# РЕФЕРАТ

Пояснювальна записка: ___ с., ___ рис., ___ табл., __ дод., ___ джерел.

Об'єкт розробки: інформаційна система для рахування заробітної плати на підприємстві.

Мета кваліфікаційної роботи: розробка інформаційної системи, що призначена для рішення на підприємстві завдання бухгалтерського обліку та здійснює ведення обліку заробітної плати.

У вступі розглядається аналіз та сучасний стан проблеми, конкретизується мета кваліфікаційної роботи та галузь її застосування, наведено обґрунтування актуальності теми та уточнюється постановка завдання.

У першому розділі проведено аналіз предметної галузі, визначено актуальність завдання та призначення розробки, розроблена постановка завдання, задані вимоги до програмної реалізації, технологій та програмних засобів.

У другому розділі виконано аналіз існуючих рішень, обрано вибір платформи для розробки, виконано проектування і розробка програми, наведено опис алгоритму і структури функціонування програми, визначені вхідні і вихідні дані, наведені характеристики складу параметрів технічних засобів, описаний виклик та завантаження програми, описана робота програми.

В економічному розділі визначено трудомісткість розробленої інформаційної системи, проведений підрахунок вартості роботи по створенню програми та розраховано час на його створення.

Практичне значення роботи полягає в тому, що дана програма спрощує роботу бухгалтера та створює йому комфортні можливості для виконання своїх обов’язків, підвищує ефективність праці робітників та надає доступ до застосування сучасних інформаційних технологій в своїй роботі.

Актуальність теми кваліфікаційної роботи визначається тим, що на даний момент існує мало програмних продуктів, які б були простими у використанні і виконували тільки ті завдання, які стоять перед підприємством. Готові програмні рішення, як правило вирішують значно більший обсяг завдань, частина з яких не потрібна конкретному підприємству. Крім того, подібні програми мають завелику для малого підприємства вартість.

Список ключових слів: КОМП’ЮТЕР, ІНФОРМАЦІЙНА СИСТЕМА, ОБЛІК, АЛГОРИТМ, ПРОЕКТУВАННЯ, МЕНЮ, ВКЛАДКА, ДОДАТОК.

## ABSTRACT

Explanatory note: ___ pp., ___ fig., ___ table, __ appendix, ___ sources.

Object of development: information system for calculating wages at the enterprise

The purpose of the qualification work: the development of an information system that is designed to solve the problem of accounting at the enterprise and maintains payroll accounting.

The introduction considers the analysis and the current state of the problem, specifies the purpose of the qualification work and the field of its application, provides a justification for the relevance of the topic and clarifies the problem.

In the first section the analysis of the subject area is carried out, the urgency of the task and purpose of development is defined, the statement of the task is developed, requirements to software realization, technologies and software are set.

The second section analyzes existing solutions, selects a platform for development, performs design and development of the program, describes the algorithm and structure of the program, determines the input and output data, provides characteristics of the parameters of hardware, describes the call and download of the program, describes the program .

The economic section determines the complexity of the developed information system, calculates the cost of work to create a program and calculates the time for its creation.

The practical significance of the work is that this program simplifies the work of the accountant and creates comfortable opportunities for him to perform his duties, increases the efficiency of workers and provides access to the use of modern information technology in their work.

The relevance of the topic of qualification work is determined by the fact that at the moment there are few software products that would be easy to use and perform only those tasks that face the company. Ready-made software solutions, as a rule, solve a much larger volume of tasks, some of which are not needed by a particular company. In addition, such programs are too expensive for a small business.

List of keywords: COMPUTER, INFORMATION SYSTEM, ACCOUNTING, ALGORITHM, DESIGN, MENU, TAB, APPENDIX.

## ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

АІС – Автоматизована інформаційна система;

АІС-БУ – Автоматизована інформаційна система бухгалтерського обліку;

АСУ – автоматизована система управління;

ІС – інформаційна система;

БД – база даних;

ООП – об’єктно-орієнтоване програмування;

ОС – операційна система;

ПЗ – програмне забезпечення;

ПК – персональний комп'ютер;

СКБД – система керування базою даних.

**ЗМІСТ**

РЕФЕРАТ……..……..... 3
ABSTRACT……..... 4
СПИСОК УМОВНИХ ПОЗНАЧЕНЬ……..... 5
ВСТУП……..... 8
РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАВДАННЯ……..... 9
1.1. Загальні відомості з предметної галузі ……..... 9
1.2. Призначення розробки та галузь застосування…….. 12
1.3. Підстава для розробки……..……..... 13
1.4. Постановка завдання……..… 13
1.5. Вимоги до програми або програмного виробу……..…… 14
1.5.1. Вимоги до функціональних характеристик……..…… 14
1.5.2. Вимоги до інформаційної безпеки…… 16
1.5.3. Вимоги до складу та параметрів технічних засобів…… 16
1.5.4. Вимоги до інформаційної та програмної сумісності ……..…… 16
РОЗДІЛ 2. ПРОЄКТУВАННЯ ТА РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ……..…… 17
2.1. Функціональне призначення системи ……. 17
2.2. Опис застосованих математичних методів……..... 17
2.3. Опис використаних технологій та мов програмування…..…... 21
2.4. Опис структури програми та алгоритмів її функціонування … 27
2.4.1. Архітектура програми..... 27
2.4.2. UML-проєктування..... 31
2.4.3. Опис складових програми..... 35
2.4.4. Схема і опис бази даних..... 38
2.5. Обґрунтування та організація вхідних та вихідних даних програми…….. 44

2.6. Опис розробленої системи ……..... 44
2.6.1. Використані технічні засоби…… 44
2.6.2. Використані програмні засоби…….. 45
2.6.3. Виклик та завантаження програми……... 45
2.6.4. Опис інтерфейсу користувача……..…… 45
РОЗДІЛ 3. ЕКОНОМІЧНИЙ РОЗДІЛ……..... 53
3.1. Розрахунок трудомісткості та вартості розробки програмного продукту…….. 53
3.2. Розрахунок витрат на створення програми……... 56
ВИСНОВКИ…… 58
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ…….. 60
Додаток А. Код програми……. 63
Додаток Б. Відгук керівника економічного розділу……... 85
Додаток В. Перелік файлів на диску…… 86

# ВСТУП

Революційний розвиток комп'ютерної техніки призвело до автоматизації бухгалтерського обліку. Дуже складно доводилося бухгалтерам, які освоювали новий інструмент - комп'ютер. Особливо важко доводилося бухгалтерам старшого покоління, які до сих пір з недовірою ставляться до комп'ютера.

Незважаючи на це, з плином часу, все змінилося і сьогодні комп'ютер або ноутбук - це відмінна заміна застарілих бухгалтерських записів. Саме, тому, бухгалтера протягом швидкого терміну адаптації стали довіряти комп'ютеру найскладніші бухгалтерські обчислення.

Переваги комп'ютеризованої бухгалтерії в тому, що можна легко і просто створити всю бухгалтерську інформацію у вигляді електронної бази даних, а також реалізувати можливість для друку різних документів і форм звітності.

І на сьогодні комп’ютер для бухгалтера являється основним інструментом.

Актуальність теми кваліфікаційної роботи визначається тим, що на даний момент існує мало програмних продуктів, які б були простими у використанні і виконували тільки ті завдання, які стоять перед підприємством. Готові програмні рішення, як правило вирішують значно більший обсяг завдань, частина з яких не потрібна конкретному підприємству. Крім того, подібні програми мають завелику для малого підприємства вартість.

Темою даного проєкту є «Розробка інформаційної системи ведення обліку заробітної плати на підприємстві засобами мови С++».

В даній кваліфікаційній роботі бакалавра засобами середовища qt розроблено інформаційну систему, що призначена для бухгалтерів на підприємстві та здійснює ведення обліку заробітної плати.

Дана програма спрощує роботу бухгалтера та створює йому комфортні можливості для виконання своїх обов’язків, підвищує ефективність праці робітників та надає доступ до застосування сучасних інформаційних технологій в своїй роботі.

# РОЗДІЛ 1
# АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАВДАННЯ

## 1.1. Загальні відомості з предметної галузі

Автоматизована інформаційна система (АІС) - це система, в якій інформаційний процес управління автоматизований за рахунок застосування спеціальних методів обробки даних, що використовують комплекс обчислювальних, комунікаційних та інших технічних засобів, з метою отримання і доставки результатної інформації користувачеві-фахівця для виконання покладених на нього функцій управління.

Рішення в системі управління приймаються людьми на основі інформації, що є продуктом ІС. На її вході знаходиться первинна інформація про всі зміни, що відбуваються в об'єкті управління. Вона фіксується в результаті виконання функцій оперативного обліку. В ІС первинна інформація перетворюється в результатную, придатну для прийняття рішень. В автоматизованих ІС частина процедур формального перетворення первинної інформації в результатную автоматично виконуються технічними засобами по заздалегідь заданими алгоритмами, без безпосереднього втручання людини.

Це не означає, що ІС може повністю функціонувати в автоматичному режимі. Персонал системи управління визначає склад і структуру первинної та результатної інформації, порядок збору та реєстрації первинної інформації, контролює її повноту і достовірність, визначає порядок виконання перетворень первинної інформації в результаті, контролює хід виконання процесу перетворень. До того ж досі слабо автоматизована процедура збору первинної інформації, тому її введення в технічні засоби також здійснюється персоналом ІС.

Найважливішою частиною технічних засобів перетворення інформації є комп'ютери, які здійснюють автоматичний процес обробки даних на основі заздалегідь заданих програм. В сучасних АІС процедури інформаційного

процесу децентралізовані і виконуються в діалоговому режимі роботи користувача з комп'ютером, що дозволяє йому контролювати процес перетворення даних, оперативно направляючи його в потрібне йому русло. Цим вони відрізняються від АІС, що базуються на великих ЕОМ, у яких процес обробки інформації виконувався централізовано і був відділений від управлінського персоналу. Останній отримував лише кінцеві результати обробки даних і, якщо вони його з тих чи інших причин (наприклад, внаслідок пізно виявлених помилок у вихідних даних) не влаштовували, змушений був робити запит відповідним службам на повторення процесу рішення, що цікавить його завдання.

Таким чином, в сучасних АІС автоматично виконуються процедури інформаційного процесу інтегровані з функціями управління. Поряд зі своїми основними функціями, їх безпосередньо виконує управлінський персонал. Більш того, використовуючи інструментальні програмні засоби, орієнтовані на користувача, що не має професійної комп'ютерної підготовки, фахівець-управлінець часто сам може автоматизувати виконання необхідних йому процедур обробки даних, виступаючи і в ролі постановника завдання і програміста.

Відзначимо, що в сучасному понятті термін «інформаційні системи» має на увазі автоматизацію інформаційних процесів. Тому обидва терміни використовуються як рівноправні. Але слід пам'ятати про те, що інформаційні системи можуть використовувати і неавтоматизовану технологію обробки інформації.

Одне з найважливіших місць в інформаційних системах підприємств займає функція бухгалтерського обліку. Для виконання в повному обсязі функцій бухгалтерського обліку в управлінні підприємством і для складання звітності, що надається зовнішнім користувачам, необхідно здійснювати збір, реєстрацію, передачу, накопичення, зберігання і обробку облікових даних. Для реалізації цього інформаційного процесу потрібні відповідні форми організації роботи, технічні засоби, методи і способи перетворення даних, а також

персонал певної кваліфікації. Все це і становить автоматизовану інформаційну систему бухгалтерського обліку, яка є невід'ємною частиною АІС підприємства.

Автоматизована інформаційна система бухгалтерського обліку (АІС-БУ) - це система, в якій інформаційний процес бухгалтерського обліку автоматизовано за рахунок застосування спеціальних методів обробки даних, що використовують комплекс обчислювальних, комунікаційних та інших технічних засобів, з метою отримання і доставки інформації, необхідної фахівцям-бухгалтерам для виконання функцій управлінського і фінансового обліку. У порівнянні з визначенням АІС, тут обмежується предметна область, в якості якої виступає бухгалтерський облік як функція управління підприємством.

АІС-БУ як складова частина АІС містить три основні компоненти:

– інформацію як предмет і продукт праці;

– засоби, методи і способи переробки інформації;

– персонал, який реалізує інформаційний процес обліку, використовуючи наявні засоби обробки інформації.

Окремі завдання бухгалтерського обліку розподіляються по комплексам завдань. Традиційно в інформаційній системі бухгалтерського обліку виділяються комплекси задач обліку: основних засобів, матеріальних цінностей, праці і заробітної плати, готової продукції і її реалізації, фінансово-розрахункових операцій витрат на виробництво, зведеного обліку і складання звітності.

При виділенні одних комплексів приймається до уваги однорідність об'єктів спостереження першого типу, наприклад при виділенні комплексу завдань обліку матеріальних цінностей.

В основі виділення інших комплексів лежать виробничі процеси, зокрема виробничого споживання при виділенні комплексу обліку витрат на виробництво.

У третьому випадку поєднуються обидві ознаки, наприклад при виділенні

комплексу обліку основних засобів: об'єкти спостереження, з одного боку, однорідні, в той же час завдання цього комплексу відображають процеси відтворення основних засобів.

У практиці АІС-БУ часто присутні і інші підходи до виділення комплексів і складу їх завдань. Все залежить від функціональної структури АІС підприємства в цілому, розмежування завдань між інформаційними системами різних служб управління і багатьох інших факторів. Однак в будь-якому випадку інформація про об'єкти спостереження бухгалтерського обліку повинна бути повною і достовірною, а узагальнена результативна інформація - актуальною і достатньою для прийняття управлінських рішенні [21].

В даній кваліфікаційній роботі бакалавра розроблено інформаційну систему, що призначена для рішення на підприємстві завдання бухгалтерського обліку та здійснює ведення обліку заробітної плати.

## 1.2. Призначення розробки та галузь застосування

На сучасному етапі розвитку технічних засобів та ринку програмних продуктів існує досить велика кількість програм бухгалтерського та складського обліку, але і дотепер є необхідність розробки програм такого класу, тому що кожна фірма має свої особливості роботи, які не враховані в уже розробленому програмному забезпеченні.

Метою даної кваліфікаційної роботи бакалавра є розробка інформаційної системи, що призначена для рішення на підприємстві завдання бухгалтерського обліку та здійснює ведення обліку заробітної плати.

Дана програма спрощує роботу бухгалтера та створює йому комфортні можливості для виконання своїх обов’язків, підвищує ефективність праці робітників та надає доступ до застосування сучасних інформаційних технологій в своїй роботі.

Програма призначена для використання на будь-якому підприємстві зі схожими параметрами розрахунку зарплатні.

## 1.3. Підстава для розробки

Підставою для розробки кваліфікаційної роботи бакалавра на тему «Розробка інформаційної системи ведення обліку заробітної плати на підприємстві засобами мови С++ в середовищі IDE Qt Creator» є наказ по Національному технічному університету «Дніпровська політехніка» від __.__. 2021р. № ____-_.

## 1.4. Постановка завдання

Метою даної кваліфікаційної роботи бакалавра є розробка інформаційної системи, що призначена для рішення на підприємстві завдання бухгалтерського обліку та здійснює ведення обліку заробітної плати.

В рамках поставленої мети необхідно вирішити наступні завдання:

– проаналізувати існуючі автоматизовані інформаційні системи ведення бухгалтерського обліку в рамках нарахування зарплат на підприємстві;

– розробити алгоритм роботи ІС;

– розробити структуру бази даних ІС;

– виконати програмну реалізацію розробленої системи.

Для рішення завдання роботи необхідно виконати наступні етапи:

1. Розробити постановку завдання.
2. Побудувати концептуальну модель обраної предметної області, а саме запропонувати список сутностей й список атрибутів, що описують їх.
3. Розробити логічну модель, відповідну концептуальної моделі.
4. Виконати побудову реляційної моделі для проектованої бази даних на основі логічної моделі.
5. Розробити фізичну модель проектованої бази даних.
6. Розробити інтерфейс для роботи з базою даних.
7. Розробити механізми захисту даних від несанкціонованого доступу.

## 1.5. Вимоги до програми або програмного виробу

### 1.5.1. Вимоги до функціональних характеристик

Розроблена інформаційна система повинна відповідати таким вимогам:

- зручний і зрозумілий інтерфейс;
- перевірка на коректність введених даних;
- ведення списку робітників на підприємстві;
- ведення табелю робітника;
- ведення лікарняних, відпускних і відряджень;
- печать документів;
- авторизація працівника за паролем;
- клієнт-серверна архітектура системи.

Програма повинна надавати такі можливості користувачу:

– перегляд даних робітника;

– введення та редагування табелю працівника за місяць;

– швидкий розрахунок зарплатні;

– друк;

– ведення документів;

– редагування даних працівника.

Початкові дані:

1. Робочі місця:
    – посада(текстове значення);
    – тип оплати (текстове значення)
    – оклад (числове значення);
    – кошти за відпрацьовану годину (числове значення);
    – кількість годин в день (числове значення);
    – тариф (числове значення);
    – шкідливі умови праці (числове значення).

2. Робітники:

- табельний номер (числове значення);
- прізвище (текстове значення);
- посада (текстове значення);
- дата прийняття на роботу (дата);
- дата звільнення (дата);
- премія (числове значення);

3. Місяці:
   - рік та місяць (текстове значення );
   - табельний номер (текстове значення);
   - нараховане (числове значення).
4. Табель:
   - рік та місяць (текстове значення );
   - число (числове значення);
   - часи (текстова значення).
5. Документи:
   - табельний номер (числове значення);
   - серія (текстова значення);
   - тип (текстове значення);
   - початок (дата);
   - кінець (дата).
6. Інформація:
   - прізвище (текстове значення);
   - адреса (текстове значення);
   - телефон (цифрове значення);
   - пароль (текстове значення).

### 1.5.2. Вимоги до інформаційної безпеки

Під час проектування даної системи необхідно:

1. Передбачити контроль введеної інформації.
2. Передбачити блокування некоректних дій користувача при роботі з системою.
3. Забезпечити цілісність інформації, що зберігається.
4. Забезпечити захист від несанкціонованого доступу до інформації.

### 1.5.3. Вимоги до складу та параметрів технічних засобів

Рекомендовані вимоги до програмного та апаратного забезпечення хостингу (серверу):

- можливість підтримки однієї бази даних SQL;
- пропускна здатність - 1Гб;
- деякий строк безкоштовного користування;
- операційна система: Windows Server 2012;
- підтримка протоколів: FTP, HTTP;
- підтримувані бази даних: MySQL, SQL Server 2012;
- об'єм жорсткого диску від 1 Гб.

Рекомендованими вимогами до програмного забезпечення клієнта програми є програма для перегляду текстових фалів.

### 1.5.4. Вимоги до інформаційної та програмної сумісності

Проєкт має бути створено за допомогою Microsoft Visual Studio 2017.

ІС має функціонувати у таких операційних системах: Windows 7, Windows 8, Windows 10.

# РОЗДІЛ 2
# ПРОЄКТУВАННЯ ТА РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ

## 2.1. Функціональне призначення системи

Розроблена інформаційна система призначена для виконання наступних функцій:

- авторизація працівника за паролем;
- перегляд даних робітника;
- ведення списку робітників на підприємстві;
- ведення лікарняних, відпускних і відряджень;
- введення та редагування табелю працівника за місяць;
- ведення документів;
- редагування даних працівника;
- швидкий розрахунок зарплатні;
- друк звітів
- перевірка на коректність введених даних..

## 2.2. Опис застосованих математичних методів

Метод вираховування заробітної плати за окладом:

1. Робітники завжди отримують кошти 2 рази в місяць. Перше отримання – аванс. Це 55% відсотків від окладу:

$$CO * 0{,}55 = A, \text{ грн} \quad (2.1)$$

де СО - стартовий оклад,

А - аванс, але робітник може не мати авансу, якщо кількість відпрацьованих робочих днів менше ніж половина норми часу.

2. Перед тим, як починати нараховувати зарплату співробітнику, треба розрахувати оклад в місяці через кількість відпрацьованих годин:

$$(СО:Ч)*Г=О, грн \quad (2.2)$$

де СО - стартовий оклад,

Ч - норма часу,

Г - відпрацьовані години робітником,

О - оклад.

3. В зарплатню також входе і премія. Для всіх робітників премія 5%.

$$О * 0,05 = П, грн \quad (2.3)$$

де О - оклад,

П - премія.

4. Якщо робітник був на лікарняному або у відпустці, то це також є зарплата. Щоб розрахувати суму за лікарняні або відпускні, треба знати суму всього нарахованої зарплатні за попередні 12 місяців.

$$(М : 365) * Л = СЛ, грн \quad (2.4)$$

де М - сума всього нарахованого за попередні 12 місяців,

Л - кількість лікарняних днів,

СЛ - сума коштів за лікарняні.

$$(М : 354) * В = СВ, грн \quad (2.5)$$

де В - кількість відпускних днів,

СВ - сума коштів за відпускні.

5. Нарешті розраховуємо всього нараховану заробітну плату за місяць.

$$О+П+СЛ+СВ=Н, \text{ грн} \quad (2.6)$$

де Н – нараховані кошти за місяць.

6. Визначимо розмір утримань із заробітної плати (податки). Всього податків складає 20,5% від нарахованих коштів за місяць:

- ПДФО 18%;
- військовий збір 1,5%;
- проф. внески 1%.

$$Н - (Н * 0,205) = З, \text{ грн} \quad (2.7)$$

де З - зарплатня.

Метод рахування заробітної плати за тарифом:

1. Тарифники получають аванс - це:

$$166.8*0,55*Т = А \text{ , грн} \quad (2.8)$$

де Т- тариф.

2. Для розрахунку другої заробітної плати, потрібно визначити кількість кожного виду часів:

- р - робочі години(з 7:00 до 19:00);
- вч - вечірні(з 19:00 до 22:00);
- нч - нічні(з 22:00 до 6:00);
- тн - лікарняні;
- в - відпустка ;
- вд - відрядження.

За кількість вечірніх годин працівника отримує 20% до тарифу, а за нічні - 40%. За всі робочі години нараховується 12% за шкідливі праці.

$$К*Т*ПР=С, грн \quad (2.9)$$

де К- кількість годин відпрацьованих годин певного типу,

Т - тариф,

ПР - відповідний процент,

С - сума коштів за певний тип годин.

3. Розрахунок премії: [нараховані за роб. год.] * 0,05 .

4. Для розрахунку лікарняних і відпускних потрібно знати суму нарахованих за попередніх 12 місяців:

$$(М : 365) * Л = СЛ, грн \quad (2.10)$$

де М - сума всього нарахованого за попередні 12 місяців,

Л - кількість лікарняних днів,

СЛ - сума коштів за лікарняні.

$$(М : 354) * В = СВ, грн \quad (2.11)$$

де В - кількість відпускних днів,

СВ - сума коштів за відпускні.

5. Для розрахунку коштів за відрядження потрібно вирахувати суму нарахованих окрім лікарняних і відпускних за попередні 12 місяців:

$$(З : С) * Д = СД, грн \quad (2.12)$$

де З - сума нарахованих крім лікарняних та відпускних за попередні 2 місяця,

С - кошти за відпрацьовані робочі години,

В - кількість годи відрядження,

СД - сума коштів за відрядження.

6. Вираховуємо зарплатню:

$$З+С+П+СЛ+СВ+СД=Н, \text{ грн} \quad (2.13)$$

де С - сума нарахованих за нічні, вечірні та шкідливі умови праці,

Н- нараховані.

7. Визначимо розмір утримань із заробітної плати (податки). Всього податків складає 20,5% від нарахованих коштів за місяць.

$$Н - (Н * 0,205) = З, \text{ грн} \quad (2.14)$$

де З – зарплатня [4].

## 2.3. Опис використаних технологій та мов програмування

Qt Creator вільна IDE для розробки на C, C ++ і QML. Розроблено Trolltech (Digia) для роботи з FreamWork Qt. Включає в себе графічний інтерфейс відладчика і візуальні засоби розробки інтерфейсу.

Qt Creator розроблена норвезькою компанією Trolltech, яку у 2008 році поглинула Nokia. Анонс проєкту відбувся на Qt Developer Days в жовтні 2008 року. Публічна бета-версія проєкту була опублікована 30 жовтня 2008. Фінальний реліз відбувся 3 березня 2009 року.

Після укладення стратегічного союзу з Microsoft Nokia втратила інтерес до розвитку технологій Qt. У березні 2011 фінська компанія Digia, постачальник ERP-систем, послуг і рішень в області мобільних систем і користувацьких інтерфейсів, оголосила про укладення угоди з Nokia про викуп у тої прав на комерційне ліцензування та надання послуг з підтримки розробки

з використанням бібліотеки Qt. У вересні 2012 Nokia повністю відмовилася від Qt і Digia купує у Nokia весь бізнес і програмні технології, пов'язані з Qt.

Основні особливості QT:

– вбудований редактор форм (Qt Designer) і довідкова система (Qt Assistant);

– контекстно-залежна система допомоги;

– розширюваність плагінами;

– є графічний фронтенд для GDB;

– підтримка зневадження за допомогою CDB;

– узагальнене підсвічування синтаксису, підтримується велика кількість мов програмування і розмітки. Є можливість створення своїх стилів підсвічування;

– можливість редагувати етапи складання проєкту;

– підтримка розробки на мовах C/C++, JavaScript, QML.

Основне завдання Qt Creator - спростити розробку програми за допомогою фреймворка Qt на різних платформах. Тому серед можливостей, властивих будь-якому середовищі розробки, є і специфічні, такі як налагодження додатків на QML і відображення у відладчику даних з контейнерів Qt, вбудований дизайнер інтерфейсів. Для виділення певних частин інформації, що виводиться на екран монітора, використовувати різні кольори.

Програма написана мовою C ++. Вона широко використовується для розробки програмного забезпечення, будучи одним з найпопулярніших мов програмування. Область його застосування включає створення операційних систем, різноманітних прикладних програм, драйверів пристроїв, додатків для вбудованих систем, високопродуктивних серверів, а також ігор. Існує безліч реалізацій мови C ++, як безкоштовних, так і комерційних і для різних платформ. Наприклад, на платформі x86 це GCC, Visual C ++, Intel C ++ Compiler, Embarcadero (Borland) C ++ Builder і інші. C ++ зробив величезний вплив на інші мови програмування, в першу чергу на Java і C #.

Мова виникла на початку 1980-х років, коли співробітник фірми Bell Labs Бйорн Страуструп придумав ряд удосконалень до мови С під власні потреби. Коли в кінці 1970-х років Страуструп почав працювати в Bell Labs над завданнями теорії черг (в додатку до моделювання телефонних викликів), він виявив, що спроби застосування існуючих в той час мов моделювання виявляються неефективними, а застосування високоефективних машинних мов занадто складно через їх обмеженою виразності.

Назва «Сі++» була вигадана Ріком Масситті (Rick Mascitti) і вперше було використана в грудні 1983 року. Раніше, на етапі розробки, нова мова називалася «Сі з класами». Ім'я, що вийшло у результаті, походить від оператора Сі «++» (збільшення значення змінної на одиницю) і поширеному способу присвоєння нових імен комп'ютерним програмам, що полягає в додаванні до імені символу «+» для позначення поліпшень.

Стандарт С ++ складається з двох основних частин: опис ядра мови і опис стандартної бібліотеки.

Розвиток мови супроводив розвиток крос-компілятора cfront. Нововведення в мові відбивалися в зміні номера версії крос-компілятора. Ці номери версій крос-компілятора розповсюджувалися і на саму мову, але стосовно до теперішнього часу мова про версії мови С++ не ведуть. Лише в 1998 році мова стала стандартизованим.

Стандартна бібліотека С++ включає стандартну бібліотеку Сі з невеликими змінами, які роблять її більш відповідною для мови С ++. Інша велика частина бібліотеки С++ заснована на Стандартній Бібліотеці Шаблонів (STL). Вона надає такі важливі інструменти, як контейнери (наприклад, вектори і списки) і ітератори (узагальнені вказівники), що надають доступ до цих контейнерів як до масивів. Крім того, STL дозволяє схожим чином працювати і з іншими типами контейнерів, наприклад, асоціативними списками, стеками, чергами.

Доступ до можливостей стандартної бібліотеки С++ забезпечується за допомогою включення в програму (за допомогою директиви #include)

відповідних стандартних заголовків файлів. Всього в стандарті C++ 11 визначено 79 таких файлів. Засоби стандартної бібліотеки оголошуються які входять в простір імен std. Заголовки, імена яких відповідають шаблону «X», де X - ім'я заголовки стандартної бібліотеки C без розширення (cstdlib, cstring, cstdio та ін.). Містять оголошення, відповідні даної частини стандартної бібліотеки C. Стандартні функції бібліотеки C також знаходяться в просторі назв std.

C ++ містить засоби розробки програм контрольованої ефективності для широкого спектра задач, від низькорівневих утиліт і драйверів до вельми складних програмних комплексів. Зокрема висока сумісність з мовою Сі: код на Сі може бути з мінімальними переробками скомпільована компілятором C ++. Візуально-мовний інтерфейс є прозорим, так що бібліотеки на Сі можуть викликатися з C++ без додаткових витрат, і більш того при певних обмеженнях код на C ++ може експортуватися зовні не відрізнятися від коду на Сі.

До числа недоліків мови можна віднести:

– відсутність системи модулів C++ успадкував від Сі підключення заголовних файлів за допомогою препроцесора. Це змушує дублювати опису об'єктів, породжує неочевидні вимоги до коду і збільшує обсяг компілюємого тексту, а значить і час компіляції;

– успадковані від Сі небезпечні і провокують помилки можливості (макроозначення, адресна арифметика, неявне приведення типів, можливість прямого управління розподілом пам'яті).

– відсутність вбудованих механізмів статичної валідації часу життя об'єктів, що приводить до раптового краху програм через звернення до знищеної змінної, або через неправильну багатопотокової роботи з об'єктами.

– шаблони породжують об'ємний і не завжди оптимальний код. Часткове визначення шаблонів ускладнює як сама мова, так і програми, де воно використовується.

Єдиним прямим нащадком C ++ є мовою D, задуманий як переробка C ++

для усунення найбільш очевидних його проблем. Автори відмовилися від сумісності з Сі, зберігши синтаксис і багато базові принципи C ++ і ввівши в мову можливості, характерні для нових мов. У D немає препроцесора, заголовків файлів, множинного спадкоємства, але є система модулів, інтерфейси, асоціативні масиви, підтримка unicode в рядках, прибирання сміття (при збереженні можливості ручного управління пам'яттю) вбудована багатопоточність, висновок типів, явне оголошення чистих функцій і незмінних значень. Використання D вельми обмежена, вважати його реальним конкурентом C ++ можна [17].

Щоб здійснювати збереження і обробку даних в додатку використовується база даних (далі – БД). Вона є невід'ємною частиною додатку.

Сучасні бази даних зберігають великі об'єми інформації, тому обробляти її вручну, послідовно переглядаючи і редагуючи дані в таблицях, стає досить важко. Для підвищення ефективності користування базами даних застосовують запити, які дозволяють виконувати множинну обробку даних, тобто одночасно вводити, редагувати та видаляти велику кількість записів, а також вибирати дані з таблиць.

Бази даних становлять невід'ємну складову діяльності сучасних підприємств та організацій. Невпинне зростання обсягів інформації, що зберігається в базах даних, та розширення кола користувачів спонукають до використання систем керування базами даних (далі - СКБД). Для створення і реалізації бази даних було обрано СКБД Microsoft SQL Server 2017. Microsoft SQL Server – система керування базами даних, розроблена корпорацією Microsoft [20].

Базовий код MS SQL Server (до версії 7.0) ґрунтувався на коді Sybase SQL Server.

Серед нових можливостей і удосконалень Microsoft SQL Server 2017 слід зазначити появу нових типів даних, а саме – для просторових даних, кращу сумісність з застосунками сторонніх розробників, наприклад Oracle, тіснішу

інтеграцію з Office, оптимізовані засоби шифрування даних, засоби управління на основі політик, а також покращені інструменти звітності і аналізу.

Основною використовуваною мовою запитів при роботі з базою даних є мова SQL. Вона є реалізацією стандарту ANSI / ISO по структуруванню мови запитів (SQL) з розширеннями.

SQL (англ. Structured Query Language) – мова структурованих запитів - це універсальна мова для створення, модифікації та керування інформацією, яка входить до складу реляційних баз даних [9].

На сьогоднішній день SQL – це єдиний механізм, який здатний зв’язати прикладне програмне забезпечення та базу даних.

Сама мова запитів володіє кількома видами запитів. Варто відзначити, що будь-який запит SQL має під собою на увазі звернення до бази даних. У зв’язку з цим прийнято виділяти такі види запитів:

– створення, зміна в базі даних нових або вже існуючих об’єктів;

– отримання даних;

– додавання нових даних в таблицю;

– видалення даних;

– звернення до системи керування базами даних.

Розглянемо основні переваги SQL:

– незалежність від існуючої в даній системі СКБД, а саме тексти SQL є універсальними для багатьох СКБД;

– наявність стандартів SQL сприяє «стабілізації» мови;

– декларативність, при роботі з даними, програміст вибирає тільки ту інформацію, яка повинна бути змінена або модифікована.

Перейдемо до недоліків мови SQL:

– складність SQL;

– деяка невідповідність стандартів;

– невідповідність реляційної моделі даних.

## 2.4. Опис структури системи та алгоритмів її функціонування

### 2.4.1. Архітектура програми

Програма складається з файлів специфікації, реалізації та форм, які зображені на рис. 2.1.

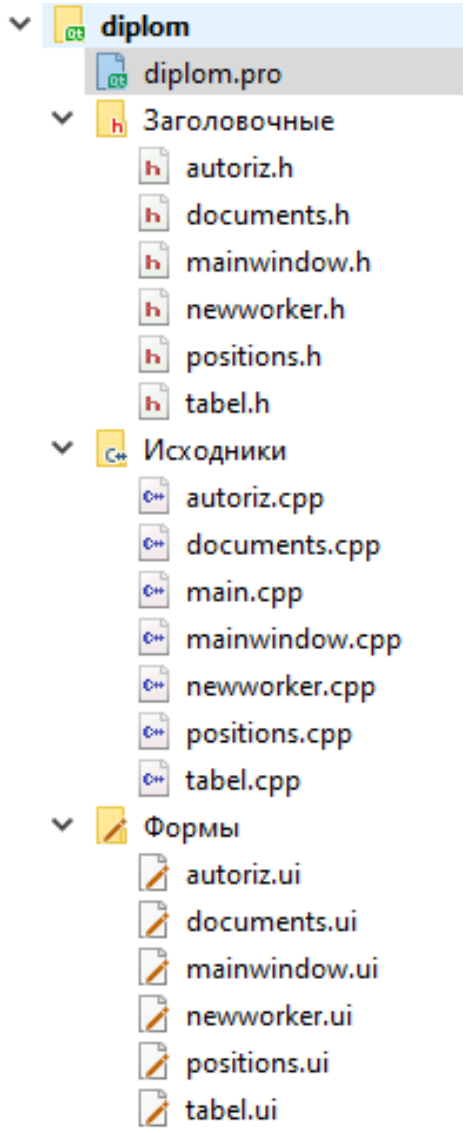



Рис. 2.1. Структура програми

Програма має вікно для входу до головного меню, саме головне вікно та декілька додаткових, які виконують кожну окрему функцію. Опис кожного вікна зображено в таблиці 2.1.

Таблиця 2.1

**Опис модулів**

| Назва | Призначення |
|---|---|
| autoriz<br>(Вхід в програму) | Додатковий модуль, призначений для введення паролю працівника. |
| (mainwindow)<br>Головне меню | Головний модуль програми, який дозволяє переглядати список працівників та їх дані. В ній містяться кнопки та контекстне меню для виклику додаткових вікон. |
| (newworker)<br>Працівник | Додатковий модуль програми, призначений для введення та редагування працівника. |
| (documentu)<br>Документи | Додатковий модуль програми, призначений для додавання та редагування документів на працівника. В даній формі є кнопка для виводу документа у word. |
| (tabel)<br>Табель | Додатковий модуль програми, призначений для ведення табеля працівників. |
| (postions)<br>Посади | Додатковий модуль програми, призначений для додавання та редагування посад на підприємстві |

Схему зв’язку між модулями програми зображено на рис. 2.2.

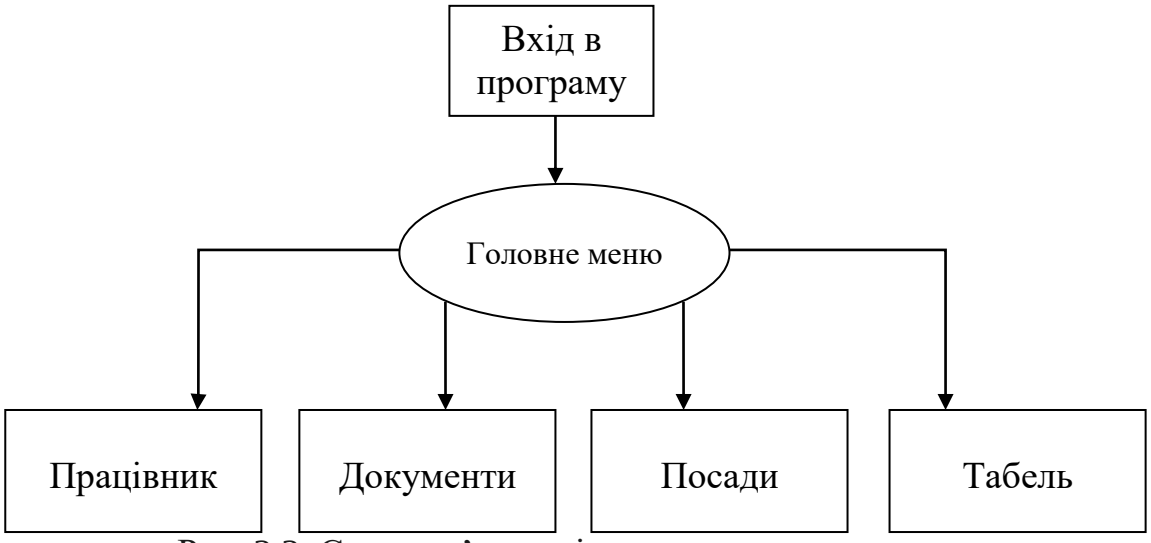



Рис. 2.2. Схема зв’язку між складовими програми

На рис. 2.3 зображена блок-схема алгоритму входу користувача у систему.

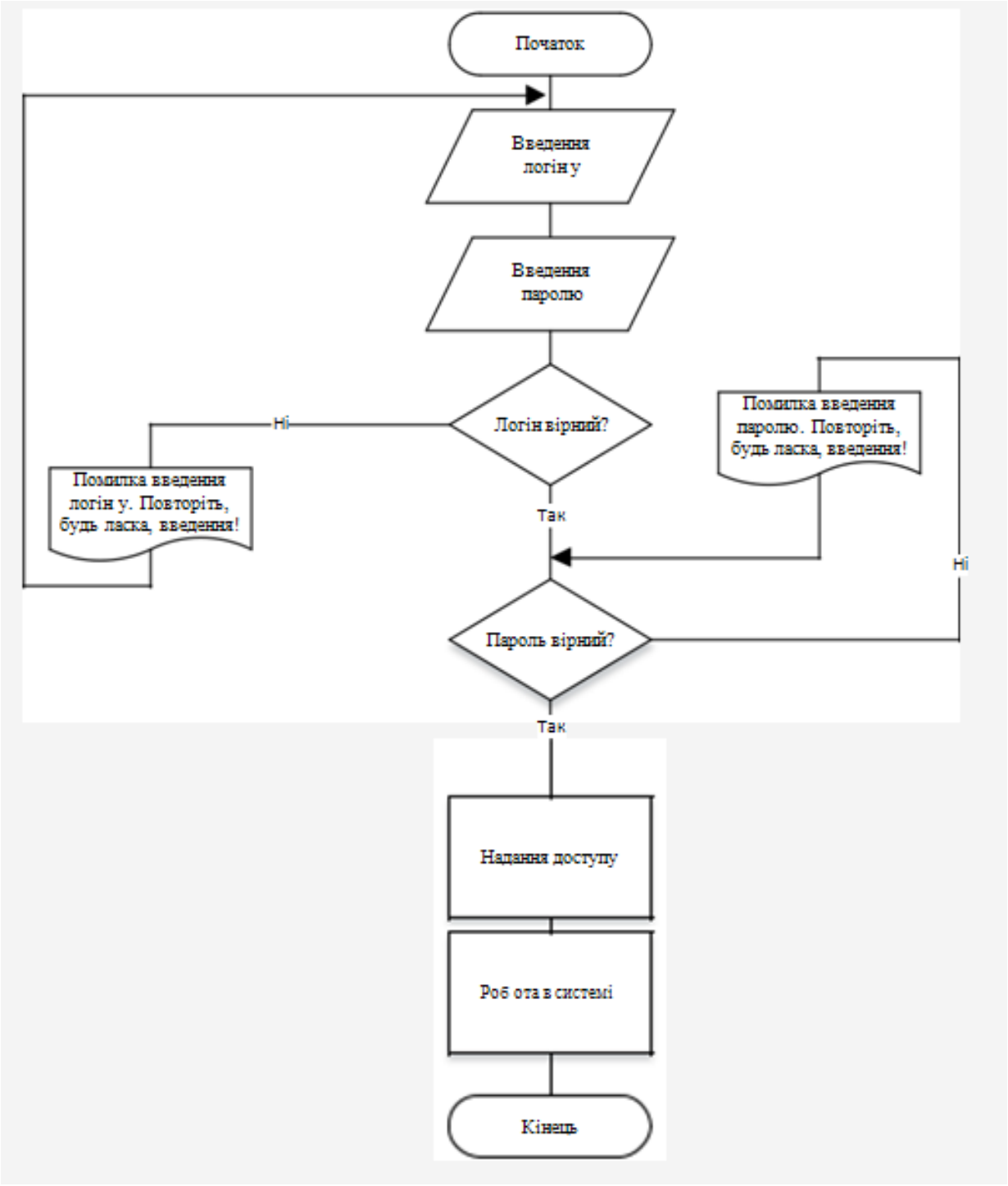



Рис. 2.3. Блок-схема алгоритму входу у систему

На рис. 2.4. зображена блок-схема алгоритму пошуку даних:

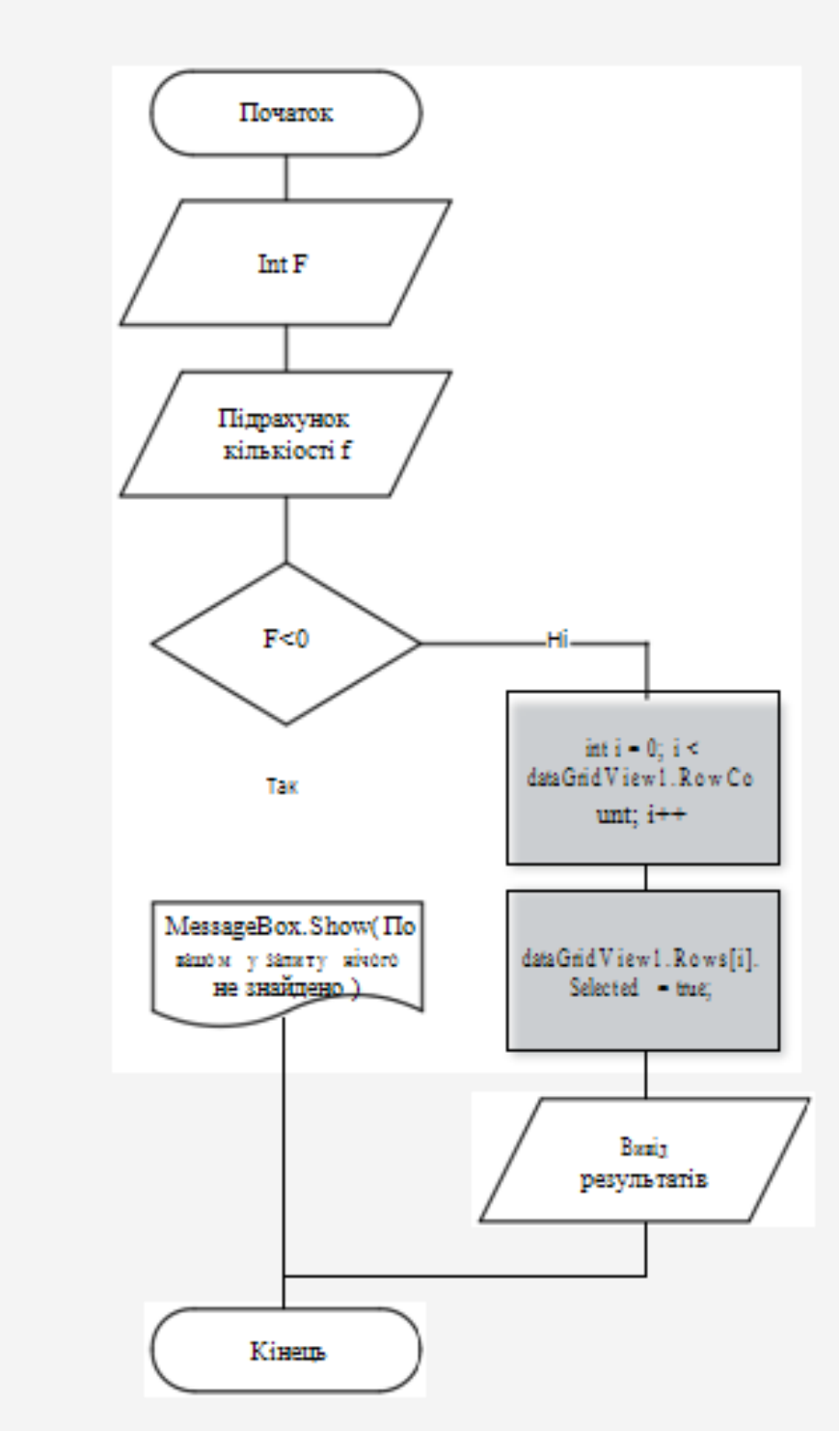



Рис. 2.4. Блок-схема алгоритму пошуку даних

На рис. 2.5. зображена блок-схема алгоритму визначення рівня доступу користувача

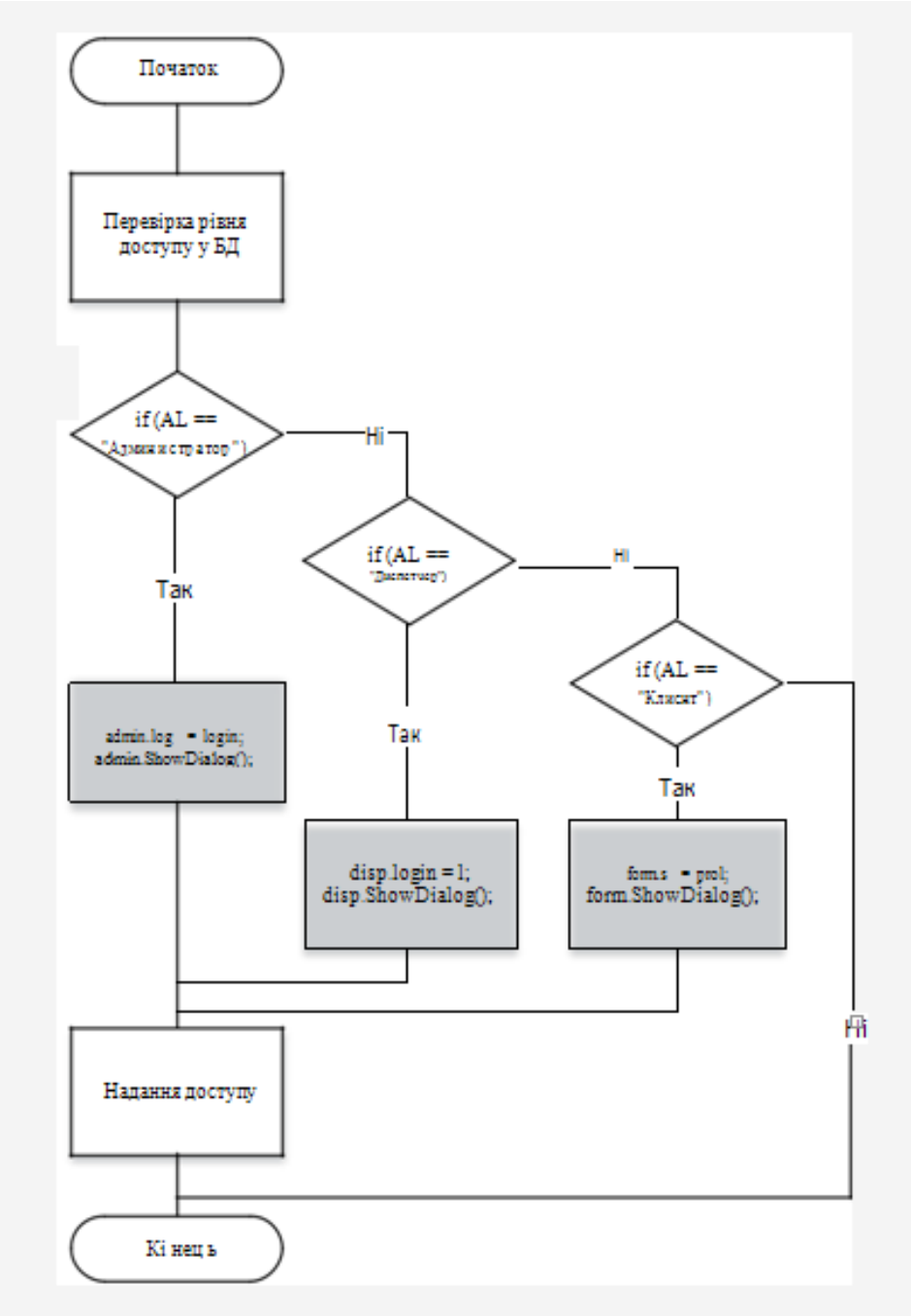



Рис. 2.5. Блок-схема алгоритму визначення рівня доступу користувача

### 2.4.2. UML-проєктування

UML - уніфікована мова моделювання, використовується у парадигмі об'єктно-орієнтованого програмування. Є невід'ємною частиною уніфікованого процесу розробки програмного забезпечення. UML є мовою широкого профілю, це відкритий стандарт, що використовує графічні

позначення для створення абстрактної моделі системи, яка називається UML-моделлю. UML був створений для визначення, візуалізації, проєктування й документування в основному програмних систем. UML не є мовою програмування, але в засобах виконання UML-моделей як інтерпретованого коду можлива кодогенерація [17].

UML може бути застосовано на всіх етапах життєвого циклу аналізу бізнес-систем і розробки прикладних програм. Різні види діаграм які підтримуються UML, і найбагатший набір можливостей представлення певних аспектів системи робить UML універсальним засобом опису як програмних, так і ділових систем.

При проєктуванні програмної системи курсового проєкту було створено 2 UML-діаграми: діаграма сценаріїв, діаграма класів.

Діагрма сценаріїв - це опис поведінки системи, як вона відповідає на зовнішні запити. Іншими словами, різновид використання описує, «хто» і «що» може зробити з розглянутою системою. Методика сценаріїв застосовується для виявлення вимог до поведінки системи, відомих також як функціональні вимоги.

У системному проєктуванні сценарії застосовуються на більш високому рівні ніж при розробці програмного забезпечення, часто представляючи цілі зацікавлених осіб або місії. На стадії аналізу вимог сценарії можуть бути перетворені на ряд детальних вимог і задокументовані за допомогою діаграм вимог SysML або інших подібних механізмів.

Кожен сценарії зосереджується на описі того, як досягти мети або завдання. Для більшості програмних проєктів це означає, що потрібно безліч сценаріїв щоб визначити необхідний набір властивостей нової системи. Ступінь формальності програмного проєкту і його стадії буде впливати на необхідний рівень деталізації, для кожного сценарію діаграми.

Діаграма сценаріїв зображена на рис. 2.6.

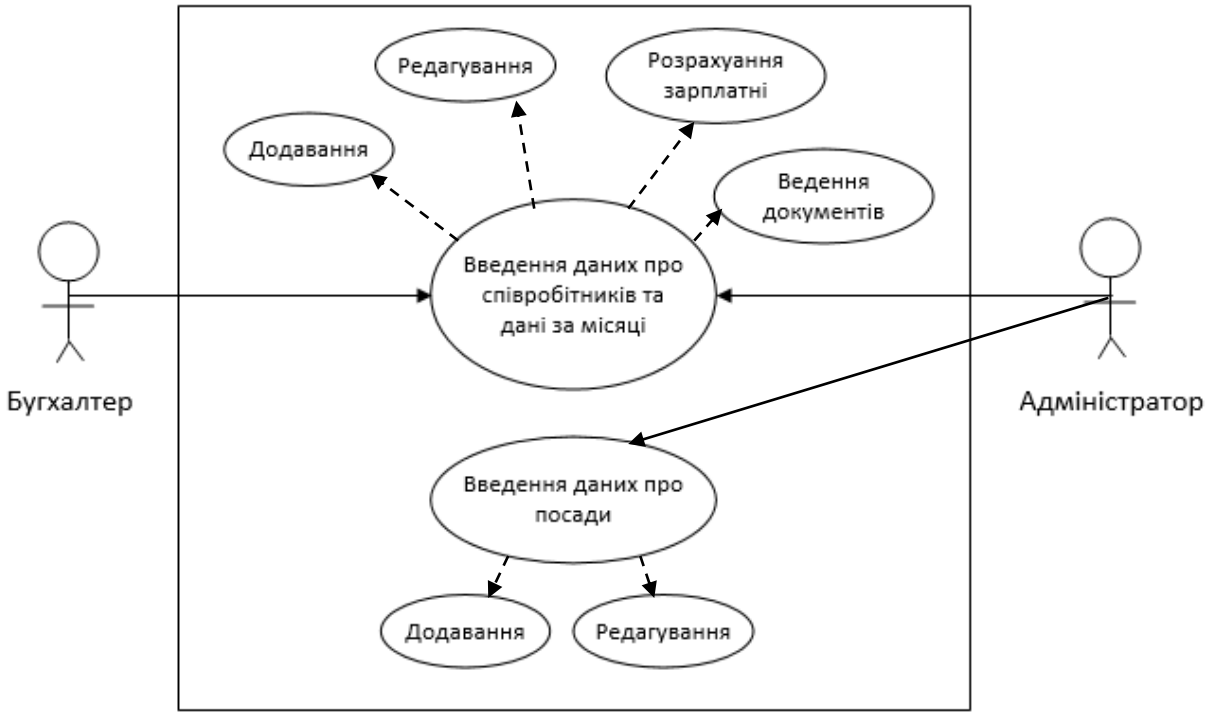



Рис. 2.6. Діаграма сценаріїв

Діаграма класів служить для представлення статичної структури моделі системи в термінології класів об'єктно-орієнтованого програмування. Діаграма класів може зображати, зокрема, різні взаємозв'язки між окремими сутностями предметної області, такими як об'єкти і підсистеми, а також описує їх внутрішню структуру і типи відношень. На даній діаграмі не вказується інформація про часові аспекти функціонування системи. З цієї точки зору діаграма класів є подальшим розвитком концептуальної моделі проєктованої системи.

Діаграма класів представляє собою деякий граф, вершинами якого є елементи типу «класифікатор», які зв'язані різними типами структурних відношень. Варто зауважити, що діаграма класів може також містити інтерфейси, пакети, відношення і навіть окремі екземпляри, такі як об'єкти і зв'язки. Коли говорять про дану діаграму, мають на увазі статичну структурну модель проєктованої системи. Тому діаграму класів прийнято вважати

графічним представленням таких структурних взаємозв'язків логічної моделі системи, які не залежать від часу.

Діаграма класів складається з множини елементів, які в сукупності зображають декларативні знання про предметну область. Ці знання інтерпретуються в базових поняттях мови UML, таких як класи, інтерфейси і відношення між ними і їх складовими компонентами. При цьому окремі компоненти цієї діаграми можуть утворювати пакети для представлення більш загальної моделі системи. Якщо діаграма класів є частиною деякого пакету, то її компоненти повинні відповідати елементам цього пакету [9].

Діаграма класів зображена на рис. 2.7.

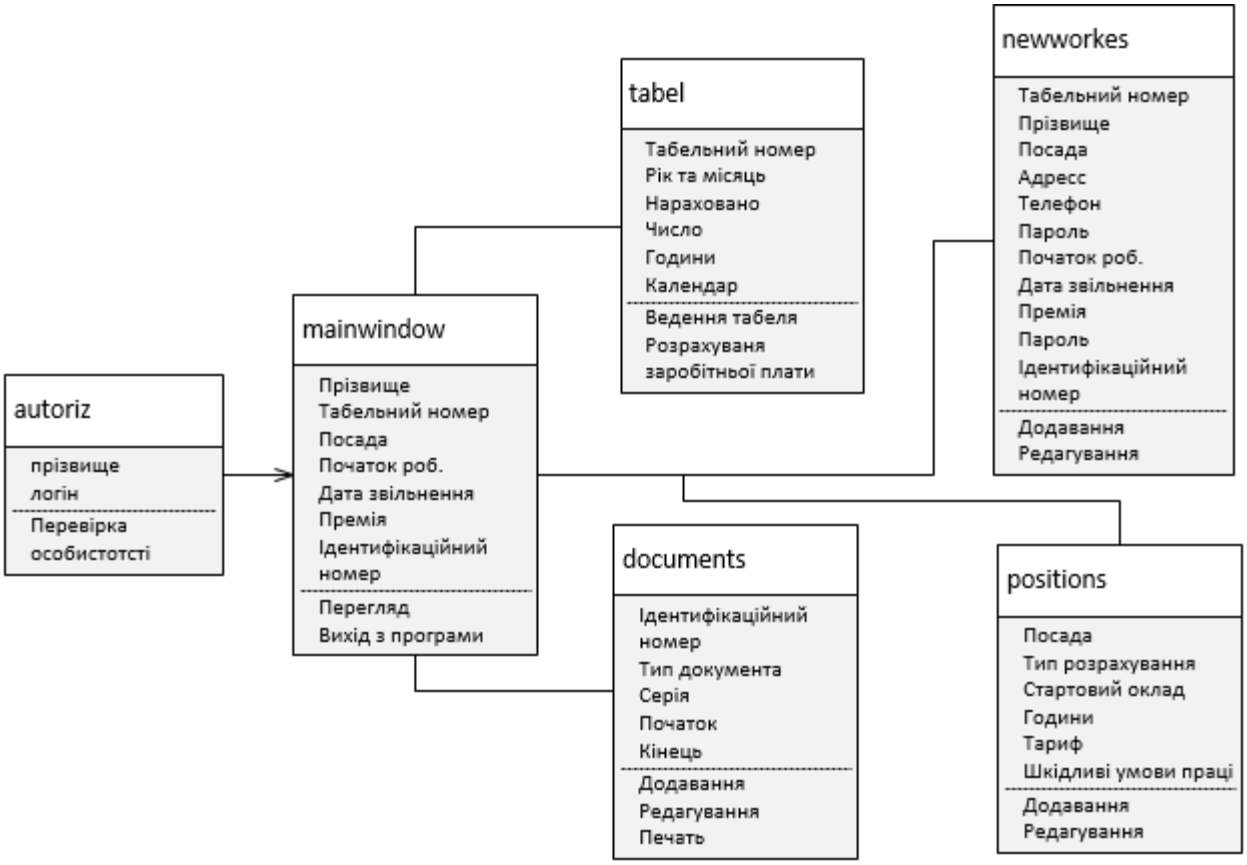



Рис. 2.7. Діаграма класів

### 2.4.3. Опис складових програми

На рис. 2.5 зображено форму autoriz з усіма пронумерованими його компонентами.

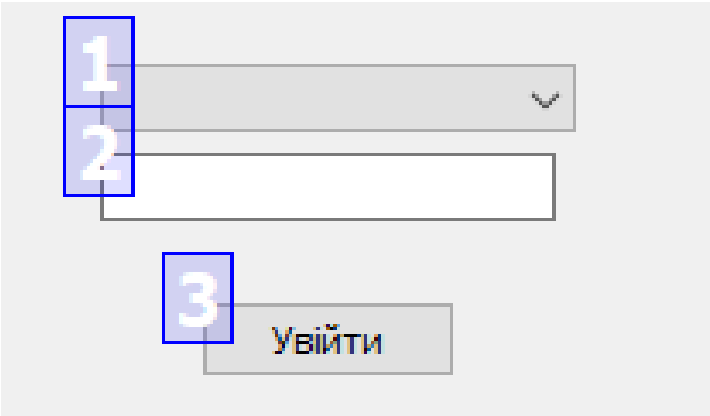



Рис. 2.8. Форма autoriz

Дана форма складається з наступних компонентів:

1. QComboBox - список працівників(окладників), які мають доступ.
2. QLineEdit - введення паролю користувача.
3. QPushButton - кнопка, для перевірки паролю. В разі правильного результату відкриється головне вікно(mainwindow).

На рис. 2.6 зображено форму mainwindow з усіма пронумерованими його компонентами.

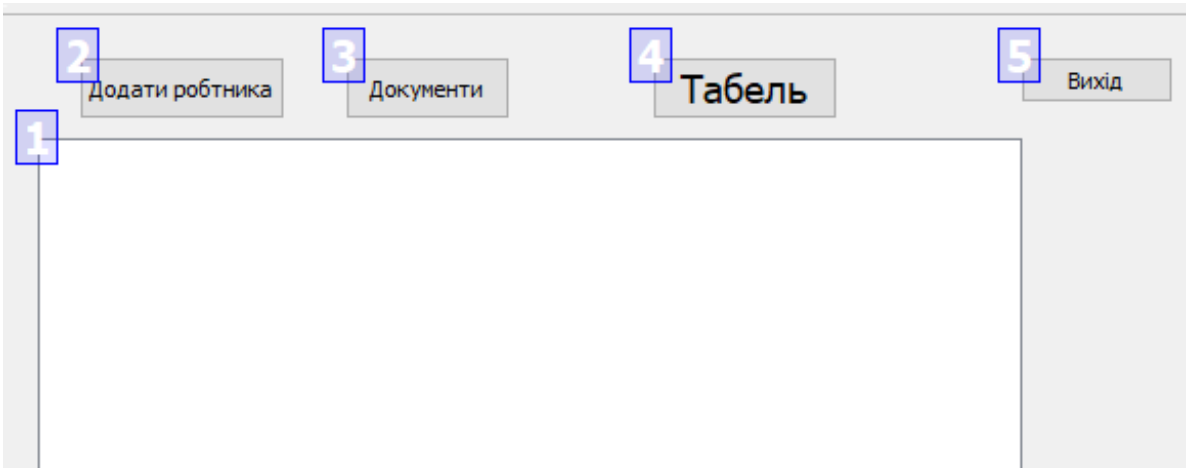



Рис. 2.8. Форма mainwindow

Дана форма складається з наступних компонентів:

1. QTableWidget - таблиця з усіма працівниками.
2. QPushButton - кнопка для переходу на форму додавання нового робітника (newworker).
3. QPushButton - кнопка для переходу на форму ведення документів (documents).
4. QPushButton - кнопка для переходу на форму ведення табеля (tabel).
5. QPushButton - кнопка для виходу з програми.

На рис. 2.9 зображено форму newworker з усіма пронумерованими його компонентами.

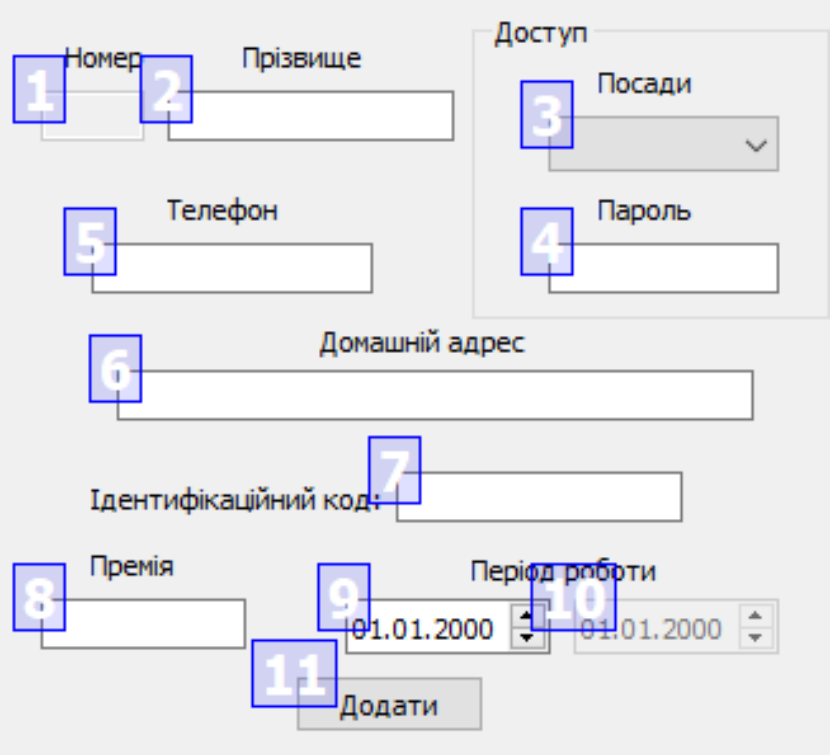



Рис. 2.9. Форма newworker

Дана форма складається з наступних компонентів:

1. QLineEdit - поле для виводу табельного номеру робітника.
2. QLineEdit - поле для введення прізвища.
3. QComboBox - список доступних посад.

4. QLineEdit - в залежності від посади потрібно ввести пароль.
5. QLineEdit - поле для введення номеру телефону.
6. QLineEdit - поле для введення домашнього адреса.
7. QLineEdit - поле для введення ідентифікаційного коду.
8. QLineEdit - поле для введення відсотка премія.
9. QDateEdit - поле для введення дати початку роботи працівника.
10. QDateEdit - поле для виведення дати звільнення.
11. QPushButton - кнопка для додавання даних робітника в базу даних.

На рис. 2.10 зображено форму documents з усіма пронумерованими його компонентами.

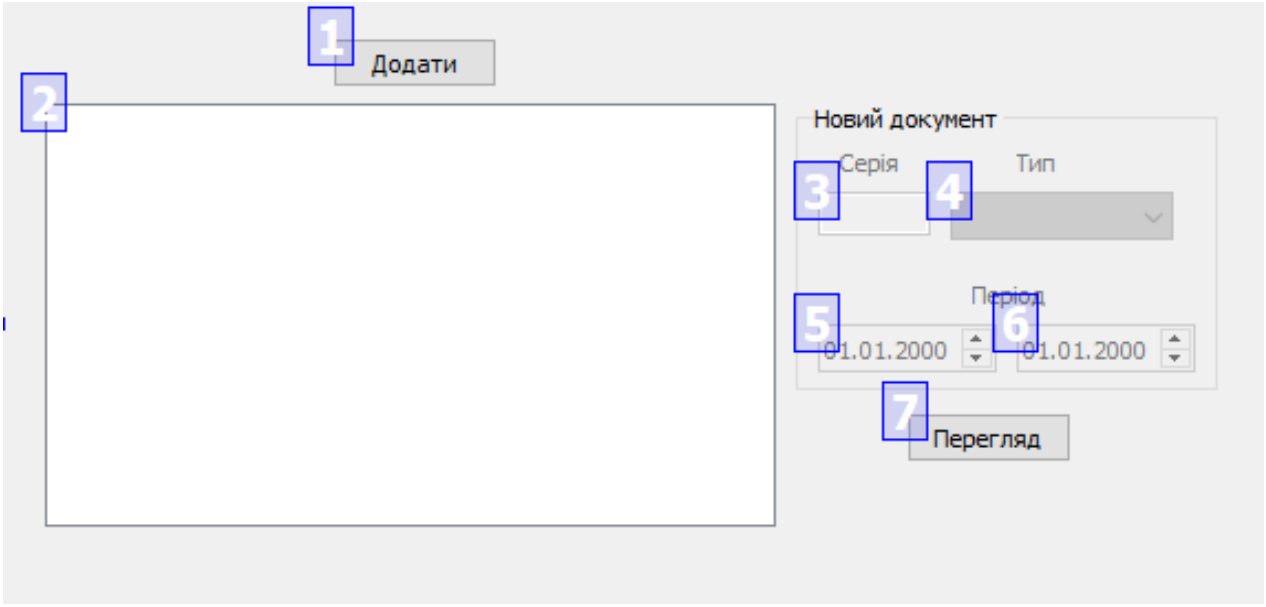



Рис. 2.10. Форма documents

Дана форма складається з наступних компонентів:

1. QPushButton - кнопка для додавання нового документа в БД.
2. QTableWidget - таблиця для виводу введених документів.
3. QLineEdit - поле для виводу серії документа.
4. QComboBox - список типів документів.
5. QDateEdit - поле для введення дати початок дії документа.
6. QDateEdit - поле для введення дати кінця дії документа.

7. QPushButton - кнопка для виводу документа у word.

На рис. 2.11 зображено форму positions з усіма пронумерованими його компонентами.

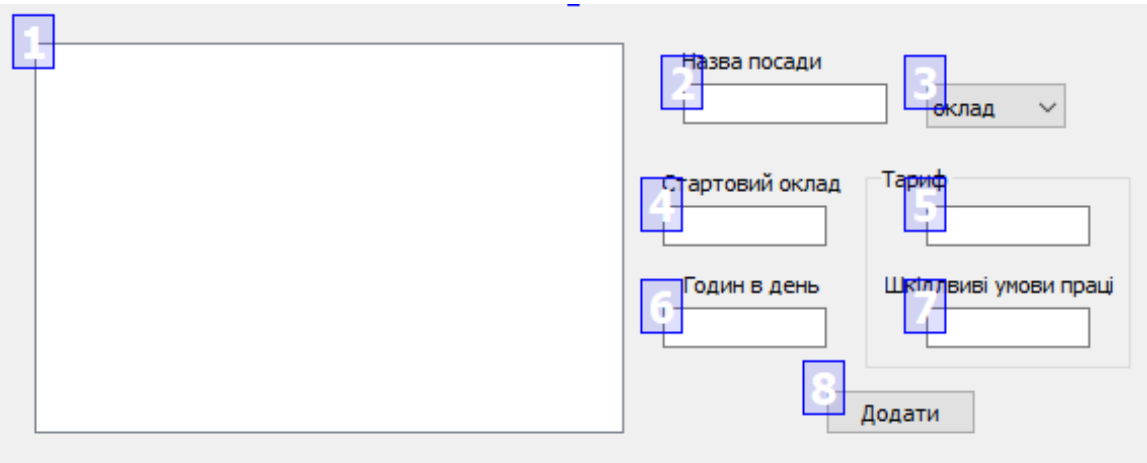



Рис. 2.11. Форма documents

Дана форма складається з наступних компонентів:

1. QTableWidget - таблиця для виводу посад.
2. QLineEdit - поле для виводу назви посади.
3. QComboBox - список типів рахування зарплати.
4. QLineEdit - поле для виводу стартового окладу.
5. QLineEdit - поле для виводу тарифу.
6. QLineEdit - поле для виводу робочих годин на 1 день.
7. QLineEdit - поле для виводу процентів за шкідливі умови праці.
8. QPushButton - кнопка для редагування або додавання.

### 2.4.4. Схема і опис бази даних

Інформаційна система використовує базу даних для зберігання інформації про робітників на підприємстві та їх заробітною плату за відпрацьовані місяці. Умовна схема бази даних зображена на рис. 2.12.

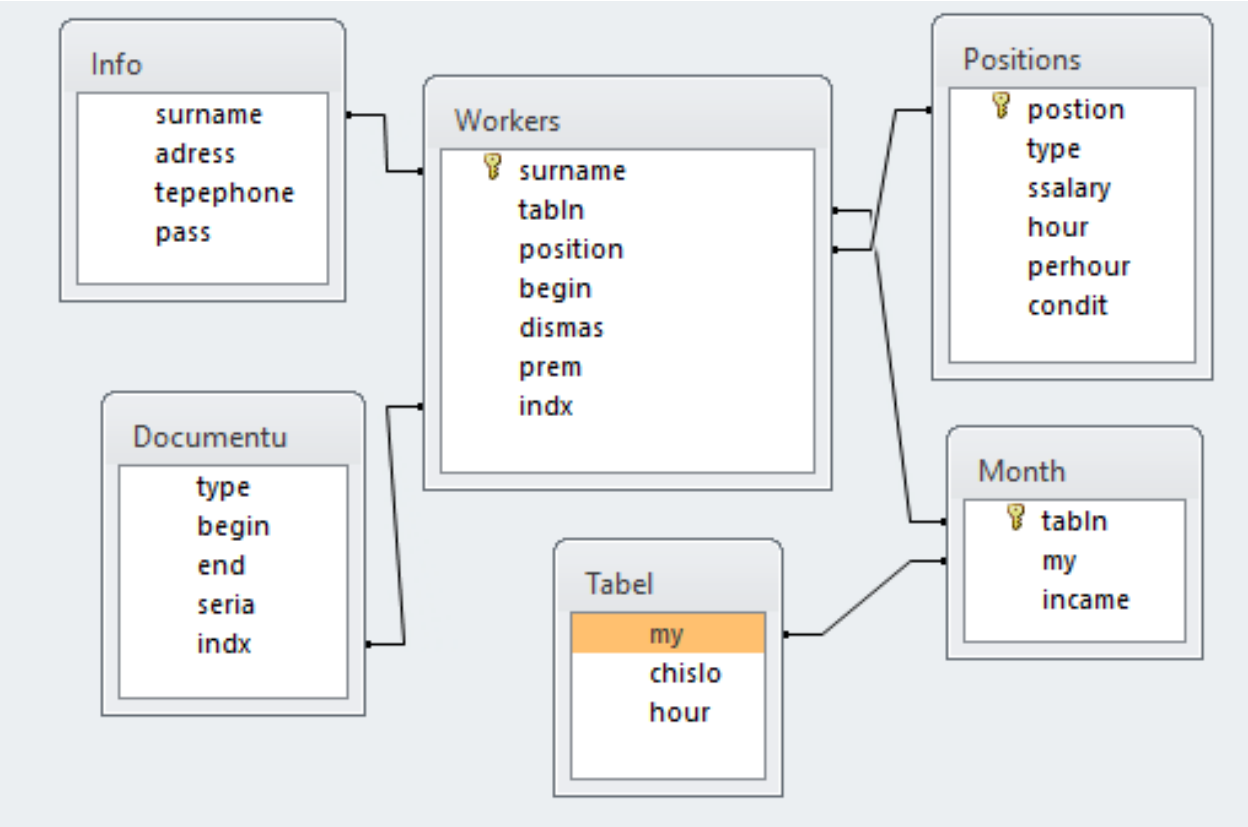



Рис. 2.12. Структурна схема бази даних

Програма використовує базу даних, яка має 6 таблиць: positions, workers, documents, info,month, tabel.

Поля таблиці «positions» зображені в таблиці 2.2 .

Таблиця 2.2

**Таблиця positions «Посади»**

| Поле | Призначення |
|---|---|
| position | Назва посади |
| type | Тип розрахунку |
| ssalary | Оклад |
| hour | Годин в день |
| perhour | Тариф |
| condit | Шкідливі умови праці |

Модель Course:

```
CREATE TABLE "Positions" (
	"position"	TEXT NOT NULL,
	"type"	TEXT NOT NULL,
	"ssalary"	REAL NOT NULL,
	"hour"	REAL NOT NULL,
	"perhour"	REAL,
	"condit"	REAL,
	PRIMARY KEY("position")
);
```

Поля таблиці «workers» зображені в таблиці 2.3 .

Таблиця 2.3

**Таблиця workers «Робітники»**

| Поле | Призначення |
|---|---|
| surname | Прізвище |
| tabln | Табельний номер |
| positions | Посада |
| begin | Початок роботи |
| dismas | Дата звільнення |
| prem | Премія |
| indx | Ідентифікаційний номер |

Модель workers:

```
CREATE TABLE "Workers" (
	"surname"	TEXT NOT NULL UNIQUE,
	"tabln"	INTEGER NOT NULL UNIQUE,
	"position"	TEXT NOT NULL,
```

```
    "begin"      TEXT NOT NULL,
    "dismas"     TEXT,
    "prem"       REAL NOT NULL,
    "indx"       TEXT NOT NULL UNIQUE,
    FOREIGN KEY("position") REFERENCES "Positions"("position"),
    PRIMARY KEY("surname","indx","tabln"));
```

Поля таблиці «info» зображені в таблиці 2.4 .

Таблиця 2.4

**Таблиця info «Данні про робітників»**

| Поле | Призначення |
|---|---|
| surname | Прізвище |
| adress | Домашній адрес |
| telephone | Мобільний телефон |
| pass | Пароль |

Модель info:

```
CREATE TABLE "Info" (
    "surname"   TEXT NOT NULL,
    "adress"    TEXT,
    "telephone" INTEGER,
    "pass"      TEXT,
    FOREIGN KEY("surname") REFERENCES "Workers"("surname")
);
```

Поля таблиці «documents» зображені в таблиці 2.5 .

Таблиця 2.5

**Таблиця documents «Вхідні документи»**

| Поле | Призначення |
|---|---|
| serial | Серія документа |
| type | Тип документа |
| begin | Початок |
| end | Кінець |
| indx | Ідентифікаційний номер |

Модель documents:

```
CREATE TABLE "Documentu" (
        "seria"         TEXT NOT NULL,
        "type"          TEXT NOT NULL,
        "begin"         TEXT,
        "end" TEXT,
        "indx"          TEXT NOT NULL,
        "sum"REAL,
        FOREIGN KEY("indx") REFERENCES "Workers"("indx")
);
```

Поля таблиці «month» зображені в таблиці 2.6 .

Таблиця 2.6

**Таблиця month «Місяць»**

| Поле | Призначення |
|---|---|
| tabln | Табельний номер |
| my | Рік та місць |
| incame | Нараховано |

Модель month:

```
CREATE TABLE "Month" (
        "tabln"         TEXT NOT NULL,
        "my"  TEXT NOT NULL UNIQUE,
        "incame"      REAL,
        FOREIGN KEY("tabln") REFERENCES "Workers"("tabln"),
        PRIMARY KEY("my")
);
```

Поля таблиці «tabel» зображені в таблиці 2.7 .

Таблиця 2.7

**Таблиця tabel «Табель»**

| Поле | Призначення |
|---|---|
| my | Кількість днів |
| chislo | Рік та місць |
| hour | Відпрацьовані години |

Модель tabel:

```
CREATE TABLE "Tabel" (
        "my"  TEXT NOT NULL,
        "chislo"       INTEGER NOT NULL,
        "hou" TEXT,
        FOREIGN KEY("my") REFERENCES "Month"("my")
);
```

## 2.5. Обґрунтування та організація вхідних та вихідних даних програми

Вхідні дані:

1. Робочі місця: посада, тип оплати, оклад, кошти за відпрацьовану годину, кількість годин в день, тариф, шкідливі умови праці.
2. Робітники: табельний номер, прізвище, посада, дата прийняття на роботу, дата звільнення, премія.
3. Місяці: рік та місяць, табельний номер, нараховане.
4. Табель: рік та місяць, число, часи.
5. Документи: табельний номер, серія, тип, початок, кінець.
6. Інформація: прізвище, адрес, телефон, пароль.

Інформаційна система використовує базу даних для зберігання інформації про робітників на підприємстві та їх заробітною плату за відпрацьовані місяці.

Вихідні дані: відповіді на запити бази даних та розрахунок зарплати робітників.

## 2.6. Опис роботи розробленої системи

### 2.6.1. Використані технічні засоби

Для швидкої і стабільної роботи з розробленою ІС технічні параметри повинні відповідати встановленій версії операційної системи. Для розробки даної ІС використовувався ПК з наступними технічними характеристиками:

- процесор із тактовою частотою 1 ГГц 64-розрядний ;
- оперативна пам’ять 2 ГБ;
- графічний пристрій із підтримкою DirectX 9 і драйвером WDDM 1.0 або новішим.

### 2.6.2. Використані програмні засоби

Розроблена ІС написана на об'єктно-орієнтованій мові програмування з безпечною системою типізації C++. Розробка велася в інтегрованому середовищі розробки Microsoft Visual Studio 2017 Community. БД програмного додатку була розроблена на SQL Server.

### 2.6.3. Виклик та завантаження програми

Запуск додатку «Zarplata» відбувається за допомогою файлу Zarplata.exe після копіювання на ПК директорії проєкту «Zarplata» .

### 2.6.4. Опис інтерфейсу користувача

Постановка задачі кваліфікаційної роботи вимагає забезпечення безпеки доступу до даних програми. Тому було створено авторизацію під час входження в програму, яка вимагає введення паролю обраного працівника із списку. Авторизація зображена на рис. 2.13.

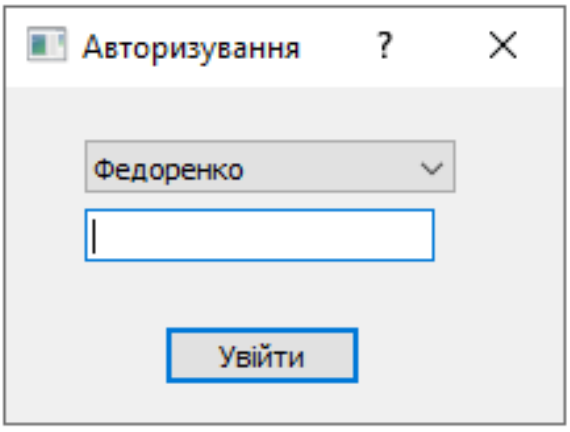



Рис. 2.13. Авторизація користувача

Після авторизації користувача відкривається таблиця зі всіма робітниками, які введені в базу. Кнопка «Посади» доступна тільки адміністратору. Результат зображено на рис. 2.13.

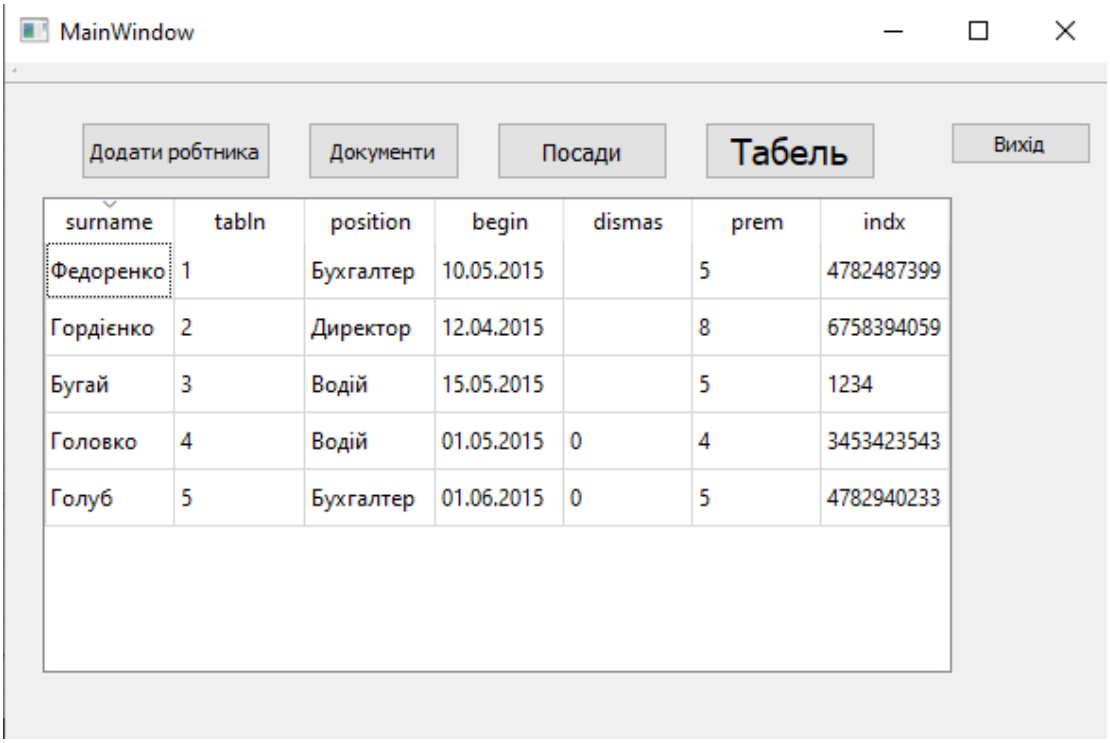



Рис. 2.13. Список робітників

Над кожним працівником можна провести окрему операцію за допомогою контекстного меню . Список операцій зображено на рис. 2.13.

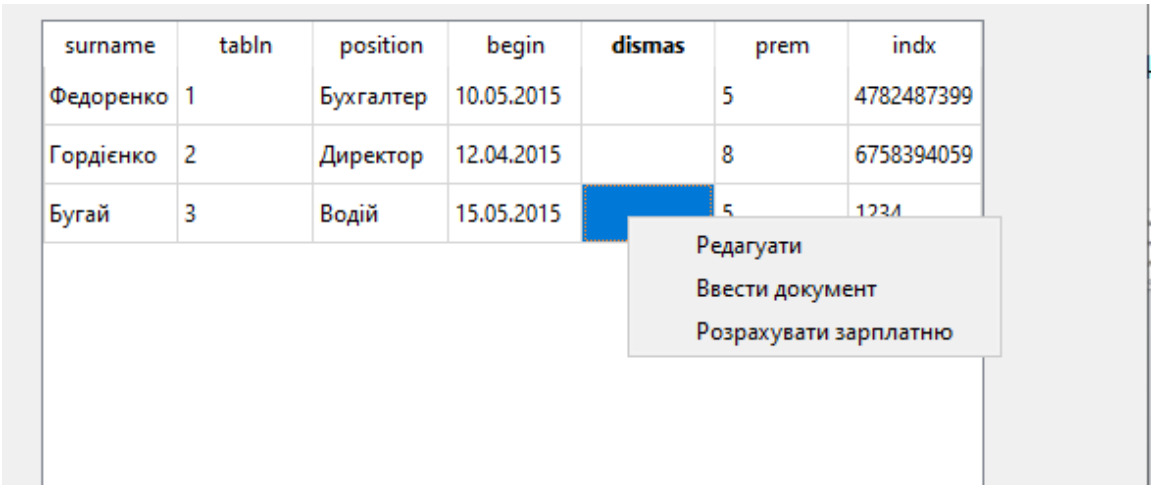



Рис. 2.14. Контекстне меню

Після натискання на кнопку «Додати робітника» відкриється вікно, в якому потрібно заповнити поля  для нового робітника. Табельний номер та дата початку роботи формується автоматично. Вікно зображено на рис. 2.14.

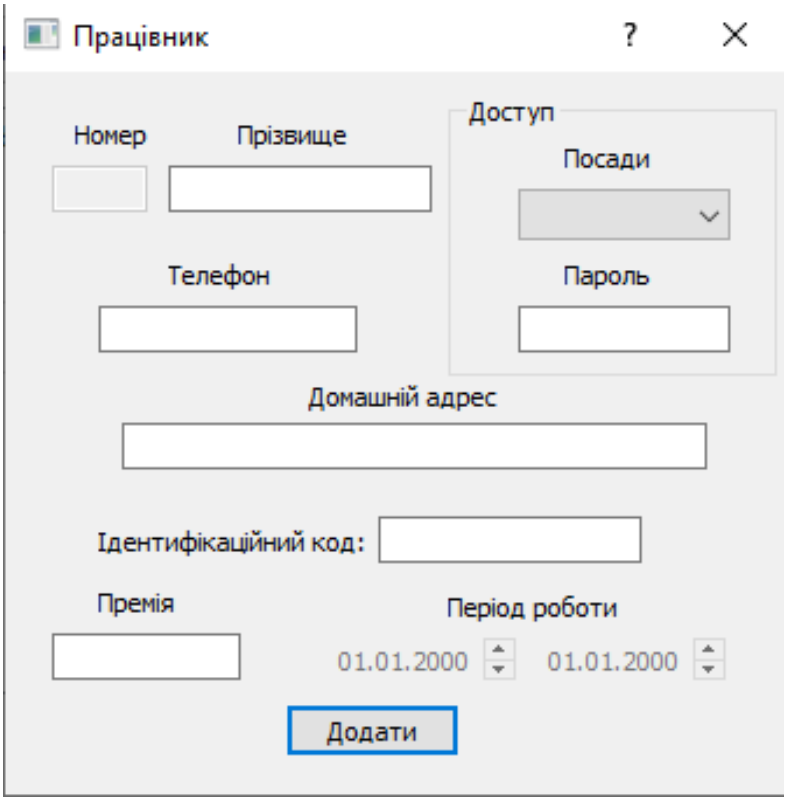



Рис. 2.14. Вікно «Працівник»

Після натискання в контекстному меню на «редагування» відкриється відповідне вікно, яке зображено на рис. 2.15.

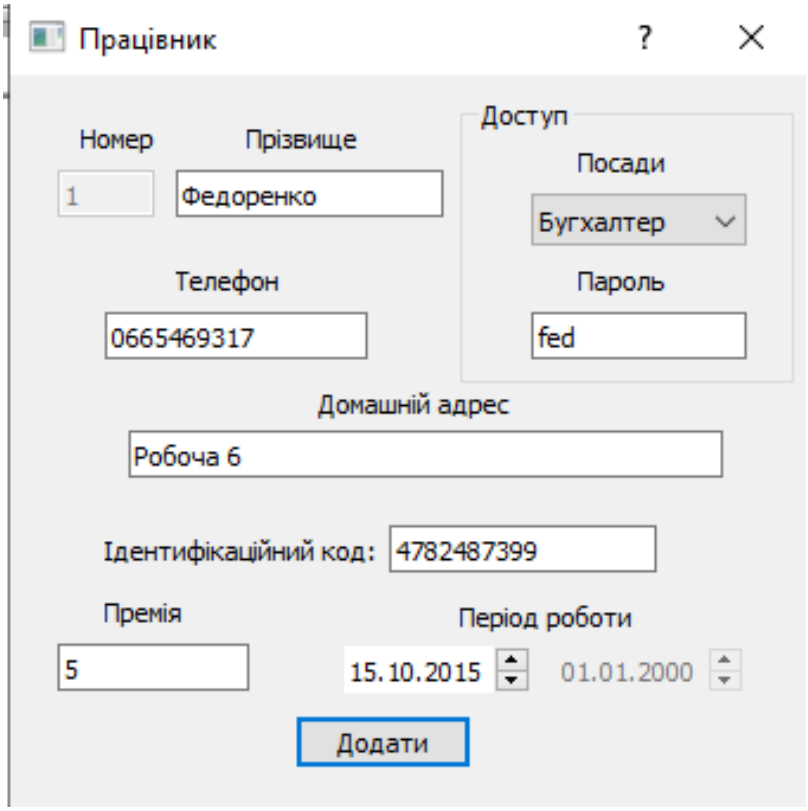



Рис. 2.15. Вікно для редагування

Після натискання на кнопку «Документи» відкриється вікно зі списком усіх введених документів. Якщо викликати цю функцію через контекстне меню то відкриються тільки документи для обраного працівника. В даному вікні можна додавати, редагувати та видаляти документи. Вікно зображено на рис. 2.16.

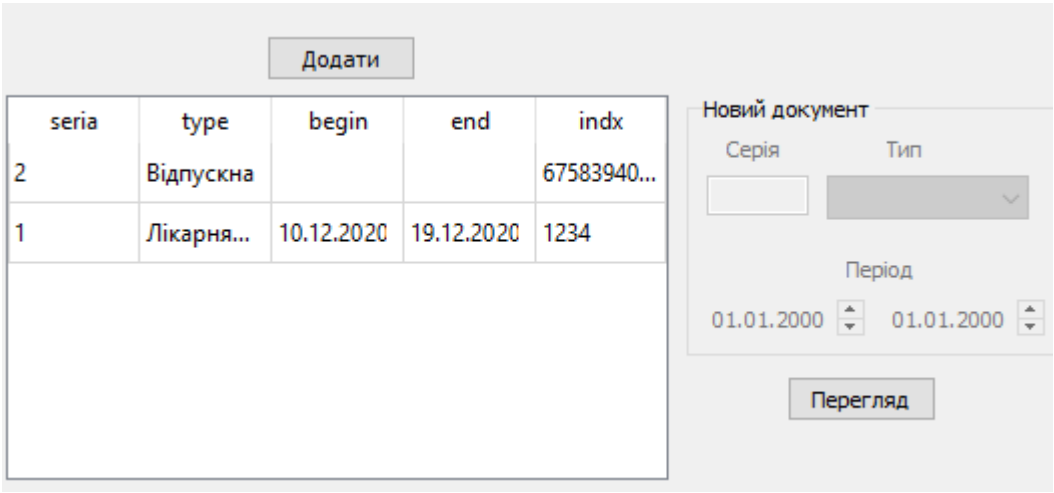
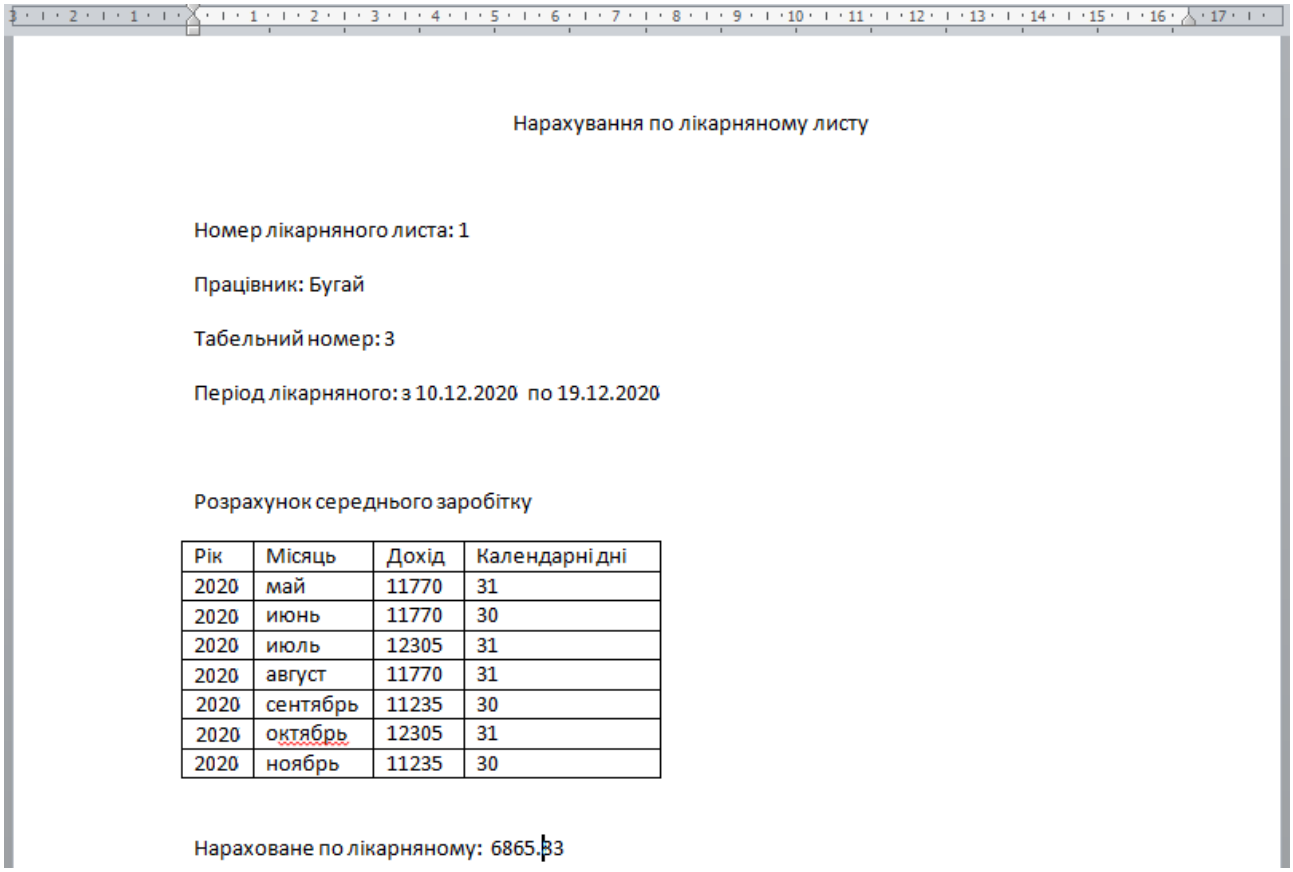



Рис. 2.16. Вікно «Документи»

Для перегляду документа в текстовому виді, потрібно клацнути два рази на документі зі списку та натиснути кнопку «Перегляд». Приклад документа зображено на рис. 2.17.



Нарахування по лікарняному листу

Номер лікарняного листа: 1

Працівник: Бугай

Табельний номер: 3

Період лікарняного: з 10.12.2020 по 19.12.2020

Розрахунок середнього заробітку

| Рік | Місяць | Дохід | Календарні дні |
|---|---|---|---|
| 2020 | май | 11770 | 31 |
| 2020 | июнь | 11770 | 30 |
| 2020 | июль | 12305 | 31 |
| 2020 | август | 11770 | 31 |
| 2020 | сентябрь | 11235 | 30 |
| 2020 | октябрь | 12305 | 31 |
| 2020 | ноябрь | 11235 | 30 |

Нараховане по лікарняному: 6865.33

Рис. 2.17. Лікарняний лист

Після натискання на кнопку «Табель» відкриється вікно з таблицею, яка заповнюється автоматично робочими годинами на робочі дні. Всі пропуски це вихідні дні. Користувач може обрати рік та місяць для рахування зарплати. Якщо на обраний місяць буде зареєстрований документ: лікарняний, відпускний або відрядження, то це буде відображено у табелі іншим типом часу. Вікно зображене на рисунку 2.18.

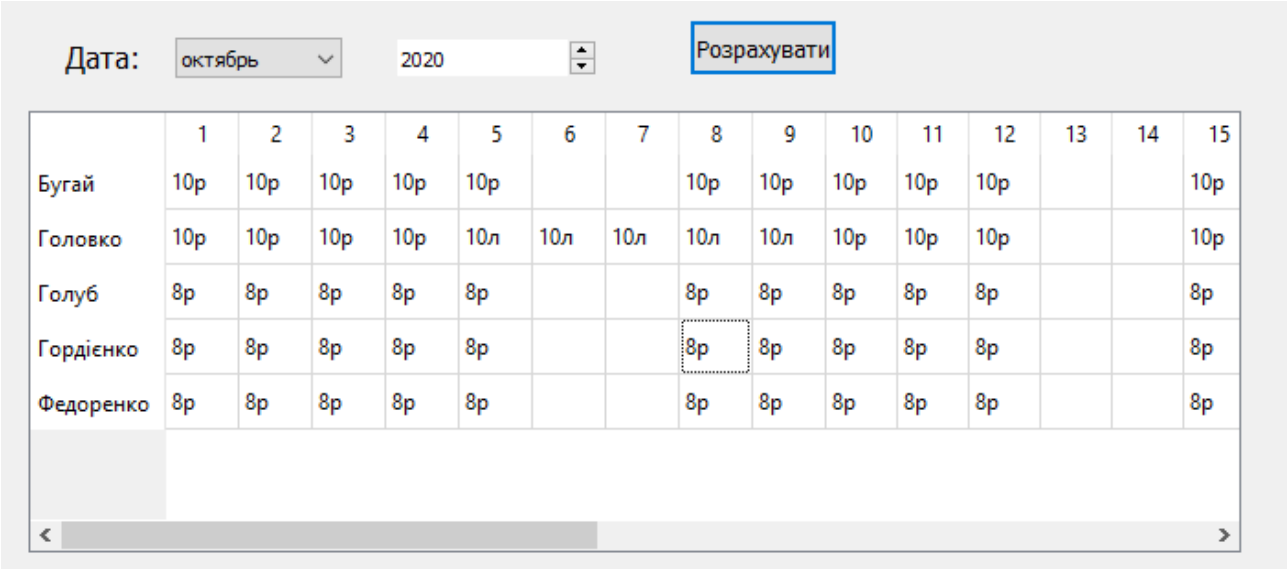



Рис. 2.18. Заповнений табель

Після натиснення на кнопку «Розрахувати» в останній колонці з’явиться результат розрахунку. Результат зображено на рис. 2.19.

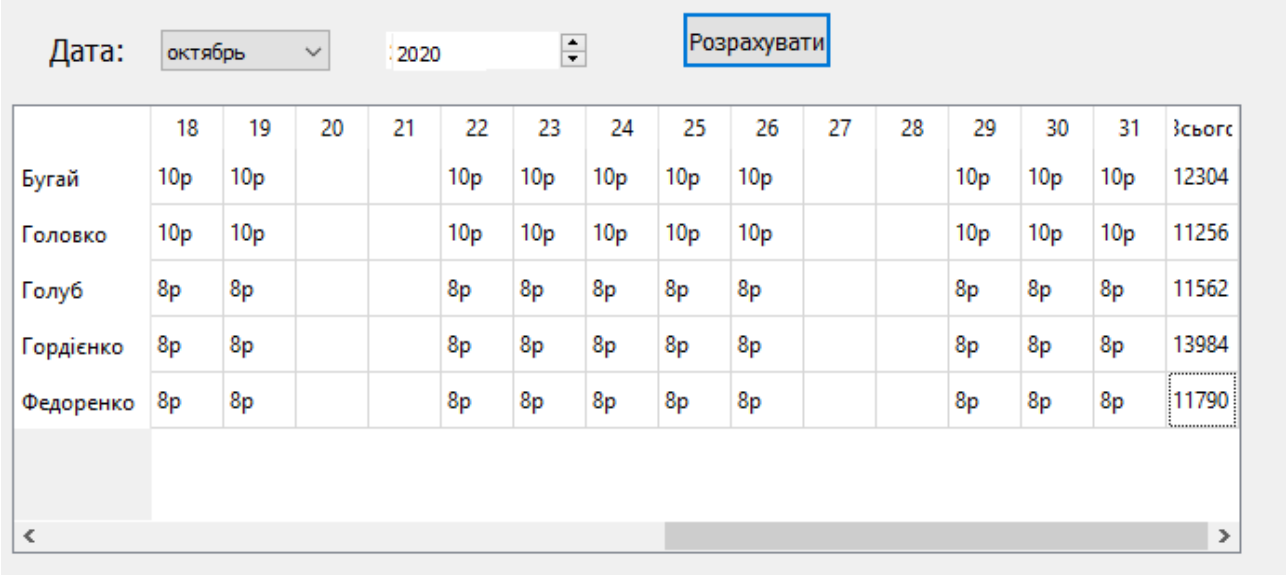



Рис. 2.19. Результат розрахунку

Після натиснення на кнопку «Посади» з’явиться вікно для додавання та редагування посад на підприємстві. Результат зображено на рис. 2. 20.

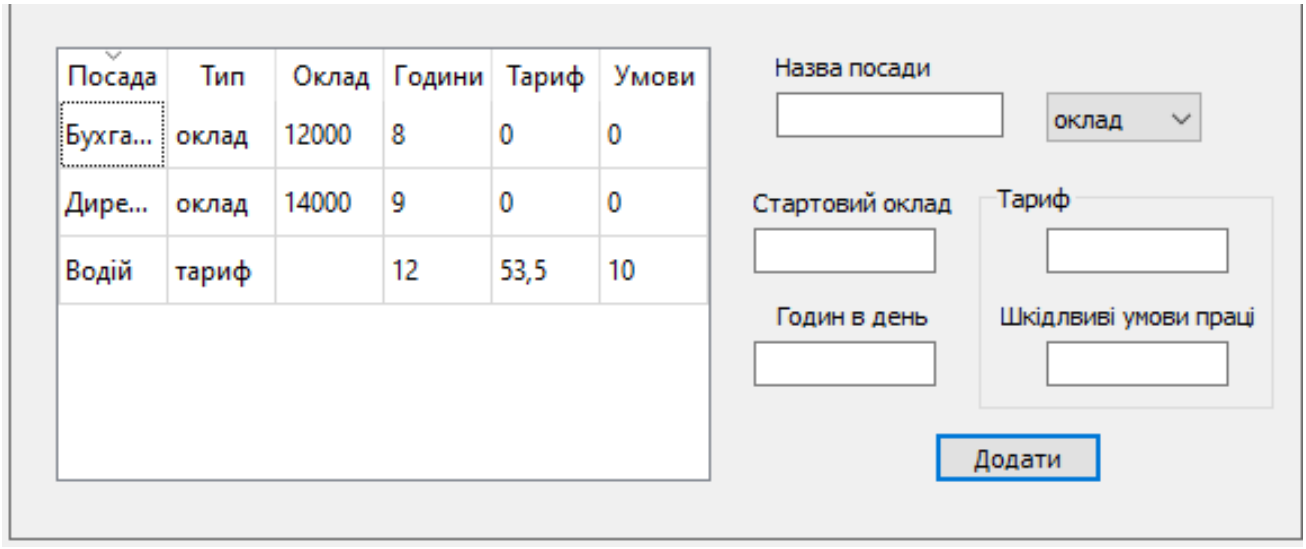



Рис. 2.20. Вікно «Посади»

### 2.6.5. Тестування програми

Коректність відображення сторінок, відсутність непрацюючих посилань, 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 додатку, тобто ситуації, що можуть призвести до виникнення помилок [20].

При вході в програму виконується перевірка введеного пароля. В разі невірності з’явиться відповідне повідомлення, яке зображене на рис. 2.21.

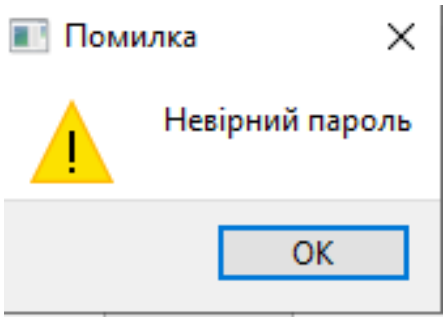


Рис. 2.21. Повідомлення про невірність паролю

Для деяких полів у візуальних формах були присвоєні маски для введення тільки:

– букв: ui->lineEdit->setInputMask("99999999999");

– цифр: ui->lineEdit->setInputMask("AAAAAA").

При помилці редагування з‘явиться повідомлення, яке зображено на рис. 2.22.

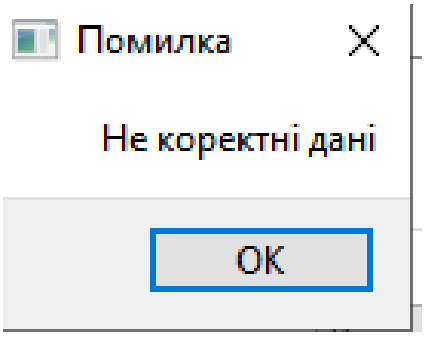



Рис. 2.22. Повідомлення про помилку

При додаванні і редагуванні співробітника виконуються ряд перевірок:

– пароль може водитись тільки для окладників;

– дата звільнення не раніше дати початку роботи;

– коректний період документів;

– на одну дату не може бути введено більше одного документа;

– правильне рахування заробітної плати;

– при заповнення табеля, виконується перевірка на виявлення вихідних днів та святкових.

# РОЗДІЛ 3
# ЕКОНОМІЧНИЙ РОЗДІЛ

## 3.1. Розрахунок трудомісткості та вартості розробки програмного продукту

Вхідні дані:

- передбачуване число операторів – 1500;
- коефіцієнт складності програми – 2;
- коефіцієнт корекції програми в ході її розробки – 0,08;
- годинна заробітна плата програміста, грн / год – 30.

Нормування праці в процесі створення ПЗ істотно ускладнено в силу творчого характеру праці програміста. Тому трудомісткість розробки ПЗ може бути розрахована на основі системи моделей з різною точністю оцінки.

Трудомісткість розробки ПЗ можна розрахувати за формулою:

$$t = t_u + t_a + t_n + t_{отл} + t_д, \text{ людино-годин,} \quad (3.1)$$

де to - витрати праці на підготовку й опис поставленої задачі (приймається 50),

ти - витрати праці на дослідження алгоритму рішення задачі,

та - витрати праці на розробку блок-схеми алгоритму,

тп - витрати праці на програмування по готовій блок-схемі,

тотл - витрати праці на налагодження програми на ЕОМ,

тд - витрати праці на підготовку документації.

Складові витрати праці визначаються через умовне число операторів у ПЗ, яке розробляється.

Умовне число операторів (підпрограм):

$$Q = q \times C \times (1+ p) \text{ , людино-годин,} \qquad (3.2)$$

де q - передбачуване число операторів,

С - коефіцієнт складності програми,

р - коефіцієнт кореляції програми в ході її розробки.

$$Q = 1500 \cdot 2 \cdot (1 + 0{,}08) = 3240 \text{ людино-годин.}$$

Витрати праці на вивчення опису задачі ti визначається з урахуванням уточнення опису і кваліфікації програміста:

$$t_u = \frac{QB}{(75...85)K} \text{ , людино-годин,} \qquad (3.3)$$

де В - коефіцієнт збільшення витрат праці внаслідок недостатнього опису задачі; В=1.2 ... 1.5,

k - коефіцієнт кваліфікації програміста, обумовлений від стажу роботи з даної спеціальності.

Витрати праці на розробку алгоритму рішення задачі:

$$t_u = \frac{3240 \cdot 1{,}3}{80 \cdot 1{,}2} = 44 \text{ , людино-годин.}$$

Витрати на складання програми по готовій блок-схемі:

$$t_a = \frac{Q}{(20...25)K} \quad \text{людино-годин.} \qquad (3.4)$$

$$t_a = \frac{3240}{22 \cdot 1{,}2} = 123 \text{ людино-годин.}$$

Витрати на складання програми по готовій блок-схемі:

$$t_n = \frac{Q}{(20...25)K} \quad \text{людино-годин.} \tag{3.5}$$

$$t_n = \frac{3240}{23 \cdot 1{,}2} = 117 \text{ людино-годин.}$$

Витрати праці на налагодження програми на ЕОМ:

$$t_{отл} = \frac{Q}{(4...5)K} \quad \text{людино-годин.} \tag{3.6}$$

$$t_{отл} = \frac{3240}{5 \cdot 1{,}2} = 540 \text{ людино-годин.}$$

Витрати праці на підготовку документації:

$$t_д = t_{др} + t_{до}, \text{ людино-годин,} \tag{3.7}$$

де tдр - трудомісткість підготовки матеріалів і рукопису.

$$t_{др} = \frac{Q}{(15...20)K}, \text{ людино-годин.} \tag{3.8}$$

$$t_{др} = \frac{3240}{18 \cdot 1{,}2} = 150 \text{ людино-годин,}$$

tдо - трудомісткість редагування, печатки й оформлення документації

$$t_{до} = 0{,}75 \cdot t_{др}, \text{ людино-годин.} \tag{3.9}$$

$$t_{до} = 150 + 73 = 223 \text{ людино-годин.}$$

Отримуємо трудомісткість розробки програмного забезпечення:

$$t = 50 + 44 + 123 + 117 + 540 + 223 = 1097 \text{ людино-годин.}$$

### 3.2. Розрахунок витрат на створення програми

Витрати на створення ПЗ Кпо включають витрати на заробітну плату виконавця програми Зз/п і витрат машинного часу, необхідного на налагодження програми на ЕОМ.

$$K_{ПО} = З_{ЗП} + З_{МВ}, \text{ грн,} \quad (3.10)$$

Заробітна плата виконавців визначається за формулою:

$$З_{ЗП} = t \cdot C_{ПР}, \text{ грн,} \quad (3.11)$$

де t - загальна трудомісткість, людино-годин,

Спр - середня годинна заробітна плата програміста, грн/година.

$$З_{ЗП} = 1097 \cdot 30 = 32910 \text{ грн.}$$

Вартість машинного часу, необхідного для налагодження програми на ЕОМ:

$$З_{МВ} = t_{отл} \times C_{М}, \text{ грн,} \quad (3.12)$$

де tотл - трудомісткість налагодження програми на ЕОМ, год,

Смч - вартість машино-години ЕОМ, грн/год.

Визначені в такий спосіб витрати на створення програмного забезпечення є частиною одноразових капітальних витрат на створення АСУП.

$$З_{МВ} = 540 \times 5 = 2700 \text{ грн.}$$

$$Кпо = 2700 + 32910 = 35610 \text{ грн.}$$

Очікуваний період створення ПЗ:

$$T = \frac{t}{B_k \cdot F_p} \text{ міс.} \qquad (3.13)$$

де Bk - число виконавців,

Fp - місячний фонд робочого часу (при 40 годинному робочому тижні Fp=176 годин).

$$T = \frac{1097}{1 \cdot 176} = 6.23 \text{ міс.}$$

Висновки: визначено трудомісткість розробленого додатку (1097 люд-год), проведений підрахунок вартості роботи по створенню програми (35610 грн.) та розраховано час на його створення (6,2 міс).

# ВИСНОВКИ

Темою даного проєкту є «Розробка інформаційної системи ведення обліку заробітної плати на підприємстві засобами мови С++».

Даний програмний продукт був створений для ведення місячного обліку зарплат робітників на підприємстві. Дана програма спрощує роботу бухгалтера та створює комфортні можливості для виконання своїх обов'язків.

Результат використання роботи представлено у вигляді інформаційної системи, яка містить структуру спроектованих таблиць, схему даних зі зв'язками між таблицями, екранні форми для занесення вхідних даних та екранні форми для перегляду результатів їх обробки, запити для аналізу даних, звіти, головну форму.

Розроблена інформаційна система призначена для виконання наступних функцій:

- авторизація працівника за паролем;
- перегляд даних робітника;
- ведення списку робітників на підприємстві;
- ведення лікарняних, відпускних і відряджень;
- введення та редагування табелю працівника за місяць;
- ведення документів;
- редагування даних працівника;
- швидкий розрахунок зарплатні;
- друк звітів
- перевірка на коректність введених даних..

Розроблена програма виконує всі функції відповідає всім вимогам, які були задані у постановці завдання.

Дана програма спрощує роботу бухгалтера та створює йому комфортні можливості для виконання своїх обов'язків, підвищує ефективність праці

робітників та надає доступ до застосування сучасних інформаційних технологій в своїй роботі.

В економічному розділі визначено трудомісткість розробленого додатку (1097 люд-год), проведений підрахунок вартості роботи по створенню програми (35610 грн.) та розраховано час на його створення (6,2 міс).

## СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Бібліографічний запис. Бібліографічний опис. Загальні вимоги та правила складання : (ГОСТ 7.1-2003, IDT) : ДСТУ ГОСТ 7.1:2006. – Чинний з 2007–07–01. – К. : Держспоживстандарт України, 2007. – 47 с. – (Система стандартів з інформації, бібліотечної та видавничої справи) (Національний стандарт України).

2. Бланшет Ж. Qt 4: Програмування GUI на C++. 2-е додаткове видання/Ж. Бланшет, М. Саммерфілд. - М.: Кудиц-пресс, 2008. - 216 с.

3. Бенджамін П. Types and Programming Languages/П. Бенджамін. - MIT Press, 2002. - 432 с.

4. Бухгалтерія URL: http://1c.ua/ua/v8/RegionalSolutions_UA_BUH.php// дата звернення: 5.04.2021.

5. Бусигін Б.С., Коротенко Г.М., Коротенко Л.М. Прикладна інформатика. Підручник для студентів комп'ютерних спеціальностей. – Дніпропетровськ: Видавництво НГУ, 2004. – 559 с. URL: http://www.programmer.dp.ua/book-ua-k01.php. дата звернення: 15.03.2019.

6. Бусыгин Б.С., Дивизинюк М.М., Коротенко Г.М., Коротенко Л.М. Введение в современную информатику. Учебник. – Севастополь: Издательство СНУЯЭиП, 2005. – 644 с. / URL: http://www.programmer.dp.ua/book-ru-k02.php. дата звернення: 15.01.2018.

7. Вставка формул в word URL: https://evileg.com/ru/forum/topic/596/ дата звернення: 5.04.2021.

8. Використання календаря URL: http://blog.kislenko.net/show.php?id=1507/ дата звернення: 5.04.2021.

9. Джозеф Шмуллер. Оволодій самостійно UML 2 за 24 часа/Ш. Джозеф //Практичне руководство: Sams Teach Yourself UML in 24 Hours.Complete Starter Kit/М. Вильямс. – 2005.— 334 с.

10. Документація. Звіти у сфері науки і техніки. Структура і правила оформлення : ДСТУ 3008-95. – Чинний від 1996–01–01. – К. : Держстандарт України, 1996. – 39 с.

11. ДСТУ 2394-94 Інформація та документація. Комплектування фонду, бібліографічний опис, аналіз документів. Терміни та визначення. – Чинний від 01.01.1995. - Київ : Держстандарт України, 1994. – 88 с.

12. Методичні вказівки з виконання економічного розділу в дипломних проектах студентів спеціальності “Комп’ютерні системи ” / Укладачі О.Г. Вагонова, Нікітіна О.Б. Н.Н. Романюк – Дніпропетровськ: Національний гірничий університет. – 2013. – 23с.

13. Методичні рекомендації до виконання кваліфікаційних робіт бакалаврів напряму підготовки 122 «Комп’ютерні науки» галузі знань 12 Інформаційні технології/, Л.М. Коротенко , О.С. Шевцова; Нац. гірн. ун-т. – Д : ДВНЗ НГУ, 2019. – 65 с.

14. Методичні рекомендації щодо написання, оформлення та представлення учнівських науково-дослідницьких робіт учнів – членів Малої академії наук України / Г.Г. Півняк, Л.М. Коротенко, І.М. Удовик, Є.М. Головня – Д.: ДВНЗ «Національний гірничий університет», 2017. – 24 с.

15. Офіційний сайт середовища розробки Visual Studio Code /URL: документація - Режим доступу : https://code.visualstudio.com/docs. дата звернення: 5.04.2021.

16. Скорочення слів в українській мові у бібліографічному описі. Загальні правила та вимоги : ДСТУ 3582-97. – Чинний від 1998–07–01. – К. : Держстандарт України, 1998. – 24 с. – (Державний стандарт України).

17. Фаулер С. «UMLосновы» С. Фаулер, К. Скотт – СПб: Символ-плюс, 2002. – 167.

18. Фуртат Ю.О. «Про вплив адаптивних користувацьких інтерфейсів на надійність та ефективність функціонування автоматизованих систем». – URL: http://ntv.ifmo.ru/file/article/8397.pdf. дата звернення: 5.04.2021.

19. Царев, В.А. Проектирование, анализ и программная реализация структур данных и алгоритмов: Учебное пособие / В.А. Царев, А.Ф. Дробанов. – Череповец., 2007. – 84 с.

20. Шлее Макс. Qt 5.3 Професіональне програмування на C++./М. Шлее. - СПб.: БХВ-Петербург, 2018. - 500 с.

21. Шуремов Е.Л., Умнова Э.А., Воропаева Т.В. Автоматизированные информационные системы бухгалтерского учета, анализа, аудита: Учебное пособие для вузов / М: Перспектива, 2001. 363 с.

**ДОДАТОК А**

# КОД ПРОГРАМИ

autoriz.cpp

```
autoriz::autoriz(QWidget *parent) :
    QDialog(parent),
    ui(new Ui::autoriz)
{

    db = QSqlDatabase::addDatabase("QSQLITE");
    db.setDatabaseName("E:\\bazax.db");
    db.open();
    ui->setupUi(this);
    QSqlQuery query;
    query.exec("SELECT w.surname FROM workers w INNER JOIN positions p ON w.position =
p.position where p.type='оклад'");
   //qDebug()<<query.first();
query.seek(-1);
   while (query.next()){
      ui->comboBox->addItem(query.value(0).toString());
   }
       db.close();
}

autoriz::~autoriz()
{
    delete ui;
}

void autoriz::on_pushButton_clicked()
{
QString n;
n=ui->comboBox->currentText();
 db.open();
```

```
QSqlQuery query;
query.exec("SELECT * FROM Info WHERE surname = \'"+n+"\' AND pass = \'"+ui->lineEdit-
>text()+"\'");
if(query.first()){  MainWindow* w=new MainWindow();w->show();db.close();this->close();}
else {QMessageBox::warning(this,"Помилка","Невірний пароль");}
}
```

mainwindow.cpp

```
#include "mainwindow.h"
#include "ui_mainwindow.h"
#include "newworker.h"
#include "documents.h"
#include "tabel.h"
#include "positions.h"
#include <QtGui>
#include <QVBoxLayout>
#include "QDebug"

MainWindow::MainWindow(QWidget *parent) :
    QMainWindow(parent),
    ui(new Ui::MainWindow)
{
    bz = QSqlDatabase::addDatabase("QSQLITE");
    bz.setDatabaseName("E:\\bazax.db");
    bz.open();
    ui->setupUi(this);
    ui->table1->setSortingEnabled(true);
    QSortFilterProxyModel *proxyModel = new QSortFilterProxyModel(this);
        model->setQuery(QSqlQuery(("SELECT * FROM Workers "), bz));
        proxyModel->setSourceModel(model);
        ui->table1->setModel(proxyModel);
        ui->table1->show();
        ui->table1->verticalHeader()->setVisible(false);
        QVBoxLayout *l=new QVBoxLayout(this);
```

```
            // table1=new QTableView(ui->table1);
             ui->table1->setContextMenuPolicy(Qt::CustomContextMenu);
             connect(ui->table1, SIGNAL(on_table1_customContextMenuRequested(const QPoint
&pos)),this,
                     SLOT(on_table1_customContextMenuRequested(const QPoint &pos)));
            l->addWidget(ui->table1);
            ui->table1->horizontalHeader()->setSectionResizeMode(QHeaderView::Stretch);

bz.close();
}

MainWindow::~MainWindow()
{
   delete ui;
}

void MainWindow::on_pushButton_clicked()
{
   newworker* w=new newworker(bz);
  w->show();
  connect(w,SIGNAL(buttonClicked()),this,SLOT(obnov()));

    //this->close();
}

void MainWindow::on_table1_customContextMenuRequested(const QPoint &pos)
{

   QModelIndex index;
         index=ui->table1->indexAt(pos);
    // int a=1,b=2,c=3;
      QMenu *menu=new QMenu(this);
      QString d;
      QModelIndex ind = ui->table1->selectionModel()->currentIndex();
      qDebug()<<ind.row();
```

```
        d=ui->table1->model()->index(ind.row(),0).data().toString();
        qDebug()<<d;
        QAction * redakt=new QAction("Редагуати",this);
        QAction * docum=new QAction("Ввести документ",this);
        QAction * salar=new QAction("Розрахувати зарплатню",this);

        connect(redakt, &QAction::triggered, this,[=](){;newworker* w=new newworker(d,bz);
           w->show();bz.close();});
        connect(docum, &QAction::triggered, this,[=](){;documents* w=new documents(d);
           w->show();bz.close();});
        //connect(salar, &QAction::triggered, this,[=](){;newworker* w=new newworker(d);
          //  w->show();});
       menu->addAction(redakt);
       menu->addAction(docum);
       menu->addAction(salar);

        menu->popup(ui->table1->viewport()->mapToGlobal(pos));
}

void MainWindow::on_pushButton_3_clicked()
{
    this->close();
}

void MainWindow::on_pushButton_4_clicked()
{
   documents* w=new documents(bz);
              w->show();

}

void MainWindow::obnov()
{
    bz = QSqlDatabase::addDatabase("QSQLITE");
    bz.setDatabaseName("E:\\bazax.db");
```

```
 bz.open();
ui->table1->setSortingEnabled(true);
    QSortFilterProxyModel *proxyModel = new QSortFilterProxyModel(this);
          model->setQuery(QSqlQuery(("SELECT * FROM Workers "), bz));
         proxyModel->setSourceModel(model);
          ui->table1->setModel(proxyModel);
          ui->table1->show();
          ui->table1->verticalHeader()->setVisible(false);
          ui->table1->horizontalHeader()->setSectionResizeMode(QHeaderView::Stretch);
          bz.close();

}

void MainWindow::on_pushButton_2_clicked()
{
    tabel* w=new tabel(bz);
   w->show();
  // this->close();
}


void MainWindow::on_pushButton_5_clicked()
{
    qDebug()<<"Work";
    positions* b=new positions(bz);
   b->show();

}
```

documentu.cpp

```
#include "documents.h"
#include "ui_documents.h"
#include <QtGui>
#include "QFile"
#include <QAxObject>
documents::documents(QSqlDatabase bz,QWidget *parent) :
    QDialog(parent),
    ui(new Ui::documents)
{

    bz.open();
    ui->setupUi(this);
   ui->table->setSortingEnabled(true);
   QSortFilterProxyModel *proxyModel = new QSortFilterProxyModel(this);
    model->setQuery(QSqlQuery(("SELECT * FROM Documentu "), bz));
    proxyModel->setSourceModel(model);
    ui->table->setModel(proxyModel);
    ui->table->show();
    ui->table->verticalHeader()->setVisible(false);
ui->table->horizontalHeader()->setSectionResizeMode(QHeaderView::Stretch);
}

documents::documents(QString nm,QWidget *parent) :
    QDialog(parent),
    ui(new Ui::documents)
{bz = QSqlDatabase::addDatabase("QSQLITE");
    bz.setDatabaseName("E:\\bazax.db");
    bz.open();
    ui->setupUi(this);
mn=nm;
   QSortFilterProxyModel *proxyModel = new QSortFilterProxyModel(this);
    model->setQuery(QSqlQuery(("SELECT * FROM Documentu WHERE indx =(SELECT indx
FROM Workers WHERE surname=\'"+nm+"\')"), bz));
```

```
    proxyModel->setSourceModel(model);
    ui->table->setModel(proxyModel);
    ui->table->show();
    ui->table->verticalHeader()->setVisible(false);
ui->table->horizontalHeader()->setSectionResizeMode(QHeaderView::Stretch);
}

documents::~documents()
{
    delete ui;
}

void documents::on_pushButton_3_clicked()
{
    db.open();
    QSqlQuery query;
    QString docm;
    docm=ui->comboBox->currentText();
if(docm=="Лікарняний"){
    QAxObject *wordApp = new QAxObject("Word.Application");
    QAxObject *wordDoc = wordApp->querySubObject("Documents()");
    QAxObject *newDoc = wordDoc->querySubObject("Add()");

    //Заголовок для входных данных
    QAxObject *rangeInputData = newDoc->querySubObject("Range()");
    rangeInputData->dynamicCall("SetRange(int, int)", 0, 100);
    rangeInputData->setProperty("Text", "Нарахуваннння по лікарняному листу");

    /*размер, шрифт, выравнивание
    QAxObject *font_rangeInputData = rangeInputData->querySubObject("Font");
    font_rangeInputData->setProperty("Size", 12);
    font_rangeInputData->setProperty("Name", "Arial");*/
    QAxObject *alignment_rangeInputData = rangeInputData-
>querySubObject("ParagraphFormat");
    alignment_rangeInputData->setProperty("Alignment", "wdAlignParagraphCenter");
```

```
 alignment_rangeInputData->setProperty("SpaceAfter", 0);

 //отступ
 rangeInputData->dynamicCall("InsertParagraphAfter()");
 rangeInputData->dynamicCall("InsertParagraphAfter()");

 QAxObject* prange2 = newDoc->querySubObject("Range()");
prange2->dynamicCall("SetRange(int, int)",101,200);
prange2->setProperty("Text","Номер лікарняного листа № \""+ui->lineEdit->text()+"\"");
QAxObject *alignment_rangeNameT1 = prange2->querySubObject("ParagraphFormat");
alignment_rangeNameT1->setProperty("Alignment", "wdAlignParagraphLeft");

rangeInputData->dynamicCall("InsertParagraphAfter()");
rangeInputData->dynamicCall("InsertParagraphAfter()");

QAxObject* prange3 = newDoc->querySubObject("Range()");
 prange3->dynamicCall("SetRange(int, int)",201,300);
  prange3->setProperty("Text","Працівник: \""+mn+"\"");

 rangeInputData->dynamicCall("InsertParagraphAfter()");
 rangeInputData->dynamicCall("InsertParagraphAfter()");

 query.exec("SELECT tabln FROM Workers WHERE surname=\""+mn+"\"");
 query.first();
 QAxObject* prange4 = newDoc->querySubObject("Range()");
prange4->dynamicCall("SetRange(int, int)",301,400);
 prange4->setProperty("Text","Табельний номер: \""+query.value(0).toString()+"\"");
 rangeInputData->dynamicCall("InsertParagraphAfter()");
 rangeInputData->dynamicCall("InsertParagraphAfter()");

 QAxObject* prange5 = newDoc->querySubObject("Range()");
prange5->dynamicCall("SetRange(int, int)",401,500);
 prange5->setProperty("Text","Період роботи: з \""+ui->dateEdit->text()+"\' по \""+ui-
>dateEdit_2->text()+"\"");
```

```
    rangeInputData->dynamicCall("InsertParagraphAfter()");
    rangeInputData->dynamicCall("InsertParagraphAfter()");
    rangeInputData->dynamicCall("InsertParagraphAfter()");

    //Заголовок для 1 таблицы
    QAxObject *rangeNameT1 = newDoc->querySubObject("Range()");
    rangeNameT1->dynamicCall("SetRange(int, int)",501, 600);
    rangeNameT1->setProperty("Text", "Розрахуання середнього заробітку");
    //выравнивани
    rangeInputData->dynamicCall("InsertParagraphAfter()");
    rangeInputData->dynamicCall("InsertParagraphAfter()");
    //таблица 1
    QAxObject *tables = newDoc->querySubObject("Tables()");
    QAxObject *rangeTable1 = newDoc->querySubObject("Range()");
    rangeTable1->dynamicCall("SetRange(int, int)", 601, 700);
    QAxObject* table1 = tables->querySubObject("Add(Range, NumRows, NumColumns,
DefaultTableBehavior, AutoFitBehavior)", rangeTable1->asVariant(), 3, 4, 1, 1);
    //Заполняем таблицу
    //горизонтальные заголовки
    QAxObject *currentCell = table1->querySubObject("Cell(Row, Column)", 1, 1);
    QAxObject *rangeCurrentCell = currentCell->querySubObject("Range()");
    rangeCurrentCell->dynamicCall("InsertAfter(Text)", "Рік");

    currentCell = table1->querySubObject("Cell(Row, Column)", 1, 2);
    rangeCurrentCell = currentCell->querySubObject("Range()");
    rangeCurrentCell->dynamicCall("InsertAfter(Text)", "Місяць");

    currentCell = table1->querySubObject("Cell(Row, Column)", 1, 3);
    rangeCurrentCell = currentCell->querySubObject("Range()");
    rangeCurrentCell->dynamicCall("InsertAfter(Text)", "Дохід");

    currentCell = table1->querySubObject("Cell(Row, Column)", 1, 4);
    rangeCurrentCell = currentCell->querySubObject("Range()");
    rangeCurrentCell->dynamicCall("InsertAfter(Text)", "Календарні дні");
```

```
 wordApp->setProperty("Visible", true);
    }
else{
    QAxObject *wordApp = new QAxObject("Word.Application");
    QAxObject *wordDoc = wordApp->querySubObject("Documents()");
    QAxObject *newDoc = wordDoc->querySubObject("Add()");

    //Заголовок для входных данных
    QAxObject *rangeInputData = newDoc->querySubObject("Range()");
    rangeInputData->dynamicCall("SetRange(int, int)", 0, 100);
    rangeInputData->setProperty("Text", "Нарахуванння по лікарняному листу");

    /*размер, шрифт, выравнивание
    QAxObject *font_rangeInputData = rangeInputData->querySubObject("Font");
    font_rangeInputData->setProperty("Size", 12);
    font_rangeInputData->setProperty("Name", "Arial");*/
    QAxObject *alignment_rangeInputData = rangeInputData-
>querySubObject("ParagraphFormat");
    alignment_rangeInputData->setProperty("Alignment", "wdAlignParagraphCenter");
    alignment_rangeInputData->setProperty("SpaceAfter", 0);

    //отступ
    rangeInputData->dynamicCall("InsertParagraphAfter()");
    rangeInputData->dynamicCall("InsertParagraphAfter()");

    QAxObject* prange2 = newDoc->querySubObject("Range()");
   prange2->dynamicCall("SetRange(int, int)",101,200);
   prange2->setProperty("Text","Номер лікарняного листа № \"'+ui->lineEdit->text()+"\'");
   QAxObject *alignment_rangeNameT1 = prange2->querySubObject("ParagraphFormat");
   alignment_rangeNameT1->setProperty("Alignment", "wdAlignParagraphLeft");

   rangeInputData->dynamicCall("InsertParagraphAfter()");
   rangeInputData->dynamicCall("InsertParagraphAfter()");

   QAxObject* prange3 = newDoc->querySubObject("Range()");
```

```
    prange3->dynamicCall("SetRange(int, int)",201,300);
     prange3->setProperty("Text","Працівник: \'"+mn+"\'");

    rangeInputData->dynamicCall("InsertParagraphAfter()");
    rangeInputData->dynamicCall("InsertParagraphAfter()");

    query.exec("SELECT tabln FROM Workers WHERE surname=\'"+mn+"\'");
    query.first();
    QAxObject* prange4 = newDoc->querySubObject("Range()");
   prange4->dynamicCall("SetRange(int, int)",301,400);
    prange4->setProperty("Text","Табельний номер: \'"+query.value(0).toString()+"\'");
    rangeInputData->dynamicCall("InsertParagraphAfter()");
    rangeInputData->dynamicCall("InsertParagraphAfter()");

    QAxObject* prange5 = newDoc->querySubObject("Range()");
   prange5->dynamicCall("SetRange(int, int)",401,500);
    prange5->setProperty("Text","Період роботи: з \'"+ui->dateEdit->text()+"\' по \'"+ui-
>dateEdit_2->text()+"\'");

    rangeInputData->dynamicCall("InsertParagraphAfter()");
    rangeInputData->dynamicCall("InsertParagraphAfter()");
    rangeInputData->dynamicCall("InsertParagraphAfter()");

    //Заголовок для 1 таблицы
    QAxObject *rangeNameT1 = newDoc->querySubObject("Range()");
    rangeNameT1->dynamicCall("SetRange(int, int)",501, 600);
    rangeNameT1->setProperty("Text", "Розрахуання середнього заробітку");
    //выравнивани
    rangeInputData->dynamicCall("InsertParagraphAfter()");
    rangeInputData->dynamicCall("InsertParagraphAfter()");
    //таблица 1
    QAxObject *tables = newDoc->querySubObject("Tables()");
    QAxObject *rangeTable1 = newDoc->querySubObject("Range()");
    rangeTable1->dynamicCall("SetRange(int, int)", 601, 700);
```

```
    QAxObject* table1 = tables->querySubObject("Add(Range, NumRows, NumColumns,
DefaultTableBehavior, AutoFitBehavior)", rangeTable1->asVariant(), 3, 4, 1, 1);
    //Заполняем таблицу
    //горизонтальные заголовки
    QAxObject *currentCell = table1->querySubObject("Cell(Row, Column)", 1, 1);
    QAxObject *rangeCurrentCell = currentCell->querySubObject("Range()");
    rangeCurrentCell->dynamicCall("InsertAfter(Text)", "Рік");

    currentCell = table1->querySubObject("Cell(Row, Column)", 1, 2);
    rangeCurrentCell = currentCell->querySubObject("Range()");
    rangeCurrentCell->dynamicCall("InsertAfter(Text)", "Місяць");

    currentCell = table1->querySubObject("Cell(Row, Column)", 1, 3);
    rangeCurrentCell = currentCell->querySubObject("Range()");
    rangeCurrentCell->dynamicCall("InsertAfter(Text)", "Дохід");

    currentCell = table1->querySubObject("Cell(Row, Column)", 1, 4);
    rangeCurrentCell = currentCell->querySubObject("Range()");
    rangeCurrentCell->dynamicCall("InsertAfter(Text)", "Календарні дні");

 wordApp->setProperty("Visible", true);
}
}

void documents::on_pushButton_clicked()
{
    ui->groupBox->setEnabled(true);
}

void documents::on_table_doubleClicked(const QModelIndex &index)
{
    ui->groupBox->setEnabled(true);
    int a;
    QString b;
    a=ui->table->currentIndex().row();
```

```
ui->lineEdit->setText(ui->table->model()->index(a,0).data().toString());
ui->comboBox->setCurrentText(ui->table->model()->index(a,1).data().toString());
QDate dat;
dat=QDate::fromString(ui->table->model()->index(a,2).data().toString(),"dd.MM.yyyy");
ui->dateEdit->setDate(dat);
dat=QDate::fromString(ui->table->model()->index(a,3).data().toString(),"dd.MM.yyyy");
ui->dateEdit_2->setDate(dat);
}
```

newworker.cpp

```
#include "newworker.h"
#include "ui_newworker.h"
#include "QtSql/QSqlDatabase"
#include "QSqlQuery"
#include "QDebug"
#include "mainwindow.h"
newworker::newworker(QSqlDatabase bz,QWidget *parent) :
   QDialog(parent),
   ui(new Ui::newworker)
{
   ui->setupUi(this);
   addrd=0;
   ui->pushButton->setText("Додати");
   bz.open();
   QSqlQuery query;
   int k=0,i=0;
   QString p;
   do{k++;
      p=QString::number(k);
   query.exec("SELECT * FROM Workers WHERE tabln=\'"+p+"\'");
   if(!query.first()){i=1;}
   }
     while(i==0);
     query.exec("SELECT * FROM Positions GROUP BY position");
    //qDebug()<<query.first();
```

```
      while (query.next()){
        ui->comboBox->addItem(query.value(0).toString());
      }
       ui->lineEdit_2->setText(p);


}
newworker::newworker(QString nm,QSqlDatabase bz,QWidget *parent) :
    QDialog(parent),
    ui(new Ui::newworker)
{
    qDebug()<<nm;
    //
    bz.open();
    ui->setupUi(this);
    bz.open();
    QSqlQuery query;
    ui->pushButton->setText("Редагуати");
    addrd=1;
    query.exec("SELECT * FROM Positions GROUP BY position");
   //qDebug()<<query.first();
   while (query.next()){
     ui->comboBox->addItem(query.value(0).toString());
   }
   query.exec("SELECT pass FROM Info WHERE surname=\'"+nm+"\'");
   query.first();
   ui->lineEdit_3->setText(query.value(0).toString());
    query.exec("SELECT tabln FROM Workers WHERE surname=\'"+nm+"\'");
    query.first();
    ui->lineEdit_2->setText(query.value(0).toString());
    ui->lineEdit->setText(nm);
    query.exec("SELECT position FROM Workers WHERE surname=\'"+nm+"\'");
    query.first();
    ui->comboBox->setCurrentText(query.value(0).toString());
```

```
    if(query.value(0).toString()=="Директор"||query.value(0).toString()=="Бухгалтер"){ui-
>lineEdit_3->setEnabled(true);}
query.exec("SELECT telephon FROM Info WHERE surname=\'"+nm+"\'");
query.first();
ui->lineEdit_4->setText(query.value(0).toString());
query.exec("SELECT adress FROM Info WHERE surname=\'"+nm+"\'");
query.first();
ui->lineEdit_5->setText(query.value(0).toString());
query.exec("SELECT indx FROM Workers WHERE surname=\'"+nm+"\'");
query.first();
ui->lineEdit_7->setText(query.value(0).toString());
query.exec("SELECT prem FROM Workers WHERE surname=\'"+nm+"\'");
query.first();
ui->lineEdit_6->setText(query.value(0).toString());
query.exec("SELECT begin FROM Workers WHERE surname=\'"+nm+"\'");
query.first();
QDate dat;
dat=QDate::fromString(query.value(0).toString(),"dd.MM.yyyy");
ui->dateEdit->setDate(dat);
query.exec("SELECT end FROM Workers WHERE surname=\'"+nm+"\'");
query.first();
dat=QDate::fromString(query.value(0).toString(),"dd.MM.yyyy");
//qDebug()<<dat;
ui->dateEdit_2->setDate(dat);
bz.close();
}

newworker::~newworker()
{
   delete ui;
}

void newworker::on_pushButton_clicked()
{db = QSqlDatabase::addDatabase("QSQLITE");
   db.setDatabaseName("E:\\bazax.db");
```

```
db.open();
QSqlQuery query;
 float pr;

switch(addrd)
{
case 0:{
    pr=ui->lineEdit_6->text().toFloat();
    query.prepare("INSERT INTO Workers (surname,tabln,position,begin,dismas,prem,indx)"
    "VALUES (?, ?, ?, ?, ?, ?, ?);");
     query.addBindValue(ui->lineEdit->text());
    query.addBindValue(ui->lineEdit_2->text());
    query.addBindValue(ui->comboBox->currentText());
    query.addBindValue(ui->dateEdit->text());
    query.addBindValue("");
    query.addBindValue(ui->lineEdit_6->text());
    query.addBindValue(ui->lineEdit_7->text());
     query.exec();

     query.prepare("INSERT INTO Info (surname,adress,telephon,pass)"
     "VALUES (?, ?, ?, ?);");
      query.addBindValue(ui->lineEdit->text());
     query.addBindValue(ui->lineEdit_5->text());
     query.addBindValue(ui->lineEdit_4->text());
     query.addBindValue(ui->lineEdit_3->text());
      query.exec();
       emit buttonClicked();
      this->close();
      //this->deleteLater();
           //MainWindow* w=new MainWindow();
             //        w->show();db.close();
}
case 1:{this->close();
//this->deleteLater();
 //     MainWindow* w=new MainWindow();
```

```
    //              w->obnov();

   }}
}

void newworker::on_comboBox_activated(const QString &arg1)
{
   QString v=ui->comboBox->currentText();
   if(v=="Директор"||v=="Бухгалтер"||v=="Адміністратор") ui->lineEdit_3->setEnabled(true);
      else ui->lineEdit_3->setEnabled(false);
}
```

Лістинг 5— Файл реалізації tabel.cpp

```
#include "tabel.h"
#include "QDebug"
#include <QSqlTableModel>
#include "ui_tabel.h"
#include <QColor>

tabel::tabel(QSqlDatabase db,QWidget *parent) :
   QDialog(parent),
   ui(new Ui::tabel)
{
   ui->setupUi(this);
   db.open();

  selectedDate = QDate::currentDate();
  QDate date(selectedDate.year(), selectedDate.month(), 1);
  selectedDate = QDate(selectedDate.year(), selectedDate.month(), 1);
   QSqlQuery query;
   query.exec("Select surname From Workers");
   query.seek(-1);
      while (query.next()){i++;
```

```
        sp+= query.value(0).toString();
      }
      for (int month = 1; month <= 12; ++month)
       ui->comboBox->addItem(QDate::longMonthName(month));
      ui->comboBox->setCurrentIndex (selectedDate.month() - 1);
      ui->dateTimeEdit->setDateRange (QDate(1918, 1, 1), QDate(9998, 1, 1));
 ui->dateTimeEdit->setDate (selectedDate);
    insert();
}

tabel::~tabel()
{
   delete ui;
}

void tabel::insert()
{
   ui->tableWidget->clear();
   ui->tableWidget->setRowCount(i);
   ui->tableWidget->setVerticalHeaderLabels(sp);
      int lk=selectedDate.daysInMonth();
ui->tableWidget->setColumnCount(lk+1);
QTableWidgetItem *hnm_1 = new QTableWidgetItem();
   hnm_1->setText("Всього");
   ui->tableWidget->setHorizontalHeaderItem(lk,hnm_1);
int weekDay = selectedDate.dayOfWeek();

ui->tableWidget->horizontalHeader()->setSectionResizeMode(QHeaderView::Stretch);

//ui->tableWidget->item(1,1)->setBackgroundColor(0xFF00FF00);
/*QAbstractItemModel* m;
ui->tableWidget->model();
m->setData( m->index( 1, 1 ), QBrush( Qt::red ), Qt::BackgroundRole );*/

//ui->tableWidget->setColumnWidth(lk,120);
```

```
/*QComboBox *box=new QComboBox;
foreach(QString index, values.keys()),
   box->addItem(values.value(index), index);
box->setSizePolicy(QSizePolicy::Expanding, QSizePolicy::Preferred);
ui->tableWidget->setCellWidget(ui->tableWidget->rowCount() - 1, 5, box);
*/

}

void payment::on_pushButton_clicked()
{
   QSqlQuery query;
   QString v,b,c;
int kil,i;
double av,avr,oh,hi,okl,na,pr,hs,fr,shs=0,sfr=0,w,zar;
 kil=ui->table4->model()->rowCount();
//qDebug()<<kil;
for(i=0;i<kil;i++)
{
v=ui->table4->model()->index(i,1).data().toString();
//qDebug()<<v;
query.exec("SELECT start,norm FROM workplaces WHERE position = \'"+v+"\'");
query.first();
av=query.value(0).toDouble();
hi=query.value(1).toDouble();

oh=ui->table4->model()->index(i,4).data().toDouble();

if(oh<hi/2) {avr=0;
   query.exec("UPDATE info SET avans = NULL WHERE nomer = \'"+ui->table4->model()-
>index(i,3).data().toString()+"\'");}
else{avr=av*0.5;
   b=QString::number(avr);
```

```
   query.exec("UPDATE info SET avans =\'"+b+"\' WHERE nomer = \'"+ui->table4->model()-
>index(i,3).data().toString()+"\'");
}
okl=(av/hi)*oh;
//qDebug()<<okl;
query.exec("SELECT addm FROM workers WHERE id=\'"+ui->table4->model()-
>index(i,0).data().toString()+"\'");
query.first();
na = query.value(0).toDouble();
na=okl*(na/100);
//qDebug()<<na;
query.exec("SELECT premium FROM workers WHERE id=\'"+ui->table4->model()-
>index(i,0).data().toString()+"\'");
query.first();
pr = query.value(0).toDouble();
pr=(okl+na)*(pr/100);
//qDebug()<<pr;
hs=ui->table4->model()->index(i,5).data().toDouble();
fr=ui->table4->model()->index(i,6).data().toDouble();
query.exec("SELECT * FROM month WHERE id=\'"+ui->table4->model()-
>index(i,0).data().toString()+"\' AND nomer < \'"+ui->table4->model()-
>index(i,3).data().toString()+"\'");
if(!query.first()){
   if(hs!=0.0)  shs=(hi/30.44)*hs;
   if(fr!=0.0)  sfr=(hi/30)*hs;
}
else{   query.seek(-1);
while (query.next()){
   shs += query.value(4).toDouble();}
if(hs!=0.0)  sfr=(shs/365)*hs;
if(fr!=0.0)  shs=(shs/354)*fr;
else shs=0;
}
//qDebug()<<shs;
//qDebug()<<sfr;
```

```
w=okl+na+pr+shs+sfr;
qDebug()<<w;
c=QString::number(w);
//qDebug()<<c;
query.exec("UPDATE month SET allm =\"'+c+"\' WHERE nomer = \"'+ui->table4->model()-
>index(i,3).data().toString()+"\'");
zar=w-(w*0.205);//18% 1,5% %1
qDebug()<<av;
zar=zar-avr;
qDebug()<<zar;
c=QString::number(zar);
query.exec("UPDATE info SET salary =\"'+c+"\' WHERE nomer = \"'+ui->table4->model()-
>index(i,3).data().toString()+"\'");
}
model->setQuery(QSqlQuery(("SELECT workers.id as 'Табел. номер', workers.position as
'Посада', workers.name as 'П.І.Б.', info.nomer as '№ місяця', info.hour as 'Відпрац. годин',
info.hospital as 'Лікарняні', info.free as 'Відпускні', info.avans as 'Аванс', info.salary as
'Зарплатня' FROM workers,month,info WHERE month.id=workers.id AND
month.nomer=info.nomer GROUP BY month.nomer"), db));
ui->table4->verticalHeader()->setVisible(false);
ui->table4->setModel(model);
ui->table4->show();
ui->comboBox->setCurrentIndex(0);
}

void tabel::on_comboBox_currentIndexChanged(int index)
{
}

void tabel::on_dateTimeEdit_dateChanged(const QDate &date)
{

}

void tabel::on_comboBox_activated(int index)
```

```
{

    selectedDate = QDate(selectedDate.year(), index+1, 1);
    qDebug()<<selectedDate.daysInMonth();
    qDebug()<<selectedDate;
    insert();
}
```

**ДОДАТОК Б**

**ВІДГУК КЕРІВНИКА ЕКОНОМІЧНОГО РОЗДІЛУ**

**ДОДАТОК В**

## ПЕРЕЛІК ФАЙЛІВ НА ДИСКУ

| **Ім’я файлу** | **Опис** |
|---|---|
| Пояснювальні документи | |
| Диплом_ doc | Пояснювальна записка до кваліфікаційної роботи. Документ Word. |
| Диплом_.pdf | Пояснювальна записка до кваліфікаційної роботи в форматі PDF |
| Програма | |
| Program.rar | Архів. Містить коди програми і откомпільовану програму |
| Презентація | |
| Презентація_.ppt | Презентація кваліфікаційної роботи |