять

Рисунок 4.3 – Меню налаштувань у вікні авторизації

Лістинг 4.1 — Алгоритм підключення до бази даних

```
void Authorization::on_pushButton_Entry_clicked()
{
    QSettings settings("settings.conf", QSettings::IniFormat);

    settings.beginGroup("LoginSettings");
    settings.setValue("UserLogin", ui->lineEdit_Login->text());
    settings.endGroup();

    db = QSqlDatabase::addDatabase("QMYSQL");
    db.setDatabaseName("study");
    db.setHostName(ui->lineEdit_HostAddress->text());
    db.setUserName(ui->lineEdit_Login->text());
    db.setPassword(ui->lineEdit_Password->text());
    db.setPort(ui->lineEdit_HostPort->text().toInt());

    bool ok = db.open();

    if(ok)
    {
        ui->label_Message->setText("Введённые данные верны!");

        QSqlQuery query;
        query.exec("SELECT * FROM users");

        while (query.next())
        {
            if(query.value(0).toString() == ui->lineEdit_Login->text())
            {
                if(query.value(1).toString() == "Admin")
                {
                    AdminMain *window = new AdminMain;
                    if(window->takeConnect(db))
                    {
                        db.close();
                        this->close();

                        window->show();
                    }
                }
                else
                {
                    ui->label_Message->setText("Не удалось подключиться!");
                }
            }
            else if(query.value(1).toString() == "Exp")
            {
                ExpertMain *window = new ExpertMain;
                if(window->takeConnect(db))
                {
                    db.close();
                    this->close();

                    window->show();
                }
                else
                {
                    ui->label_Message->setText("Не удалось подключиться!");
                }
            }
            else if(query.value(1).toString() == "Oper")
            {
                OperMain *window = new OperMain;
                if(window->takeConnect(db))
                {
                    db.close();
                    this->close();

                    window->show();
                }
                else
                {
                    ui->label_Message->setText("Не удалось подключиться!");
                }
            }
        }
    }
}
```

```

        {
            OperMain *window = new OperMain;
            if(window->takeConnect(db))
            {
                db.close();
                this->close();

                window->show();
            }
            else
            {
                ui->label_Message->setText("Не удалось подключиться!");
            }
        }

        break;
    }
    qDebug() << query.value(0).toString() << " " <<
query.value(1).toString() << endl;
    }
    else
        ui->label_Message->setText("Неверный логин или пароль!");
}

```

		Вайчекаускас С.К.			ДП.ПЗ.161.03.ПЗ	Арк.
		Ланська С.С.				43
Змн.	Арк.	№ докум.	Підпис	Дата		

5 ОХОРОНА ПРАЦІ

Постійне впровадження у всіх галузях прогресивних технологій і нової техніки викликає збільшення небезпечних і шкідливих факторів, що негативно впливають на здоров'я людини, тому охорона праці є найважливішою задачею по забезпеченню безпечних і не шкідливих умов праці.

Для забезпечення безпечних і нешкідливих умов праці необхідно, в першу чергу, створювати і впроваджувати таку нову техніку, технологічні процеси і матеріали, які б були надійними і безпечними в експлуатації.

Охорона праці — це система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних та лікувально-профілактичних заходів і засобів, спрямованих на збереження здоров'я та працездатності людини в процесі праці.

Якщо праця людини протікає у сприятливих умовах, вона сприяє розвитку всіх його здібностей, забезпечує широкі можливості для високопродуктивної і творчої роботи, сприяє зниженню аварійності та випадків виробничого травматизму. Саме тому охорона праці розглядається як одна з найважливіших економічних і соціальних задач не тільки окремого підприємства, але й держави в цілому.

Основними завданнями з безпеки праці є:

- розробка та впровадження високопродуктивних технологій;
- підвищення рівня безпеки діючого виробничого устаткування за рахунок ліквідації небезпечних та шкідливих виробничих факторів;
- удосконалення оснащення підприємств сучасними технічними способами безпеки, виробничої санітарії;
- комплекс соціальних та санітарно – оздоровчих заходів;
- підвищення культури організації виробництва;
- підвищення кваліфікації виробничого персоналу;
- впровадження уніфікованих стандартів;
- підвищення дисципліни праці.

		Вайчекаускас С.К.			ДП.ПЗ.161.03.ПЗ	Арк.
		Ланська С.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		43

5.1 Аналіз небезпечних та шкідливих виробничих чинників проекто- ваного технологічного процесу, об'єкту, система або пристрою

При роботі з персональним комп'ютером можуть мати місце такі фізичні і психологічні шкідливі фактори, як — порушення стану мікроклімату, недостатня освітленість робочої зони, забруднення повітря на робочих місцях, виробничий шум та вібрація, електромагнітні випромінювання, електростатичні поля, іонний склад повітря, відсутність чи недолік природного світла, поразка електричним струмом, загоряння, монотонність праці, перенапруга очей, емоційні перевантаження.

Виробниче освітлення.

Згідно приміщення, повинне мати природне і штучне освітлення. Денне (природне) освітлення приміщення відбувається за системою одnobічного бічного освітлення. Природне світло проникає у приміщення через світлові прорізи (віконні отвори), які мають регулювальні пристрої для відкривання. Також наявні штори (жалюзі) з можливістю захисту працюючих від прямого попадання сонячних променів і регулювання рівня освітленості в приміщенні.

Як джерело штучного освітлення в приміщенні, де встановлено ПК, бажано використовувати люмінесцентні лампи. Можливе застосування ламп розжарювання в світильниках місцевого освітлення. Освітленість робочого місця у горизонтальній площині на висоті 0,8 метрів від рівня підлоги повинна бути не менш 400 люкс. Вертикальна освітленість у площині екрану не більше 200 люкс. Для зменшення напруженості зору необхідно забезпечити достатньо рівномірне розподілення яскравості робочої поверхні відео монітора та навколишнього простору.

		Вайчекаускас С.К.			ДП.ПЗ.161.03.ПЗ	Арк.
		Ланська С.С.				43
Змн.	Арк.	№ докум.	Підпис	Дата		

5.2 Інженерно-технічні заходи з охорони праці

Інженерно-технічні заходи передбачають впровадження колективних заходів забезпечення сприятливих мікрокліматичних та зорових умов праці на робочих місцях, заходи захисту від впливу шкідливих речовин у повітрі робочої зони, від шуму, ультразвуку, вібрації, електромагнітного випромінювання, іонізуючого випромінювання, а також заходи попередження ураження електричним струмом, виникнення пожеж та аварій під час експлуатації технологічного устаткування.

Захист від шуму та вібрації.

Відомо, що шум несприятливо діє на слуховий аналізатор та інші органи та системи організму людини. Визначальне значення щодо такої дії має інтенсивність шуму, його частотний склад, тривалість щоденного впливу, індивідуальні особливості людини, а також специфіка виробничої діяльності.

Для забезпечення нормованих рівнів шуму у виробничих приміщеннях та на робочих місцях застосовуються шумопоглинальні засоби, вибір яких обґрунтовується спеціальними інженерно-акустичними розрахунками.

Як засоби шумопоглинання повинні застосовуватися негорючі або важкогорючі спеціальні перфоровані плити, панелі, мінеральна вата з максимальним коефіцієнтом звукопоглинання в межах частот від 31,5 до 8000 Герц, або інші матеріали аналогічного призначення, дозволені для оздоблення приміщень органами державного санітарно-епідеміологічного нагляду. Крім того, необхідно застосовувати підвісні стелі з аналогічними властивостями.

Рівні вібрації під час виконання робіт з ЕОМ у виробничих приміщеннях не повинні перевищувати допустимих значень, визначених в СН 3044-84 та ГОСТ 12.1.012-90 «Вибрационная безопасность. Общие требования».

Для зниження вібрації обладнання, пристрої, пристосування необхідно встановлювати на спеціальні амортизуючі прокладки, передбачені нормативними документами.

		Вайчекаускас С.К.			ДП.ПЗ.161.03.ПЗ	Арк.
		Ланська С.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		43

Регулювання параметрів мікроклімату.

Параметри мікроклімату, іонного складу повітря, вміст шкідливих речовин на робочих місцях, оснащених відеотерміналами, повинні відповідати вимогам ГОСТ 12.1.005-88 «ССБТ Общие санитарно-гигиенические требования к воздуху рабочей зоны», представлено на таблиці 5.1, СН 2152-80 «Санітарно-гігієнічні норми допустимих рівнів іонізації повітря виробничих та громадських приміщень», затверджених Міністерством охорони здоров'я СРСР, представлено на таблиці 5.2. А для підтримки допустимих значень мікроклімату та концентрації позитивних та негативних іонів необхідно передбачати установки або прилади зволоження або штучної іонізації, кондиціювання повітря.

Таблиця 5.1 – Нормовані параметри мікроклімату для приміщень з ВДТ та ПЕОМ

Пора року	Категорія робіт згідно з ГОСТ 12.1-005-88	Температура повітря, °С оптимальна	Відносна вологість повітря, % оптимальна	Швидкість руху повітря, м/с оптимальна
Холодна	Легка - 1а	від 22 до 24	від 40 до 60	0,1
	Легка - 1б	від 21 до 23	від 40 до 60	0,1
Тепла	Легка - 1а	від 23 до 25	від 40 до 60	0,1
	Легка - 1б	від 22 до 24	від 40 до 60	0,2

Таблиця 5.2 – Рівні іонізації повітря приміщень при роботі на ВДТ та ПЕОМ (відповідно до СН 2152-80)

Рівні	Кількість іонів в 1 см ³ повітря	
	n +	n -
	400	600
Мінімально необхідні	400	600
Оптимальні	від 1500 до 3000	від 3000 до 5000
Максимально допустимі	50000	50000

5.3 Пожежна профілактика

З огляду на можливість виникнення пожежі слід з'ясувати, які речовини і матеріали можуть горіти. У приміщенні, що розглядається, можуть горіти вироби з дерева, пластмас, тканини і паперу. Горючі рідини, пил та волокна у приміщенні не використовуються і не виділяються. Тому приміщення, що аналізується, відноситься, відповідно до нормативної документації, до зони П-Па і до категорії пожежної небезпеки В.

Ймовірними причинами виникнення пожежу можуть бути несправність 95 електрообладнання (кабелів, розеток), короткі замикання внаслідок виходу з ладу чи експлуатації несправного електроустаткування (ПЕОМ, периферійних пристроїв), порушення правил протипожежної безпеки тощо.

Експлуатація ліній електромережі практично повністю унеможливорює виникнення електричного джерела загоряння в наслідок короткого замикання та перевантаження проводів. Застосовуються дроти з важкогорючою і негорючою ізоляцією.

Для своєчасного попередження пожеж та підвищення оперативності реагування при їх виникненні у приміщенні використовується такий комплекс заходів:

- обов'язковий інструктаж персоналу з питань охорони праці;
- зокрема, правила пожежної безпеки у приміщеннях з ЕОМ;
- заборона використання відкритого вогню у приміщенні;
- наявність системи автоматичної пожежної сигналізації з димовими пожежними оповіщувачами;
- ступінь вогнестійкості будівлі, у якій розташовано приміщення – II;
- наявність шляхів евакуації при виникненні пожежі;
- розміщення схеми евакуації людей при пожежі і ознайомлення з нею персоналу.

		Вайчекаускас С.К.			ДП.ПЗ.161.03.ПЗ	Арк.
		Ланська С.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		43

5.4 Заходи з ергономіки

Проектування робочих місць, забезпечених відеотерміналами, відноситься до числа важливих проблем ергономічного проектування в області обчислювальної техніки.

Робоче місце і взаємне розташування всіх його елементів повинне відповідати антропометричним, фізичним і психологічним вимогам. Велике значення має також характер роботи. Зокрема, при організації робочого місця програміста повинні бути дотримані наступні основні умови: оптимальне розміщення устаткування, що до складу робочого місця і достатній робочий простір, що дозволяє здійснювати всі необхідні рухи і переміщення.

Ергономічними аспектами проектування відеотермінальних робочих місць, зокрема, є: висота робочої поверхні, розміри простору для ніг, вимоги до розташування документів на робочому місці (наявність і розміри підставки для документів, можливість різного розміщення документів, відстань від очей користувача до екрану, документа, клавіатури і т.д.), характеристики робочого крісла, вимоги до поверхні робочого столу, регульованість елементів робочого місця.

Головними елементами робочого місця програміста є стіл і крісло. Основним робочим положенням є положення сидячи, яке зображено на рисунку 5.1.

Робоча поза сидячи викликає мінімальне стомлення програміста. Рациональне планування робочого місця передбачає чіткий порядок і сталість розміщення предметів, засобів праці і документації. Те, що потрібно для виконання робіт частіше, розташоване в зоні легкої досяжності робочого простору.

Моторне поле — простір робочого місця, в якому можуть здійснюватися рухові дії людини.

Максимальна зона досяжності рук — це частина моторного поля робочого місця, обмеженого дугами, описуваними максимально витягнутими руками при русі їх у плечовому суглобі.

		Вайчекаускас С.К.			ДП.ПЗ.161.03.ПЗ	Арк.
		Ланська С.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		43

Оптимальна зона — частина моторного поля робочого місця, обмеженого дугами, описуваними передпліччями при русі в ліктьових суглобах з опорою в точці ліктя і з відносно нерухомим плечем.

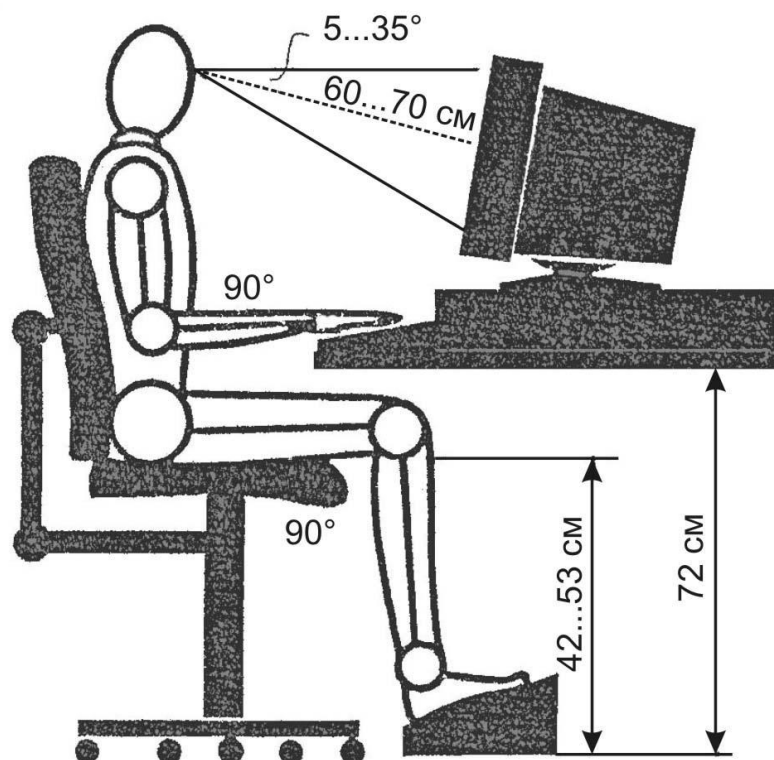


Рисунок 5.1 – Схема положення за робочим місцем

6 ЕКОНОМІЧНИЙ РОЗДІЛ

Згідно із завданням дипломного проекту необхідно визначити собівартість і ціну програмної оболонки створення бази знань для обліку зарахованих дисциплін в системі безперервної освіти. Для виконання розрахунку були використані початкові дані, представлені в таблиці 6.1.

Таблиця 6.1 – Початкові дані для розрахунку

Найменування початкових даних	Показник	Джерело отримання
1 Трудомісткість складання програми	39 днів	Фактичні витрати часу на розробку програми
2 Місячна ставка оператора – укладача програми	5200,00	Дані переддипломної практики
3 Кількість годин в місяці	310	Кількість робочих днів
4 Додаткова зарплата	10%	Дані переддипломної практики
5 Нарахування єдиного соціального внеску	22%	Дані переддипломної практики
6 Загальновиробничі витрати (%)	15%	Дані переддипломної практики
7 ПДВ (податок на додану вартість)	20%	Дані переддипломної практики

Стаття 1. Матеріали не використовуються.

Стаття 2. Електроенергія:

$$BE = \Phi\text{Ч} * BKв * CE, \quad (6.1)$$

де BE – вартість електроенергії;

$\Phi\text{Ч}$ – фактичний час;

$BKв$ – вартість 1 Кв електроенергії;

CE – електроенергія, яку споживає ноутбук за 1 годину.

$$BE = 310 * 1,68 \text{ грн} * 0.5 \text{ Кв} = 268,80 \text{ грн}$$

Стаття 3. Основна заробітна плата:

$$З_{осн} = l_{год} \times T_{год} , \quad (6.2)$$

де $l_{год}$ – годинна тарифна ставка оператора, грн.;

$T_{год}$ – кількість годин у місяці, приймається 160 – вихідні дані.

Визначаємо годинну тарифну ставку оператора:

$$l_{год} = \frac{L_{міс}}{T_{год}} , \quad (6.3)$$

де $L_{міс}$ – місячна ставка оператора, грн.

$$l_{год} = 5200,00 / 160 = 32,50 \text{ грн/год} , \quad (6.4)$$

$$ЗП = 40 * 8 * 32,50 = 10400,00 \text{ грн}$$

Стаття 4. Додаткова заробітна плата:

$$З_{дод} = \frac{З_{осн} \times Д\%}{100} , \quad (6.5)$$

де $З_{дод}$ – додаткова заробітна плата, грн.;

$Д\%$ – відсоток додаткової заробітної плати, приймається 10% – вихідні дані.

$$Д = 10400,00 * 10\% = 1040,00 \text{ грн}$$

Стаття 5. Нарахування єдиного соціального внеску:

$$З_{соц} = \frac{(З_{осн} + З_{дод}) \times С\%}{100} , \quad (6.6)$$

$$З = (10400,00 + 1040,00) * 22\% = 2516,80 \text{ грн}$$

		Вайчекаускас С.К.			ДП.ПЗ.161.03.ПЗ	Арк.
		Ланська С.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		43

Стаття 6. Загальвиробничі витрати:

$$З_{заг} = \frac{З_{осн} \times H_1 \%}{100} , \quad (6.7)$$

де $З_{заг}$ – загальвиробничі витрати, грн.;

$H_1 \%$ – відсоток загальвиробничих витрат, приймається 15% – вихідні дані.

$$З_{заг} = 10400,00 \times 15\% = 1560,00 \text{ грн}$$

Виробнича собівартість:

$$S_{nn} = З_{к} + З_{осн} + З_{дод} + З_{соц} + З_{заг} , \quad (6.8)$$

= ст.1-ст.6 = сума

Прибуток підприємства:

$$П_n = S_{nn} \times \frac{П_n \%}{100} , \quad (6.9)$$

де $П_n$ – прибуток підприємства, грн.;

$П_n \%$ – відсоток прибутку підприємства, приймається 10% – вихідні дані.

$$П = 16095,60 \times 10\% = 1609,56 \text{ грн}$$

Ціна підприємства:

$$Ц_n = S_{nn} + П_n , \quad (6.10)$$

де $Ц_n$ – ціна підприємства, грн.

$$Ц = 16095,60 + 1609,56 = 17705,16 \text{ грн}$$

Податок на додану вартість:

$$ПДВ = Ц_n \times \frac{ПДВ \%}{100} , \quad (6.11)$$

		Вайчекаускас С.К.			ДП.ПЗ.161.03.ПЗ	Арк.
		Ланська С.С.				43
Змн.	Арк.	№ докум.	Підпис	Дата		

де $ПДВ\%$ – відсоток податку на додану вартість, приймається 20% – діюча ставка $ПДВ$ на сучасний момент.

$$ПДВ = 17705,16 * 20\% = 3541,03 \text{ грн}$$

Ціна для замовника:

$$Ц_{\text{зам}} = Ц_{\text{п}} + ПДВ, \quad (6.12)$$

$$Ц = 17705,16 + 3541,03 = 21246,19 \text{ грн}$$

Таблиця 6.2 – Планова калькуляція виробничої собівартості, ціни підприємства й ціни для замовника на виконання розробки програмної оболонки створення бази знань для обліку зарахованих дисциплін в системі безперервної освіти

Статті калькуляції	Сума, грн.
Стаття 1 Спожиті ресурси інтернет	310,00
Стаття 2 Електроенергія	268,80
Стаття 3 Основна заробітна плата	10400,00
Стаття 4 Додаткова заробітна плата	1040,00
Стаття 5 Відрахування в соціальні фонди	2516,80
Стаття 6 Загальновиробничі витрати	1560,00
Виробнича собівартість	16095,60
Прибуток підприємства	1609,56
Ціна підприємства	17705,16
Податок на додану вартість	3541,03
Ціна для замовника	21246,19

Висновок: таким чином розрахунок показав, що собівартість програми програмної оболонки створення бази знань для обліку зарахованих дисциплін в системі безперервної освіти складає 16095,60 грн., якщо коледж продаватиме цю програму, то її ціна для споживача складе 21246,19 грн. При цьому з кожного

екземпляра проданої програми коледж матиме прибуток 1609,56 грн. Найбільшу питому вагу складають витрати на оплату заробітної плати.

		Вайчекаускас С.К.			ДП.ПЗ.161.03.ПЗ	Арк.
		Ланська С.С.				43
Змн.	Арк.	№ докум.	Підпис	Дата		

ВИСНОВКИ

Під час роботи над дипломним проектом було створено експертну систему для формування індивідуальних навчальних планів в системі безперервної підготовки фахівців. За допомогою засобів Qt Creator було створено програму, яка дозволяє вносити існуючі навчальні плани та на їх основі формувати вступні іспити для фахівців і послідовні навчальні плани для системи «безперервної освіти».

Програмна оболонка забезпечує роботу з моделями бази даних «Дисципліна», «Модуль», «Компетенція», «Відповідності», «Питання для тесту». Для кожної моделі оболонка забезпечує можливість перегляду, додавання даних, редагування та видалення.

Розроблена програма має інтуїтивно-зрозумілий графічний інтерфейс, а користувач потребує лише базових знань роботи з комп'ютером.

Проведений економічний розрахунок показав, що собівартість проекту складе 16095,60 грн., якщо коледж продаватиме цей додаток, то його ціна для споживача складе 21246,19 грн. При цьому з кожного екземпляра проданої програми коледж матиме прибуток 1609,56 грн.

Розроблена пояснювальна записка акцентує увагу на інтерфейсі програмного продукту та інструкції користувача, що спрощує його використання.

Програма може бути корисною для вищих навчальних закладів працюючих із системою «безперервної освіти», оскільки може звести до мінімуму помилки при складанні інтегрованих навчальних планів і вступних іспитів, а також спростити та пришвидшити роботу.

		Вайчекаускас С.К.			ДП.ПЗ.161.03.ПЗ	Арк.
		Ланська С.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		43

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

- 1 Закон України про охорону праці [Електронний ресурс] // zakon3.rada.gov.ua. Режим доступу: <http://zakon3.rada.gov.ua/laws/show/2694-12>.
- 2 Безпека при роботі з електронно-обчислювальними машинами [Електронний ресурс] // 4exam.info. Режим доступу: [http://4exam.info/book_224_glava_27_4.10._Bezpeka_pri_roboti_z_elektronno-obchisljuvalnimimashinami_\(EOM\).html](http://4exam.info/book_224_glava_27_4.10._Bezpeka_pri_roboti_z_elektronno-obchisljuvalnimimashinami_(EOM).html).
- 3 Батиршин І.З. Основні операції нечіткої логіки та їх узагальнення. Казань: Отечество, 2001. — 102 с.
- 4 Алтунін А.Е., Семухін М.В. Моделі та алгоритми прийняття рішень в нечітких умовах. Тюмень: Видав-во Тюменського державного університету, 2000.— 352 с.
- 5 Нейлор, К. Как построить свою экспертную систему / К. Нейлор. - М.: Энергоатомиздат, 2013. — 286 с.
- 6 Нильсон, Н. Принципы искусственного интеллекта / Н. Нильсон. - М.: Радио и связь, 2014. — 373 с.
- 7 Електронний научно-практический журнал «Современная техника и технологии» [Електронний ресурс] // technology.snauka.ru. Режим доступу: <http://technology.snauka.ru/2016/12/11465>.
- 8 «Qt/C++ - Урок 013. QMenu - контекстное меню в QtableView» [Електронний ресурс] // <https://evileg.com> Режим доступу: <https://evileg.com/ru/post/76/>.»
- 9 «Qt 4.7: Описание класса QTreeWidgetItem | Документация» [Електронний ресурс] // <http://doc.crossplatform.ru> Режим доступу: <http://doc.crossplatform.ru/qt/4.7.x/qtreewidget.html>.
- 10 М. Шлеє «Qt 5.3. Профессиональное программирование на C++», БХВ-Петербург, 2015.
- 11 Справочное руководство по MySQL [Електронний ресурс] // <http://www.mysql.ru> Режим доступу: <http://www.mysql.ru/docs/man/What-is.html>

		Вайчекаускас С.К.			ДП.ПЗ.161.03.ПЗ	Арк.
		Ланська С.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		43

Лістинг програми

Лістинг 1 – Специфікація класу форми адміністратора

```
#ifndef ADMINMAIN_H
#define ADMINMAIN_H

#include <QDialog>
#include <QtSql>
#include <QDebug>
#include <QVariant>
//#include <QAxObject>
#include <QPrinter>
#include <QPainter>
#include <QFileDialog>
#include <QMessageBox>

#include "database.h"
#include "databasebuffer.h"
#include "dropbuttondb.h"
#include "usersselectiondialog.h"
#include "informationdb_dialog.h"
#include "databaseselectiondialog.h"
#include "informationuser_dialog.h"

namespace Ui {
class AdminMain;
}

class AdminMain : public QDialog
{
    Q_OBJECT

    QSqlDatabase db;
    Database mydb;
    QPrinter printer;

    DatabaseBuffer db_buffer;

    QVBoxLayout *mainlayoutDB = new QVBoxLayout();
    QWidget *newWidgetDB;
    QVBoxLayout *layoutDB = new QVBoxLayout();

    QVBoxLayout *mainlayoutUser = new QVBoxLayout();
    QWidget *newWidgetUser;
    QVBoxLayout *layoutUser = new QVBoxLayout();
    QSpacerItem *my_spacer = new QSpacerItem(0,0, QSizePolicy::Expanding, QSizePolicy::Expanding);

public:
    explicit AdminMain(QWidget *parent = nullptr);
```

```

~AdminMain();

bool takeConnect(QSqlDatabase d);

void addDropButton_DB(QString name);
void clear_viewdb();

void addDropButton_User(QString fio);
void clear_viewuser();

public slots:
    void slot_getInfoFrom_UserSelectionDialog(QList <QString> resultList);
    void slot_getInfoFrom_DatabaseSelectionDialog(QList <QString> resultList);
    void slot_showDetails_DB(QString nameDB);
    void slot_showDetails_User(QString nameUser);
    //void slot_deleteDB(QString nameDB);

private slots:
    void on_pushButton_clicked();

    void on_pushButton_2_clicked();

    void on_pushButton_Return_clicked();

    void on_pushButton_Open_clicked();

    void on_pushButton_Gotovo_clicked();

//    void on_tableWidget_Zachet_itemClicked(QTableWidgetItem
*item);

    void on_tableWidget_Zachet_cellClicked(int row, int column);
    void slotPushButtonExportDoc_clicked();
    void slotAddNewDB();

    void on_pushButton_ExportPdf_clicked();

    void on_commandLinkButton_AddNewDB_clicked();

    void on_commandLinkButton_ViewDBs_clicked();

    void on_commandLinkButton_AddNewUser_clicked();

    void on_commandLinkButton_ViewUsers_clicked();

    void on_lineEdit_NameDB_editingFinished();

    void on_pushButton_AddNewDB_Next_1_clicked();

    void on_pushButton_InMainPage_DB_clicked();

```

```

void on_pushButton_AddMoreDB_clicked();

void on_pushButton_ChangeUsers_clicked();

void on_plainTextEdit_InformationDB_textChanged();

void on_lineEdit_FIO_editingFinished();

void on_radioButton_Admin_clicked();

void on_radioButton_Operator_clicked();

void on_radioButton_Expert_clicked();

void on_lineEdit_UserLogin_editingFinished();

void on_lineEdit_UserPassword_1_editingFinished();

void on_lineEdit_UserPassword_2_editingFinished();

void on_pushButton_ChangeTableDB_clicked();

void on_pushButton_SearchDB_clicked();

private:
    Ui::AdminMain *ui;
};

#endif // ADMINMAIN_H

```

Лістинг 2 – Специфікація класу форми оператора

```

#ifndef OPERMAIN_H
#define OPERMAIN_H

#include <QDialog>
#include <QtSql>
#include <QDebug>
#include <QVariant>
//#include <QAxObject>
#include <QPrinter>
#include <QPainter>
#include <QFileDialog>
#include <QMessageBox>
#include <QListWidgetItem>

#include "database.h"
#include "databasebuffer.h"

namespace Ui {
class OperMain;
}

```

```

class OperMain : public QWidget
{
    Q_OBJECT

    QSqlDatabase db;
    Database mydb;
    QPrinter printer;

    DatabaseBuffer db_buffer;

    bool Mode = true;

public:
    explicit OperMain(QWidget *parent = nullptr);
    ~OperMain();

    bool takeConnect(QSqlDatabase d);

private slots:
    void on_listWidget_ListDatabases_currentRowChanged(int
currentRow);

    void on_pushButton_Open_clicked();

    void on_commandLinkButton_clicked();

    void on_commandLinkButton_2_clicked();

    void on_commandLinkButton_3_clicked();

    void on_commandLinkButton_4_clicked();

    void on_listWidget_Competences2_currentRowChanged(int
currentRow);

    void on_listWidget_Competences1_currentRowChanged(int
currentRow);

    void on_comboBox_SetRange_currentIndexChanged(int index);

    void on_listWidget_ListCompetences_currentRowChanged(int
currentRow);

    void on_listWidget_ListQuestions_currentRowChanged(int
currentRow);

private:
    Ui::OperMain *ui;
};
#endif // OPERMAIN_H

```

Лістинг 3 – Специфікація класу форми експерта

```
#ifndef EXPERTMAIN_H
#define EXPERTMAIN_H

#include <QDialog>
#include <QtSql>
#include <QDebug>
#include <QVariant>
// #include <QAxObject>
#include <QPrinter>
#include <QPainter>
#include <QFileDialog>
#include <QMessageBox>
#include <QListWidgetItem>

#include "database.h"
#include "databasebuffer.h"

namespace Ui {
class ExpertMain;
}

class ExpertMain : public QWidget
{
    Q_OBJECT

    QSqlDatabase db;
    Database mydb;
    QPrinter printer;

    DatabaseBuffer db_buffer;

    void ValueCoefUpdate();

public:
    explicit ExpertMain(QWidget *parent = nullptr);
    ~ExpertMain();

    bool takeConnect(QSqlDatabase d);

private slots:
    void on_listWidget_ListDatabases_currentRowChanged(int
currentRow);

    void on_pushButton_Open_clicked();

    void on_listWidget_Competence1_currentRowChanged(int
currentRow);

    void on_listWidget_Competence2_currentRowChanged(int
currentRow);
```

```

        void on_pushButton_Set_clicked();

        void
on_listWidget_Competence1_currentItemChanged(QListWidgetItem
*current, QListWidgetItem *previous);

        void
on_listWidget_Competence2_currentItemChanged(QListWidgetItem
*current, QListWidgetItem *previous);

        void on_pushButton_Cancel_clicked();

        void
on_listWidget_ListDatabases_currentItemChanged(QListWidgetItem
*current, QListWidgetItem *previous);

private:
    Ui::ExpertMain *ui;
};

#endif // EXPERTMAIN_H

```

Лістинг 3 – Програми адміністратора

```

#include "adminmain.h"
#include "ui_adminmain.h"

AdminMain::AdminMain(QWidget *parent) :
    QDialog(parent),
    ui(new Ui::AdminMain)
{
    ui->setupUi(this);
    //this->resize(800, 600);

    //connect(ui->pushButton_ExportDoc, SIGNAL(clicked()), this,
    SLOT(slotPushButtonExportDoc_clicked()));

    //connect(ui->commandLinkButton_AddNewDB, SIGNAL(clicked()),
    this, SLOT(slotAddNewDB()));
}

AdminMain::~AdminMain()
{
    delete newWdgetDB;
    delete mainlayoutDB;
    delete layoutDB;
    delete ui;
}

bool AdminMain::takeConnect(QSqlDatabase d)

```



```

{
    newWdgetDB = new QWidget(ui->scrollArea_ViewDB);

    ui->scrollArea_ViewDB->setLayout(mainlayoutDB);
    ui->scrollArea_ViewDB->setWidget(newWdgetDB);
    newWdgetDB->setLayout(layoutDB);
    layoutDB->setSpacing(0);

    layoutDB->addItem(my_spacer);

    newWdgetUser = new QWidget(ui->scrollArea_ViewUsers);

    ui->scrollArea_ViewUsers->setLayout(mainlayoutUser);
    ui->scrollArea_ViewUsers->setWidget(newWdgetUser);
    newWdgetUser->setLayout(layoutUser);
    layoutUser->setSpacing(0);

    layoutUser->addItem(my_spacer);

    db = d;
    qDebug() << db.userName();

    this->setWindowTitle(db.userName()+"::Админ");

    if(db.open())
    {
        ui->stackedWidget_WorkSpace->setCurrentIndex(0);
        db_buffer.setDB(&db);

        return true;
    }

    return false;
}

void AdminMain::on_pushButton_clicked()
{
    mydb.Calculate(ui->spinBox_hours->value(), ui-
>doubleSpinBox_coeff->value());

    ui->listWidget_Temu->clear();
    ui->tableWidget_Zachet->setRowCount(0);

    if(ui->checkBox_Visble_Predmet->isChecked())
    {
        ui->groupBox_Zachet->setVisible(true);

        qDebug() << "1/3";

        vector <QString> predmet;
        vector <int> kredit;
    }
}

```

```

for(int i = 0; i < mydb.result_mod.size(); i++)
{
    bool flag = false;
    int number;

    for(int j = 0; j < predmet.size(); j++)
        if(predmet[j] == mydb.result_mod[i].discipline)
        {
            flag = true;
            number = j;
        }
    qDebug() << "1.1/3";

    if(!flag)
    {
        predmet.push_back(mydb.result_mod[i].discipline);
        kredit.push_back(mydb.result_mod[i].hours);
    }
    else
        kredit[number] = kredit[number] +
mydb.result_mod[i].hours;

    qDebug() << "1.2/3";
}

qDebug() << "2/3";

for(int i = 0; i < predmet.size(); i++)
{
    ui->tableWidget_Zachet->insertRow(ui-
>tableWidget_Zachet->rowCount());
    ui->tableWidget_Zachet->setItem(ui-
>tableWidget_Zachet->rowCount()-1, 0, new
QTableWidgetItem(predmet[i]));
    ui->tableWidget_Zachet->setItem(ui-
>tableWidget_Zachet->rowCount()-1, 1, new
QTableWidgetItem((QVariant(kredit[i]/30)).toString()));
}

}
else
    ui->groupBox_Zachet->setVisible(false);

if(ui->groupBox_ChekModule->isChecked())
{
    ui->groupBox_Temu->setVisible(true);

    if(ui->radioButton_All->isChecked())
    {
        for(int i = 0; i < mydb.result_mod.size(); i++)
            ui->listWidget_Temu-
>addItem(mydb.result_mod[i].name);
    }
}

```

```

        }
    }
    else
        ui->groupBox_Temu->setVisible(false);

    qDebug() << "3/3";

    if(ui->checkBox_Form_Quest->isChecked())
    {
        ui->groupBox_Tests->setVisible(true);
        ui->label_Tests->setText("");

        for(int i = 0; i < mydb.result_question.size(); i++)
        {
            ui->label_Tests->setText(ui->label_Tests->text() + "\n" +
                QVariant(i+1).toString() + ". " +
                mydb.result_question[i].text + "\na) " + mydb.result_question[i].a +
                "\nb) " + mydb.result_question[i].b + "\nc) " +
                mydb.result_question[i].c + "\nd) " + mydb.result_question[i].d +
                "\n");
        }
    }
    else
    {
        ui->groupBox_Tests->setVisible(false);
    }

    ui->stackedWidget_MainWidget->setCurrentIndex(2);
}

void AdminMain::on_pushButton_2_clicked()
{
    ui->stackedWidget_MainWidget->setCurrentIndex(0);
}

void AdminMain::on_pushButton_Return_clicked()
{
    ui->stackedWidget_MainWidget->setCurrentIndex(1);
}

void AdminMain::on_pushButton_Open_clicked()
{
    db.setDatabaseName(ui->listWidget_Databases->item(ui->listWidget_Databases->currentRow())->text());
    qDebug() << ui->listWidget_Databases->item(ui->listWidget_Databases->currentRow())->text();

    Database d;
    mydb = d;

    if(mydb.LoadData(&db))
        ui->stackedWidget_MainWidget->setCurrentIndex(1);
    else

```

```

        {
            QMessageBox msgBox;
            msgBox.setText("Невозможно извлечь информаию из базы дан-
ных!\n");
            msgBox.exec();
        }
    }

void AdminMain::on_pushButton_Gotovo_clicked()
{
    ui->stackedWidget_MainWidget->setCurrentIndex(0);
}

//void
AdminMain::on_tableWidget_Zachet_itemClicked(QTableWidgetItem
*item)
//{
//    ui->listWidget_Temu->clear();

//    for(int i = 0; i < mydb.result_mod.size(); i++)
//    {
//        if(mydb.result_mod[i].discipline == ui-
>tableWidget_Zachet->selectedItems()[0]->text())
//            ui->listWidget_Temu-
>addItem(mydb.result_mod[i].name);
//    }
//}

void AdminMain::on_tableWidget_Zachet_cellClicked(int row, int
column)
{
    if(ui->radioButton_Otdelno->isChecked())
    {
        ui->listWidget_Temu->clear();

        for(int i = 0; i < mydb.result_mod.size(); i++)
        {
            if(mydb.result_mod[i].discipline == ui-
>tableWidget_Zachet->selectedItems()[0]->text())
                ui->listWidget_Temu-
>addItem(mydb.result_mod[i].name);
        }
    }
}

void AdminMain::slotPushButtonExportDoc_clicked()
{
    //    QAxObject *pword = new QAxObject("Word.Application");
    //    QAxObject *pdoc = pword->querySubObject("Documents");
    //    pdoc = pdoc->querySubObject("Add()");

    //    QAxObject *prange = pdoc->querySubObject("Range()");
    //    prange->dynamicCall("SetRange(int, int)", 0, 100);

```

```

        // prange->setProperty("Text", ui->label_Tests->text());

        // QAxObject *pfont = prange->querySubObject("Font");
        // pfont->setProperty("Size", 14);
        // pfont->setProperty("Name", "Times New Roman");

        // pword->setProperty("Visible", true);

        //pdoc->dynamicCall("SaveAs()", "tests", "doc");
    }

void AdminMain::on_pushButton_ExportPdf_clicked()
{
    QString fileName = QFileDialog::getSaveFileName((QWidget* )0,
"%HOMEPATH%\\test", QString(), "*.pdf");
    if (QFileInfo(fileName).suffix().isEmpty())
    { fileName.append(".pdf"); }

    QPrinter printer(QPrinter::PrinterResolution);
    printer.setOutputFormat(QPrinter::PdfFormat);
    printer.setPaperSize(QPrinter::A4);
    printer.setOutputFileName(fileName); // устанавливаем путь к
pdf файлу

    QTextDocument doc;
    //doc.setHtml("<h1>" + ui->label_Tests->text() + "</h1>");
    doc.setPlainText(ui->label_Tests->text());
    doc.setPageSize(printer.pageRect().size()); // This is
necessary if you want to hide the page number
    doc.print(&printer);

    // // работаем с отрисовкой pdf через QPainter
    // QPainter p(&printer);
    // // выполняем действия с p
    // p.setBrush(QBrush(Qt::white, Qt::SolidPattern));
    // p.drawText(0, 0, ui->label_Tests->text());
    // p.end();
}

void AdminMain::slotAddNewDB()
{};

void AdminMain::on_commandLinkButton_AddNewDB_clicked()
{
    db_buffer.DownloadUsers();
    db_buffer.DownloadDatabaseName();
    ui->stackedWidget_WorkSpace->setCurrentIndex(1);
}

void AdminMain::on_commandLinkButton_ViewDBs_clicked()
{
    db_buffer.DownloadDatabase();
}

```

```

clear_viewdb();

QList <DatabaseInfo> tempDBs = db_buffer.getDBList();

for(int i = 0; i < tempDBs.size(); i++)
{
    addDropButton_DB(tempDBs.at(i).name);
}

ui->stackedWidget_WorkSpace->setCurrentIndex(3);
}

void AdminMain::on_commandLinkButton_AddNewUser_clicked()
{
    db_buffer.DownloadDatabaseName();
    db_buffer.DownloadUsers();
    ui->stackedWidget_WorkSpace->setCurrentIndex(4);
}

void AdminMain::on_commandLinkButton_ViewUsers_clicked()
{
    db_buffer.DownloadUsers();

    clear_viewuser();

    for(int i = 0; i < db_buffer.getUsersList().size(); i++)
    {
        addDropButton_User(db_buffer.getUsersList().at(i).fio);
    }

    ui->stackedWidget_WorkSpace->setCurrentIndex(6);
}

void AdminMain::on_lineEdit_NameDB_editingFinished()
{
    if(db_buffer.FindDatabaseName(ui->lineEdit_NameDB->text()))
    {
        if(ui->lineEdit_NameDB->text().size() != 0)
            ui->label_WarningsNameDB->setText("База данных с таким
названием уже существует!");
        else
            ui->label_WarningsNameDB->setText("Название не введе-
но!");

        ui->pushButton_AddNewDB_Next_1->setEnabled(false);
    }
    else
    {
        ui->label_WarningsNameDB->setText("Название базы данных
введено верно!");

        if(!ui->plainTextEdit_InformationDB->toPlainText().size())

```

```

        ui->label_WarningsMoreInformation->setText("Дополни-
тельная информация не введена.\nРекомендуется внести данные, кото-
рые помогут пользователю работать с ней.");
    else
        ui->label_WarningsMoreInformation->setText("Дополни-
тельная информация введена!");

        if(!ui->tableWidget_Users->rowCount())
            ui->label_WarningsUsers->setText("Не назначены пользо-
ватели, которые могут редактировать БД.\nРекомендуется сразу вне-
сти эту информацию, чтобы пользователи могли сразу приступить к
работе.");
        else
            ui->label_WarningsUsers->setText("Редакторы назна-
чены!");

        ui->pushButton_AddNewDB_Next_1->setEnabled(true);
    }
}

void AdminMain::on_pushButton_AddNewDB_Next_1_clicked()
{
    ui->stackedWidget_WorkSpace->setCurrentIndex(2);
}

void AdminMain::on_pushButton_InMainPage_DB_clicked()
{
    ui->stackedWidget_WorkSpace->setCurrentIndex(0);
}

void AdminMain::on_pushButton_AddMoreDB_clicked()
{
    ui->commandLinkButton_AddNewDB->clicked();
}

void AdminMain::on_pushButton_ChangeUsers_clicked()
{
    UsersSelectionDialog *SelectionDialog = new
UsersSelectionDialog;
    SelectionDialog->setTables(db_buffer.getUsersList());
    connect(SelectionDialog, SIGNAL(takeResult(QList <QString>)),
this, SLOT(slot_getInfoFrom_UserSelectionDialog(QList
<QString>)));
    SelectionDialog->exec();
}

void AdminMain::slot_getInfoFrom_UserSelectionDialog(QList
<QString> resultList)
{
    ui->tableWidget_Users->setRowCount(0);

    QList <User> tempList = db_buffer.getUsersList();
    int x = 0;

```

```

bool flag;

while(x < tempList.size())
{
    flag = false;

    for(int i = 0; i < resultList.size(); i++)
    {
        if(resultList.at(i) == tempList.at(x).fio)
        {
            flag = true;
            resultList.removeAt(i);
            x++;
            i = resultList.size();
        }
    }

    if(!flag)
    {
        tempList.removeAt(x);
    }
}

for(int i = 0; i < tempList.size(); i++)
{
    ui->tableWidget_Users->insertRow(ui->tableWidget_Users-
>rowCount());
    ui->tableWidget_Users->setItem(ui->tableWidget_Users-
>rowCount()-1, 0, new QTableWidgetItem(tempList.at(i).fio));

    if(tempList.at(i).role == "Admin")
        ui->tableWidget_Users->setItem(ui->tableWidget_Users-
>rowCount()-1, 1, new QTableWidgetItem("Администратор"));
    else if(tempList.at(i).role == "Oper")
        ui->tableWidget_Users->setItem(ui->tableWidget_Users-
>rowCount()-1, 1, new QTableWidgetItem("Оператор"));
    else if(tempList.at(i).role == "Exp")
        ui->tableWidget_Users->setItem(ui->tableWidget_Users-
>rowCount()-1, 1, new QTableWidgetItem("Эксперт"));
}

if(db_buffer.FindDatabaseName(ui->lineEdit_NameDB->text()))
{
    if(ui->lineEdit_NameDB->text().size() != 0)
        ui->label_WarningsNameDB->setText("База данных с таким
названием уже существует!");
    else
        ui->label_WarningsNameDB->setText("Название не введе-
но!");

    ui->pushButton_AddNewDB_Next_1->setEnabled(false);
}

```



```

else
{
    ui->label_WarningsNameDB->setText("Название базы данных
введено верно!");

    if(!ui->plainTextEdit_InformationDB->toPlainText().size())
        ui->label_WarningsMoreInformation->setText("Дополни-
тельная информация не введена.\nРекомендуется внести данные, кото-
рые помогут пользователю работать с ней.");
    else
        ui->label_WarningsMoreInformation->setText("Дополни-
тельная информация введена!");

    if(!ui->tableWidget_Users->rowCount())
        ui->label_WarningsUsers->setText("Не назначены пользо-
ватели, которые могут редактировать БД.\nРекомендуется сразу вне-
сти эту информацию, чтобы пользователи могли сразу приступить к
работе.");
    else
        ui->label_WarningsUsers->setText("Редакторы назна-
чены!");

    ui->pushButton_AddNewDB_Next_1->setEnabled(true);
}
}

```

```

void AdminMain::on_plainTextEdit_InformationDB_textChanged()
{
    if(db_buffer.FindDatabaseName(ui->lineEdit_NameDB->text()))
    {
        if(ui->lineEdit_NameDB->text().size() != 0)
            ui->label_WarningsNameDB->setText("База данных с таким
названием уже существует!");
        else
            ui->label_WarningsNameDB->setText("Название не введе-
но!");

        ui->pushButton_AddNewDB_Next_1->setEnabled(false);
    }
    else
    {
        ui->label_WarningsNameDB->setText("Название базы данных
введено верно!");

        if(!ui->plainTextEdit_InformationDB->toPlainText().size())
            ui->label_WarningsMoreInformation->setText("Дополни-
тельная информация не введена.\nРекомендуется внести данные, кото-
рые помогут пользователю работать с ней.");
        else
            ui->label_WarningsMoreInformation->setText("Дополни-
тельная информация введена!");

        if(!ui->tableWidget_Users->rowCount())

```

```

        ui->label_WarningsUsers->setText("Не назначены пользо-
ватели, которые могут редактировать БД.\nРекомендуется сразу вне-
сти эту информацию, чтобы пользователи могли сразу приступить к
работе.");
        else
            ui->label_WarningsUsers->setText("Редакторы назна-
чены!");
    }

    ui->pushButton_AddNewDB_Next_1->setEnabled(true);
}

void AdminMain::addDropButton_DB(QString name)
{
    dropButtonDB *tmpButton = new dropButtonDB;
    tmpButton->setInfo(name);
    layoutDB->removeItem(my_spacer);
    layoutDB->addWidget(tmpButton);
    layoutDB->addItem(my_spacer);

    connect(tmpButton, SIGNAL(showDetails(QString)), this,
    SLOT(slot_showDetails_DB(QString)));
    //connect(tmpButton, SIGNAL(deleteThis(QString)), this,
    SLOT(slot_deletedB(QString)));
}

void AdminMain::clear_viewdb()
{
    dropButtonDB *button;
    int i = 0;

    while(i < layoutDB->count())
    {
        button = (dropButtonDB*)layoutDB->takeAt(0)->widget();
        layoutDB->removeWidget(button);
        delete button;
    }
}

void AdminMain::slot_showDetails_DB(QString nameDB)
{
    db_buffer.DownloadUsers();
    db_buffer.DownloadDatabase();
    db_buffer.DownloadUsers_DB();

    DatabaseInfo DBInfo;
    QList <User> Users;

    for(int i = 0; i < db_buffer.getDBList().size(); i++)
    {
        if(db_buffer.getDBList().at(i).name == nameDB)
        {
            DBInfo = db_buffer.getDBList().at(i);

```

```

        i = db_buffer.getDBList().size();
    }
}

for(int i = 0; i < db_buffer.getUserDatabaseList().size(); i++)
{
    if(db_buffer.getUserDatabaseList().at(i).database_name ==
nameDB)
    {
        for(int x = 0; x < db_buffer.getUsersList().size(); x++)
        {
            if(db_buffer.getUserDatabaseList().at(i).user_name
== db_buffer.getUsersList().at(x).login)
            {

Users.push_back(db_buffer.getUsersList().at(x));
                x = db_buffer.getUsersList().size();
            }
        }
    }

    InformationDB_Dialog *InfoDB_Dialog = new
InformationDB_Dialog;
    InfoDB_Dialog->takeInfo(db_buffer.getUsersList(), Users,
DBInfo);
    InfoDB_Dialog->exec();
    delete InfoDB_Dialog;
}

void AdminMain::on_lineEdit_FIO_editingFinished()
{
    if(ui->lineEdit_FIO->text().size() != 0)
    {
        ui->label_MessageforFIO->setText("Введено!");
    }
    else
    {
        ui->label_MessageforFIO->setText("Обязательное поле для
ввода!");
    }

    if(ui->lineEdit_UserLogin->text().size() != 0)
    {
        if(db_buffer.FindUserLogin(ui->lineEdit_UserLogin-
>text()))
            ui->label_MessageforLogin->setText("Такой логин уже
существует!");
        else

```

```

        ui->label_MessageforLogin->setText("Обязательно поле для
ввода!");

        if(!ui->radioButton_Admin->isChecked() && !ui-
>radioButton_Expert->isChecked() && !ui->radioButton_Operator-
>isChecked())
            ui->label_MessageforRole->setText("Обязательное поле!");
        else
            ui->label_MessageforRole->setText("");

        if(ui->lineEdit_UserPassword_1->text().size() == 0 && ui-
>lineEdit_UserPassword_2->text().size() == 0)
            ui->label_MessageforPassword->setText("Обязательное поле
для ввода!");
        else if(ui->lineEdit_UserPassword_1->text().size() != 0 && ui-
>lineEdit_UserPassword_2->text().size() == 0)
            ui->label_MessageforPassword->setText("Повторите
пароль.");
        else if(ui->lineEdit_UserPassword_1->text().size() != 0 && ui-
>lineEdit_UserPassword_2->text().size() != 0 && ui-
>lineEdit_UserPassword_1->text() == ui->lineEdit_UserPassword_2-
>text())
            ui->label_MessageforPassword->setText("Пароль введен!");
        else
            ui->label_MessageforPassword->setText("Пароли не совпа-
дают!");

        if(ui->listWidget_AddedDB->count() == 0)
            ui->label_MessageListDB->setText("Рекосендется сразу доба-
вить доступные для редактирования базы данных, чтобы пользователь
мог сруз приступить к работе. Этот параметр можно будет настроить
и позже.");
        else
            ui->label_MessageListDB->setText("Пользователи
добавлены!");
    }

void AdminMain::on_radioButton_Admin_clicked()
{
    if(ui->lineEdit_FIO->text().size() != 0)
    {
        ui->label_MessageforFIO->setText("Введено!");
    }
    else
    {
        ui->label_MessageforFIO->setText("Обязательное поле для
ввода!");
    }

    if(ui->lineEdit_UserLogin->text().size() != 0)
    {
        if(db_buffer.FindUserLogin(ui->lineEdit_UserLogin-
>text()))

```

```

        ui->label_MessageforLogin->setText("Такой логин уже
существует!");
    }
    else
        ui->label_MessageforLogin->setText("Обязательно поле для
ввода!");

    if(!ui->radioButton_Admin->isChecked() && !ui-
>radioButton_Expert->isChecked() && !ui->radioButton_Operator-
>isChecked())
        ui->label_MessageforRole->setText("Обязательное поле!");
    else
        ui->label_MessageforRole->setText("");

    if(ui->lineEdit_UserPassword_1->text().size() == 0 && ui-
>lineEdit_UserPassword_2->text().size() == 0)
        ui->label_MessageforPassword->setText("Обязательное поле
для ввода!");
    else if(ui->lineEdit_UserPassword_1->text().size() != 0 && ui-
>lineEdit_UserPassword_2->text().size() == 0)
        ui->label_MessageforPassword->setText("Повторите
пароль.");
    else if(ui->lineEdit_UserPassword_1->text().size() != 0 && ui-
>lineEdit_UserPassword_2->text().size() != 0 && ui-
>lineEdit_UserPassword_1->text() == ui->lineEdit_UserPassword_2-
>text())
        ui->label_MessageforPassword->setText("Пароль введен!");
    else
        ui->label_MessageforPassword->setText("Пароли не совпа-
дают!");

    if(ui->listWidget_AddedDB->count() == 0)
        ui->label_MessageListDB->setText("Рекомендется сразу доба-
вить доступные для редактирования базы данных, чтобы пользователь
мог сразу приступить к работе. Этот параметр можно будет настроить
и позже.");
    else
        ui->label_MessageListDB->setText("Пользователи
добавлены!");
}

void AdminMain::on_radioButton_Operator_clicked()
{
    if(ui->lineEdit_FIO->text().size() != 0)
    {
        ui->label_MessageforFIO->setText("Введено!");
    }
    else
    {
        ui->label_MessageforFIO->setText("Обязательное поле для
ввода!");
    }
}

```

```

        if(ui->lineEdit_UserLogin->text().size() != 0)
        {
            if(db_buffer.FindUserLogin(ui->lineEdit_UserLogin-
>text()))
                ui->label_MessageforLogin->setText("Такой логин уже
существует!");
            }
            else
                ui->label_MessageforLogin->setText("Обязательно поле для
ввода!");

            if(!ui->radioButton_Admin->isChecked() && !ui-
>radioButton_Expert->isChecked() && !ui->radioButton_Operator-
>isChecked())
                ui->label_MessageforRole->setText("Обязательное поле!");
            else
                ui->label_MessageforRole->setText("");

            if(ui->lineEdit_UserPassword_1->text().size() == 0 && ui-
>lineEdit_UserPassword_2->text().size() == 0)
                ui->label_MessageforPassword->setText("Обязательное поле
для ввода!");
            else if(ui->lineEdit_UserPassword_1->text().size() != 0 && ui-
>lineEdit_UserPassword_2->text().size() == 0)
                ui->label_MessageforPassword->setText("Повторите
пароль.");
            else if(ui->lineEdit_UserPassword_1->text().size() != 0 && ui-
>lineEdit_UserPassword_2->text().size() != 0 && ui-
>lineEdit_UserPassword_1->text() == ui->lineEdit_UserPassword_2-
>text())
                ui->label_MessageforPassword->setText("Пароль введен!");
            else
                ui->label_MessageforPassword->setText("Пароли не совпа-
дают!");

            if(ui->listWidget_AddedDB->count() == 0)
                ui->label_MessageListDB->setText("Рекосендется сразу доба-
вить доступные для редактирования базы данных, чтобы пользователь
мог сразу приступить к работе. Этот параметр можно будет настроить
и позже.");
            else
                ui->label_MessageListDB->setText("Пользователи
добавлены!");
        }

void AdminMain::on_radioButton_Expert_clicked()
{
    if(ui->lineEdit_FIO->text().size() != 0)
    {
        ui->label_MessageforFIO->setText("Введено!");
    }
    else
    {

```

```

        ui->label_MessageforFIO->setText("Обязательное поле для
ввода!");
    }

    if(ui->lineEdit_UserLogin->text().size() != 0)
    {
        if(db_buffer.FindUserLogin(ui->lineEdit_UserLogin-
>text()))
            ui->label_MessageforLogin->setText("Такой логин уже
существует!");
        else
            ui->label_MessageforLogin->setText("Обязательно поле для
ввода!");

        if(!ui->radioButton_Admin->isChecked() && !ui-
>radioButton_Expert->isChecked() && !ui->radioButton_Operator-
>isChecked())
            ui->label_MessageforRole->setText("Обязательное поле!");
        else
            ui->label_MessageforRole->setText("");

        if(ui->lineEdit_UserPassword_1->text().size() == 0 && ui-
>lineEdit_UserPassword_2->text().size() == 0)
            ui->label_MessageforPassword->setText("Обязательное поле
для ввода!");
        else if(ui->lineEdit_UserPassword_1->text().size() != 0 && ui-
>lineEdit_UserPassword_2->text().size() == 0)
            ui->label_MessageforPassword->setText("Повторите
пароль.");
        else if(ui->lineEdit_UserPassword_1->text().size() != 0 && ui-
>lineEdit_UserPassword_2->text().size() != 0 && ui-
>lineEdit_UserPassword_1->text() == ui->lineEdit_UserPassword_2-
>text())
            ui->label_MessageforPassword->setText("Пароль введен!");
        else
            ui->label_MessageforPassword->setText("Пароли не совпа-
дают!");

        if(ui->listWidget_AddedDB->count() == 0)
            ui->label_MessageListDB->setText("Рекомендется сразу доба-
вить доступные для редактирования базы данных, чтобы пользователь
мог сразу приступить к работе. Этот параметр можно будет настроить
и позже.");
        else
            ui->label_MessageListDB->setText("Пользователи
добавлены!");
    }

void AdminMain::on_lineEdit_UserLogin_editingFinished()
{
    if(ui->lineEdit_FIO->text().size() != 0)
    {

```

```

        ui->label_MessageforFIO->setText("Введено!");
    }
    else
    {
        ui->label_MessageforFIO->setText("Обязательное поле для
ввода!");
    }

    if(ui->lineEdit_UserLogin->text().size() != 0)
    {
        if(db_buffer.FindUserLogin(ui->lineEdit_UserLogin-
>text()))
            ui->label_MessageforLogin->setText("Такой логин уже
существует!");
        }
        else
            ui->label_MessageforLogin->setText("Обязательно поле для
ввода!");

        if(!ui->radioButton_Admin->isChecked() && !ui-
>radioButton_Expert->isChecked() && !ui->radioButton_Operator-
>isChecked())
            ui->label_MessageforRole->setText("Обязательное поле!");
        else
            ui->label_MessageforRole->setText("");

        if(ui->lineEdit_UserPassword_1->text().size() == 0 && ui-
>lineEdit_UserPassword_2->text().size() == 0)
            ui->label_MessageforPassword->setText("Обязательное поле
для ввода!");
        else if(ui->lineEdit_UserPassword_1->text().size() != 0 && ui-
>lineEdit_UserPassword_2->text().size() == 0)
            ui->label_MessageforPassword->setText("Повторите
пароль.");
        else if(ui->lineEdit_UserPassword_1->text().size() != 0 && ui-
>lineEdit_UserPassword_2->text().size() != 0 && ui-
>lineEdit_UserPassword_1->text() == ui->lineEdit_UserPassword_2-
>text())
            ui->label_MessageforPassword->setText("Пароль введен!");
        else
            ui->label_MessageforPassword->setText("Пароли не совпа-
дают!");

        if(ui->listWidget_AddedDB->count() == 0)
            ui->label_MessageListDB->setText("Рекосендется сразу доба-
вить доступные для редактирования базы данных, чтобы пользователь
мог сразу приступить к работе. Этот параметр можно будет настроить
и позже.");
        else
            ui->label_MessageListDB->setText("Пользователи
добавлены!");
    }
}

```



```

void AdminMain::on_lineEdit_UserPassword_1_editingFinished()
{
    if(ui->lineEdit_FIO->text().size() != 0)
    {
        ui->label_MessageforFIO->setText("Введено!");
    }
    else
    {
        ui->label_MessageforFIO->setText("Обязательное поле для
ввода!");
    }

    if(ui->lineEdit_UserLogin->text().size() != 0)
    {
        if(db_buffer.FindUserLogin(ui->lineEdit_UserLogin-
>text()))
            ui->label_MessageforLogin->setText("Такой логин уже
существует!");
        else
            ui->label_MessageforLogin->setText("Обязательно поле для
вввода!");
    }

    if(!ui->radioButton_Admin->isChecked() && !ui-
>radioButton_Expert->isChecked() && !ui->radioButton_Operator-
>isChecked())
        ui->label_MessageforRole->setText("Обязательное поле!");
    else
        ui->label_MessageforRole->setText("");

    if(ui->lineEdit_UserPassword_1->text().size() == 0 && ui-
>lineEdit_UserPassword_2->text().size() == 0)
        ui->label_MessageforPassword->setText("Обязательное поле
для ввода!");
    else if(ui->lineEdit_UserPassword_1->text().size() != 0 && ui-
>lineEdit_UserPassword_2->text().size() == 0)
        ui->label_MessageforPassword->setText("Повторите
пароль.");
    else if(ui->lineEdit_UserPassword_1->text().size() != 0 && ui-
>lineEdit_UserPassword_2->text().size() != 0 && ui-
>lineEdit_UserPassword_1->text() == ui->lineEdit_UserPassword_2-
>text())
        ui->label_MessageforPassword->setText("Пароль введен!");
    else
        ui->label_MessageforPassword->setText("Пароли не совпа-
дают!");

    if(ui->listWidget_AddedDB->count() == 0)
        ui->label_MessageListDB->setText("Рекомендется сразу доба-
вить доступные для редактирования базы данных, чтобы пользователь
мог сразу приступить к работе. Этот параметр можно будет настроить
и позже.");
    else

```

```

        ui->label_MessageListDB->setText("Пользователи
добавлены!");
    }

void AdminMain::on_lineEdit_UserPassword_2_editingFinished()
{
    if(ui->lineEdit_FIO->text().size() != 0)
    {
        ui->label_MessageforFIO->setText("Введено!");
    }
    else
    {
        ui->label_MessageforFIO->setText("Обязательное поле для
ввода!");
    }

    if(ui->lineEdit_UserLogin->text().size() != 0)
    {
        if(db_buffer.FindUserLogin(ui->lineEdit_UserLogin-
>text()))
            ui->label_MessageforLogin->setText("Такой логин уже
существует!");
        else
            ui->label_MessageforLogin->setText("Обязательно поле для
вввода!");

        if(!ui->radioButton_Admin->isChecked() && !ui-
>radioButton_Expert->isChecked() && !ui->radioButton_Operator-
>isChecked())
            ui->label_MessageforRole->setText("Обязательное поле!");
        else
            ui->label_MessageforRole->setText("");

        if(ui->lineEdit_UserPassword_1->text().size() == 0 && ui-
>lineEdit_UserPassword_2->text().size() == 0)
            ui->label_MessageforPassword->setText("Обязательное поле
для ввода!");
        else if(ui->lineEdit_UserPassword_1->text().size() != 0 && ui-
>lineEdit_UserPassword_2->text().size() == 0)
            ui->label_MessageforPassword->setText("Повторите
пароль.");
        else if(ui->lineEdit_UserPassword_1->text().size() != 0 && ui-
>lineEdit_UserPassword_2->text().size() != 0 && ui-
>lineEdit_UserPassword_1->text() == ui->lineEdit_UserPassword_2-
>text())
            ui->label_MessageforPassword->setText("Пароль введён!");
        else
            ui->label_MessageforPassword->setText("Пароли не совпа-
дают!");

        if(ui->listWidget_AddedDB->count() == 0)

```

```

        ui->label_MessageListDB->setText("Рекомендуется сразу доба-
вить доступные для редактирования базы данных, чтобы пользователь
мог сразу приступить к работе. Этот параметр можно будет настроить
и позже.");
    else
        ui->label_MessageListDB->setText("Пользователи
добавлены!");
}

void AdminMain::on_pushButton_ChangeTableDB_clicked()
{
    DatabaseSelectionDialog *DBSelectionDialog = new
DatabaseSelectionDialog;
    DBSelectionDialog-
>setAvailableDB(db_buffer.getDatabaseNameList());
    connect(DBSelectionDialog, SIGNAL(takeResult(QList
<QString>)), this,
SLOT(slot_getInfoFrom_DatabaseSelectionDialog(QList <QString>)));
    DBSelectionDialog->exec();
}

void AdminMain::slot_getInfoFrom_DatabaseSelectionDialog(QList
<QString> resultList)
{
    ui->listWidget_AddedDB->clear();
    ui->listWidget_AddedDB->addItem(resultList);
}

void AdminMain::on_pushButton_SearchDB_clicked()
{
    clear_viewdb();

    if(ui->lineEdit_SearchDB->text().size() != 0)
    {
        for(int i = 0; i < db_buffer.getDBList().size(); i++)
        {
            if(db_buffer.getDBList().at(i).name.indexOf(ui-
>lineEdit_SearchDB->text()) > -1)

addDropButton_DB(db_buffer.getDBList().at(i).name);
        }
    }
    else
    {
        for(int i = 0; i < db_buffer.getDBList().size(); i++)
        {
            addDropButton_DB(db_buffer.getDBList().at(i).name);
        }
    }
}

void AdminMain::clear_viewuser()
{

```

```

dropButtonDB *button;
int i = 0;

while(i < layoutUser->count())
{
    button = (dropButtonDB*)layoutUser->takeAt(0)->widget();
    layoutUser->removeWidget(button);
    delete button;
}

}

void AdminMain::addDropButton_User(QString fio)
{
    //clear_viewuser();

    dropButtonDB *tmpButton = new dropButtonDB;
    tmpButton->setInfo(fio);
    layoutUser->removeItem(my_spacer);
    layoutUser->addWidget(tmpButton);
    layoutUser->addItem(my_spacer);

    connect(tmpButton, SIGNAL(showDetails(QString)), this,
    SLOT(slot_showDetails_User(QString)));
    //connect(tmpButton, SIGNAL(deleteThis(QString)), this,
    SLOT(slot_deleteDB(QString)));
}

void AdminMain::slot_showDetails_User(QString name)
{
    db_buffer.DownloadDatabase();
    db_buffer.DownloadUsers_DB();
    db_buffer.DownloadDatabaseName();

    User UserInfo;
    QList <QString> DBs;

    for(int i = 0; i < db_buffer.getUsersList().size(); i++)
    {
        if(db_buffer.getUsersList().at(i).fio == name)
        {
            UserInfo = db_buffer.getUsersList().at(i);
            i = db_buffer.getUsersList().size();
        }
    }

    for(int i = 0; i < db_buffer.getUserDatabaseList().size(); i+
+)
    {
        if(db_buffer.getUserDatabaseList().at(i).user_name ==
UserInfo.login)
        {

```

```
DBs.push_back(db_buffer.getUserDatabaseList().at(i).database_name)
;
    }
}

    InformationUser_Dialog *InfoUser_Dialog = new
InformationUser_Dialog;
    InfoUser_Dialog->takeInfo(db_buffer.getDatabaseNameList(),
UserInfo, DBs);
    InfoUser_Dialog->exec();
    delete InfoUser_Dialog;
}
```