

**Національний технічний університет України
«Київський політехнічний інститут
імені Ігоря Сікорського»**

Теплоенергетичний факультет

**Кафедра автоматизації проектування
енергетичних процесів і систем**

КУРСОВА РОБОТА

з курсу «Бази даних»

**Тема : Інформаційна система «Відділ бухгалтерського
обліку. Облік відряджень»**

Керівник Колумбет Вадим Петрович	Виконав Черноусов Денис Ігорович
Допущено до захисту	Студент 2-го курсу
«25» грудня 2020 р.	Групи ТІ-92
Захищено з оцінкою	Залікова книжка
97	№ ТІ-9253

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМ. ІГОРЯ
СІКОРСЬКОГО»

Кафедра автоматизації проектування енергетичних процесів і систем

КУРСОВА РОБОТА

з дисципліни «Бази даних»

(назва дисципліни)

на тему: Інформаційна система

«Відділ бухгалтерського обліку. Облік відряджень»

Студента групи ТІ-92

напряму підготовки 6.050103

спеціальності Програмна інженерія

Черноусов Д.І.

(прізвище та ініціали)

Керівник ст. викладач Колумбет В.П.

(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Національна оцінка:

Кількість балів:

Члени комісії

(підпис)

(вчене звання, науковий ступінь, прізвище та ініціали)

(підпис)

(вчене звання, науковий ступінь, прізвище та ініціали)

(підпис)

(вчене звання, науковий ступінь, прізвище та ініціали)

Київ- 2020

**Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»**

Факультет: Теплоенергетичний факультет

(повна назва)

Кафедра: Автоматизації проектування енергетичних процесів і систем

(повна назва)

Освітньо-кваліфікаційний рівень: бакалавр

Напрямок підготовки 121 “Інженерія програмного забезпечення”

(шифр і назва)

**ЗАВДАННЯ
НА КУРСОВУ РОБОТУ СТУДЕНТА**

Черноусов Денис Ігорович

(прізвище, ім'я, по-батькові)

1. Тема роботи: Інформаційна система «Відділ бухгалтерського обліку. Облік відряджень»

Керівник проекту(роботи): Колумбет Вадим Петрович ст. викладач

(прізвище, ім'я, по-батькові, науковий ступінь, вчене звання)

2. Срок подання студентом роботи: «27» грудня 2020 р.

3. Вихідні дані до проекту(роботи): мова програмування Java, середовище розробки IntelliJ IDEA, система керування базами даних – MySQL.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) розробити інформаційну систему «Відділ бухгалтерського обліку. Облік відряджень», спроектувати базу даних.

5. Дата видачі завдання «19» вересня 2020 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назви етапів виконання дипломного проекту(роботи)	Строк виконання етапів проекту(роботи)	Примітка
1	Затвердження теми роботи	19.09.2020	
2	Вивчення та аналіз задачі	22.09.2020	
3	Проектування структури БД	23.10.2020	
4	Розробка алгоритму реалізації задачі	03.11.2020	
5	Програмна реалізація задачі	5.12.2020	
6	Тестування програми	14.12.2020	
7	Оформлення пояснювальної записки	26.12.2020	

Студент _____

Керівник проекту (роботи) _____

Анотація

У курсовій роботі були розглянута програма для роботи з обліком відряджень. Вона називається «BlackMain», створена за допомогою мови програмування Java, а її база даних - реляційної системи управління базами даних Mysql. Обрані найбільш оптимальні методи та алгоритми для реалізації логіки програми, зокрема використовуються можливості бібліотек JavaFX та Swing.

Суть програми полягає в автоматизації обліку відряджень. Витрати коштів на відрядження вираховуються автоматично. Також програма зручна для зберігання, редагування та внесення даних.

Обсяг усієї пояснювальної записки 36 аркушів, кількість ілюстрацій – 35, один додаток.

ЗМІСТ

Вступ.....	6
1 Опис предметної області	7
2 Модель бази даних	9
2.1 Концептуальна модель бази даних.....	9
2.2 Фізична модель бази даних	10
2.3 Структура таблиць баз даних.....	11
2.4 Приклади заповнення таблиць.....	14
3 Опис програмної реалізації	19
3.1 Використане програмне забезпечення.....	19
3.2 Структура інформаційної системи	19
4 Опис інтерфейсу користувача.....	24
Висновки	29
Список використаних джерел	30
ДОДАТОК Лістинг програмного модуля	31

Вступ

Кожне підприємство прагне до збільшення прогресу результатів своєї діяльності. Одним із найефективніших способів нарощення впливу, збільшення кількості ринків збуту та поліпшення професійних здібностей робітників є відрядження. Відрядження – це поїздка працівника на певний термін в конкретне місце з метою виконання службового доручення, виданого керівником підприємства.

Метою даної роботи є створення програмного забезпечення направлено на роботу з обліком службових відряджень. Це програмне забезпечення має надавати такі можливості користувачу:

1. Створення нових і редагування старих даних про робітників, ордери та завдання, що отримують ці робітники, а також дані про міста, їх типи та ціна проживання в день за фіксованими тарифами.
2. Коректне видалення даних з бази даних. При видаленні міста мають видалятися пов'язані з цим містом завдання, а робітники повинні втрачати ордери на виконання цього завдання.
3. Зручна пошукова система. Ця система повинна виводити потрібні дані орієнтуючись на вказану у фільтрі назву стовпчика або по всій базі даних
4. Система має бути автоматизована, тобто програма сама повинна вираховувати кількість днів поїздки та суму грошей, що підприємство має видати, орієнтуючись на інші дані з інших таблиць.
5. Захист даних. Програма повинна бути захищена принаймні авторизованим входом.
6. Інші операції для зручної роботи користувача у даній прикладній області

Програми для оформлення відряджень, вочевидь, уже існують. Найпопулярніші серед них це Smartway, Zoho Expense, Saas. Оскільки, створення системи для роботи з обліком службових відряджень не є складним, то такі

авіакомпанії як Ozon і Aviasales містять на своїх сайтах такий функціонал. Потрібно лише зареєструватися, аби отримати можливість наперед замовляти білети, місця в готелях, а також вираховувати на всі ці операції витрати.

Основною характеристикою даного типу продуктів має бути можливість вмістити якомога більше даних про робітників, їхні завдання та витрати. Швидкість виконання процесів суттєво не впливає на ефективність програмного забезпечення, важливішими є безпека даних та точність обчислень. Також не можна нехтувати інтерфейсом програми. Він має бути інтуїтивно зрозумілим для бухгалтера.

1. Опис предметної області

На підставі наказу кожному співробітнику, відправленому у відрядження, виписується посвідчення про відрядження. Посвідчення ідентифікується номером і датою видачі. У ньому також вказується П.І.Б. відрядженого, займана посада, пункти призначення (їх може бути не більше двох) і термін відрядження (дата початку та дата закінчення).

На підставі посвідчення про відрядження відділом бухгалтерського обліку проводиться попередній розрахунок витрат для видачі авансу. Попередній розрахунок ведеться за двома статтями:

- 1) проїзд;
- 2) квартирні.

Для розрахунку проїзду беруться дані з купейного квитка.

Сума квартирних розраховується виходячи з кількості днів проживання в місті і його категорії:

- в столиці - 1000 грн. на добу;
- в звичайному місті - 250 грн. на добу;
- районний центр - 500 грн. на добу.

Робітник отримує розрахункову суму авансу в касі бухгалтерії за касовим ордером. У ньому міститься номер ордера, П.І.Б., сума, дата отримання грошей.

Після повернення з відрядження працівник складає авансовий звіт, в якому вказує фактично витрачені суми на проїзд, бронювання місць, проживання в готелі, телефонні переговори, суму добових у розрахунку на фактичну кількість днів.

2. Модель бази даних

2.1 Концептуальна модель бази даних

Дана концептуальна модель бази даних представляє опис основних таблиць та зв'язків між ними.

На рисунку 2.1 подано концептуальну модель розробленої бази даних відділу бухгалтерського обліку, а саме обліку відряджень.

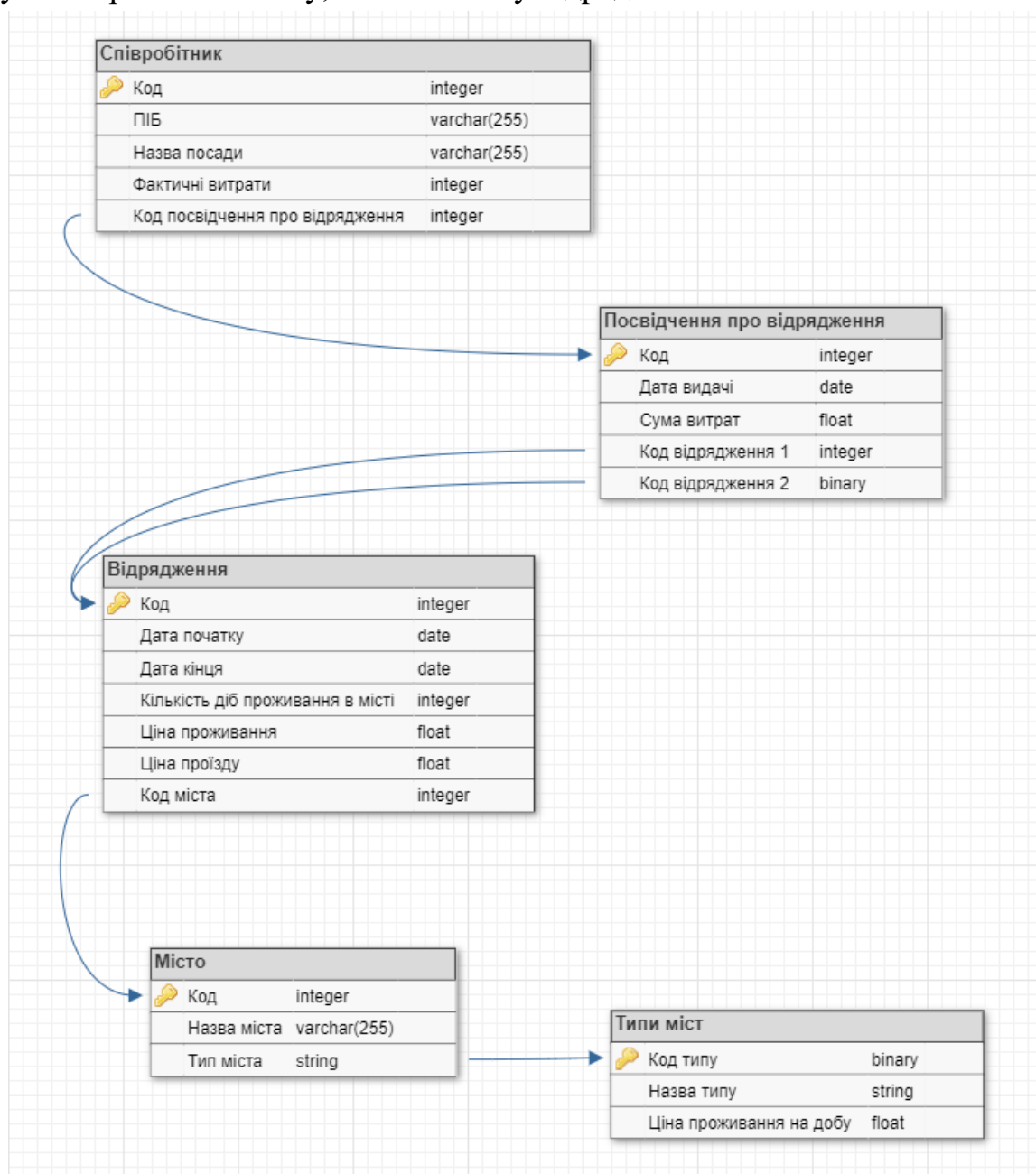


Рисунок 2.1 Концептуальна модель бази даних

2.2 Фізична модель бази даних

Дана фізична модель даних описує реалізацію об'єктів логічної моделі на рівні об'єктів конкретної бази даних.

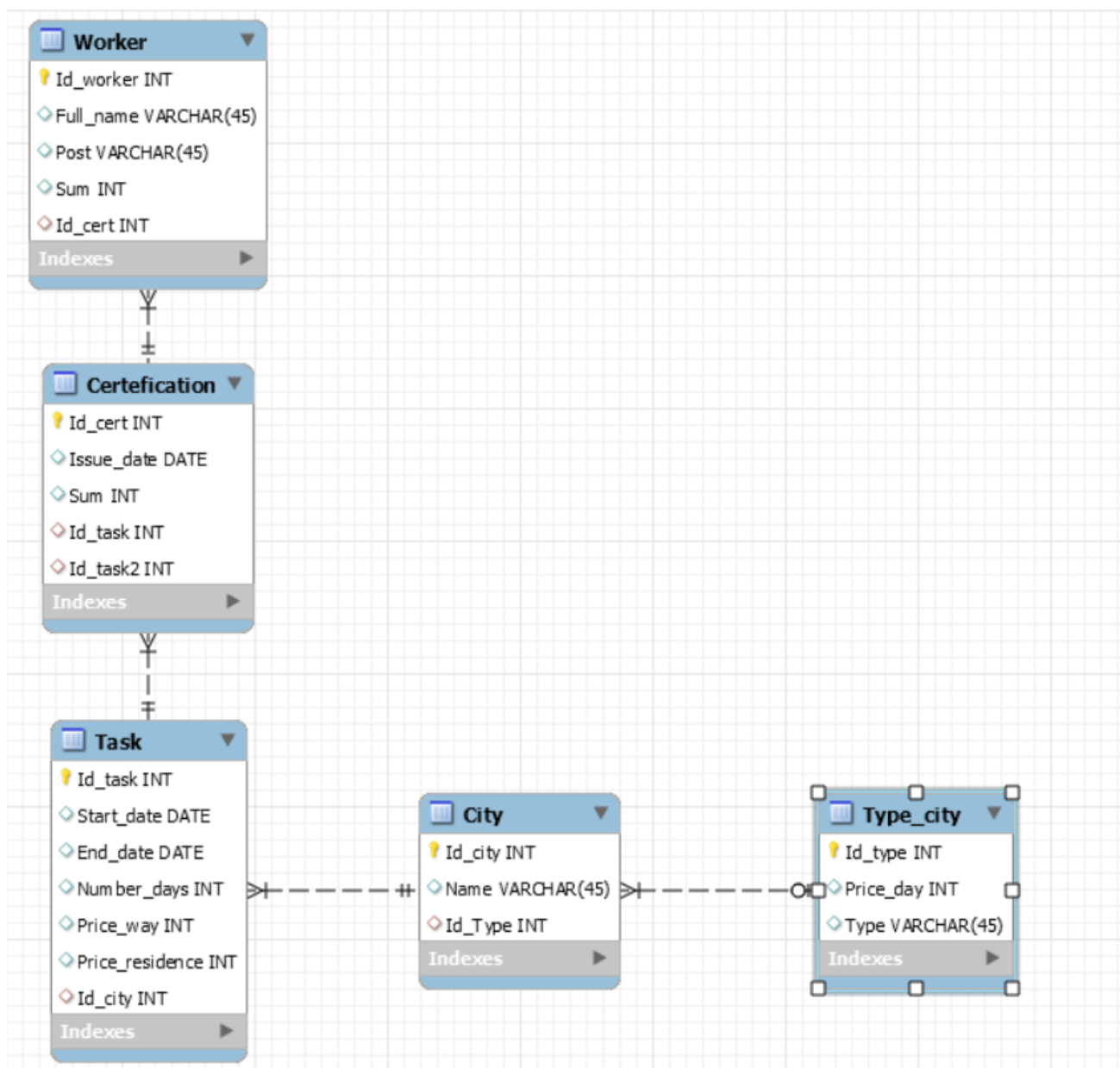


Рисунок **Error! No text of specified style in document..2** Фізична модель БД

У базі даних було створено 5 таблиць. Кожна таблиця містить свій специфічний набір даних.

Таблиця Worker містить інформацію про робітника і є основною таблицею системи. Вона зв'язана з таблицею Certefication відношеннями один до одного, оскільки один робітник може отримати один ордер на відрядження. Згідно ордеру про відрядження робітник може поїхати на два відрядження, тобто таблиці Certefication і Task зв'язані один до двох .

Таблиця City описує можливі міста для відрядження. Зв'язок між Task і City – один до одного. При цьому існують 3 типи міст. Місто може бути великим, середнім та малим. Дані про них внесені в таблицю Type_City, який має зв'язок з City один до одного.

2.3 Структура таблиць баз даних

Таблиця Worker складається з п'яти полів, які зображені на рисунку 2.3. Перше поле - це поле унікального коду робітника Id_worker. Воно є первинним ключем таблиці (Primary key). Друге поле називається Full_name – воно відповідає за ім'я робітника. Поле Post зберігає назву посади робітника в компанії. Поле Sum відповідає за фактичні витрати коштів співробітником під час відрядження. Якщо робітник не їде у відрядження то це поле і наступне поле набувають значення null.

Полю Id_cert задається обмеження Foreign key. Тобто Id_cert є зовнішнім ключем, що зв'язує таблицю Worker з таблицею Certefication. Таблиця Worker має такі обмеження зовнішнього ключа ON DELETE SET NULL ON UPDATE SET NULL. Це означає, що при видаленні сертифікату, на який посилається робітник, значення поля Id_cert стане null.

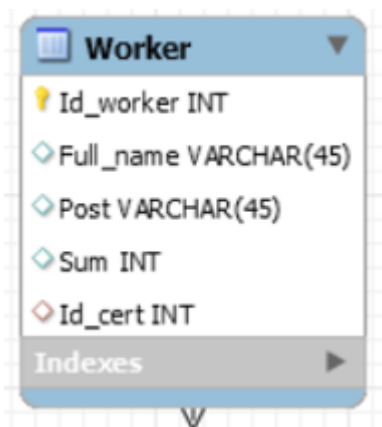


Рисунок 2.3 Таблиця Worker з полями

Значення числа Id_cert таблиці Worker відповідає значенню поля Id_cert, що є в таблиці Certefication первинним ключем. Ця таблиця зображена на рисунку 2.4. Issue_date – це поле типу Date, воно відповідає даті отримання ордеру, значення цього поля отримуємо віднявши 7 днів від початку першого відрядження. Поле Sum – це сума усіх грошових витрат на виконання завдань. Ці витрати є арифметично обчисленими за допомогою значень інших полів, інших таблиць.

Таким два полям Id_task та Id_task2 задано обмеження Foreign key. Кожне з полів зв'язується з таблицею Task. При цьому, встановлено обмеження ON DELETE CASCADE ON UPDATE CASCADE. Це означає, що при видаленні, відрядження з певним кодом, ордер, який містить в собі це завдання, теж видалиться .

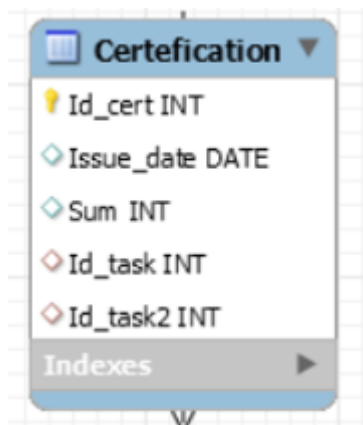


Рисунок 2.4 Таблиця Certefication з полями

Найбільш масивною є таблиця Task, що зображена на рисунку 2.5. Поле Id_task є первинним ключем. Поля Start_date і End_date зберігають відповідно дату початку відрядження і дату його завершення. Поле Number_days набуває тільки додатнього значення при відніманні від дати кінця відрядження дату його початку. Таким чином знаходиться кількість днів, скільки робітник має провести у відрядженні. Поле Price_way вміщує дані про кількість грошей, які треба виділити робітнику для прибуття до місця призначення і повернення. Поле Price_residence вираховується як добуток кількості днів на кількість грошей, які треба витрати, проживаючи в конкретному місті. Тобто поле Price_residence зберігає дані про кількість грошей, які треба заплатити за проживання.

Поле Id_city є зовнішнім ключем і має такі обмеження ON DELETE CASCADE ON UPDATE CASCADE.

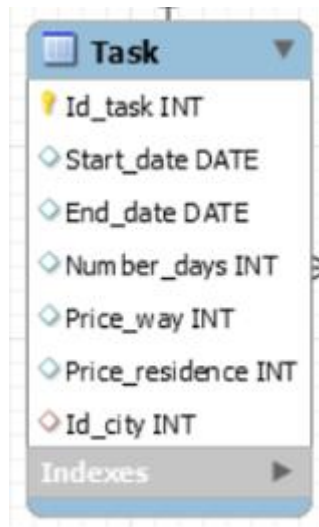


Рисунок 2.5 Таблиця Task з полями

Таблиця City (рисунок 2.6) складається з трьох полів : первинного ключа Id_city, що зв'язаний з зовнішнім ключем Id_city таблиці Task, звичайного поля Name, який містить рядок, назву міста. Полю Id_type задані обмеження foreign key ON DELETE CASCADE ON UPDATE CASCADE.

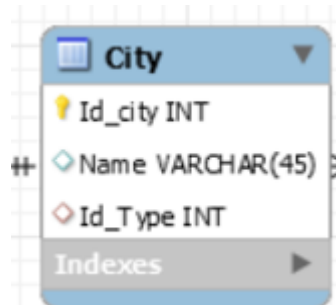


Рисунок 2.6 Таблиця City з полями

Таблиця Type_city має 3 поля (рисунок 2.7) . Перше поле Id_type задано обмеження Primary key. При цьому, значень Id_type за умовою завдання може бути лише три, тобто є тільки 3 типи міст. Поле Type вміщує в собі назву одного з трьох можливих типів, мається на увазі значення Big city, або Average city, або Little city. З поля Price_day можна отримати дані про витрати на проживання в місті за один день.

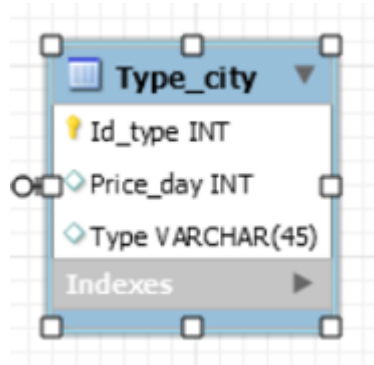


Рисунок 2.6 Таблиця Type_city з полями

Що важливо відмітити : значення Primary key кожної з таблиць має бути строго більше нуля.

2.4 Приклади заповнення таблиць

Першу таблицю Worker заповнюємо такими даними про робітників : код робітника(Id_worker), ім'я робітника(Full_name), його посада(Post), його фактичні витрати під час відрядження(Spent), код ордеру про відрядження(Id_Cert). Заповнюємо усі таблиці за допомогою команди insert into.

```

✓ insert into mydb.Worker (Id_Worker, Full_name, Post, Spent, Id_Cert)
values (1, 'Vasya Dancer', 'teacher', 12000, 1),
      (2, 'Arthur Lancer', 'manager', 3000, 2),
      (3, 'Arthur Ginger', 'programmer', 3000, 2),
      (4, 'George Soros', 'programmer', 9000, 2),
      (5, 'Anton Truba', 'cleaner', NULL, NULL),
      (6, 'Olia Dancer', 'teacher', 10000, 1),
      (7, 'Uther Octopus', 'programmer', 10000, 2),
      (8, 'Klenovich Dub', 'cleaner', NULL, NULL);

```

Рисунок 2.7 заповнення даними таблиці Worker

Використовуючи команду select, виведемо всі дані про таблицю Worker. Результат на рисунку 2.8.

	Id_Worker	Full_Name	Post	Spent	Id_Cert
1	1	Vasya Dancer	teacher	12000	1
2	2	Arthur Lancer	manager	3000	2
3	3	Arthur Ginger	programmer	3000	2
4	4	George Soros	programmer	9000	2
5	5	Anton Truba	cleaner	<null>	<null>
6	6	Olia Dancer	teacher	10000	1
7	7	Uther Octopus	programmer	10000	2
8	8	Klenovich Dub	cleaner	<null>	<null>

Рисунок 2.8 дані в таблиці Worker

Табличку Certification заповнюємо даними за порядком :

- 1) Код сертифікату (Id_cert)
- 2) Код першого і другого завдання (Id_task1 і Id_task2)
- 3) Дата видачі ордера (Issue_date) визначається як менша з дат початку відрядження відняти сім днів.
- 4) Сума витрат (sum) – сума витрат на проїзд та витрат на проживання в місці для кожного завдання.

```

✓ insert into mydb.Certification (Id_Cert, Id_task1, Id_Task2, Issue_date, sum)
values (1, 1, 2,
      least((select (Start_date - interval 7 day)
              from mydb.Task
              where Id_task = 1), (select (Start_date - interval 7 day)
                                   from mydb.Task
                                   where Id_task = 2)
      ),
      ((select Price_way + Price_residence
        from mydb.Task
        where Id_task = 1)
      ) + (select Price_way + Price_residence
          from mydb.Task
          where Id_task = 2)),
      (2, 2, null,
      least((select (Start_date - interval 7 day)
              from mydb.Task
              where Id_task = 2), '3000-01-01'),
      (select Price_way + Price_residence
        from mydb.Task
        where Id_task = 2));

```

Рисунок 2.9 заповнення даними таблиці Certification

Виведенні дані з цієї таблиці матимуть ось такий вигляд.

	Id_Cert	Issue_Date	Sum	Id_Task1	Id_Task2
1	1	2019-12-26	8500	1	2
2	2	2020-02-20	4500	2	<null>

Рисунок 2.10 дані в таблиці Certification

Таблицю Task заповнюємо даними :

- 1) Код відрядження (Id_task)
- 2) Дата початку й кінця відрядження (Start_date і End_date)
- 3) Кількість грошей, які мають бути витраченими на поїдку(Price_way).
Вводиться довільне значення більше нуля.
- 4) Id_city – код міста, в яке їде співробітник на відрядження.
- 5) Поле Number_days заповнюється автоматично. Воно отримує значення від різниці кінцевої дати і початкової дати відрядження.
- 6) Ціна за проживання (Price_residence) визначається як добуток кількості днів проведених у відрядженні на ціну проживання в місті на один день.

```

insert into mydb.Task (Id_Task ,Start_date, End_date, Price_way, Id_city, Price_residence)
values (1, '2020-01-02', '2020-01-05', 1000, 1,
        (datediff(End_date, Start_date) * (
            select Price_day
            from mydb.Type_city
            where Id_type = (select Id_type
                            from mydb.City
                            where Id_city = 1))),
        (2, '2020-02-27', '2020-03-03', 2000, 2,
        (datediff(End_date, Start_date) * (
            select Price_day
            from mydb.Type_city
            where Id_type = (select Id_type
                            from mydb.City
                            where Id_city = 2))));

```

Рисунок 2.11 заповнення даними таблиці Task

Відповідно результат введення даних отримаємо через select

	Id_Task	Start_Date	End_Date	Number_Days	Price_Way	Price_Residence	Id_City
1	1	2020-01-02	2020-01-05	3	1000	3000	1
2	2	2020-02-27	2020-03-03	5	2000	2500	2

Рисунок 2.12 дані в таблиці Task

В таблицку City вставляємо дані про місто :

- 1) Код міста (Id_city)
- 2) Назва міста (Name)
- 3) Код типу міста(Id_type).

```
✓ insert into mydb.City (Id_City , `Name` , Id_type)
values (1, 'Kyiv' , 1),
       (2, 'Lviv' , 2),
       (3, 'White Church' , 3),
       (4, 'Ivanovo' , 3),
       (5, 'Harkiv' , 2);
```

Рисунок 2.13 заповнення даними таблиці City

Таблиця з містами зображена на рисунку 2.14

	Id_City	Name	Id_Type
1	1	Kyiv	1
2	2	Lviv	2
3	3	White Church	3
4	4	Ivanovo	3
5	5	Harkiv	2

Рисунок 2.14 дані в таблиці City

Вставляємо дані в таблицю Type_city :

- 1) Код типу міста, а їх може бути за умовою лише 3
- 2) Назва типу міста
- 3) Кількість грошей, що треба сплатити за день проживання в такому місті

```
✓ insert into mydb.type_city
values (1, 'Big city' , 1000),
       (2, 'Average city' , 500),
       (3, 'Little city' , 250);
```

Рисунок 2.15 заповнення даними таблиці Type_city

Відповідно, виводимо дані про ці три типи міст.

	🔑 Id_Type ▾	📄 Type ▾	📄 Price_day ▾
1	1	Big city	1000
2	2	Average city	500
3	3	Little city	250

Рисунок 2.16 дані в таблиці Type_city

3 Опис програмної реалізації

3.1 Використане програмне забезпечення

Для створення інформаційної системи «Відділ бухгалтерського обліку. Облік відряджень» була використані мова програмування java та система керування базами даних Mysql.

Середовищем для програмування java коду було вибрано IntelliJ Idea. Для створення функціоналу програми використали дві бібліотеки JavaFX та Swing. За допомогою цих двох бібліотек було створено таблиці, кнопки, вкладки, поля та вікна. Для роботи з fxml кодом, тобто з кодом графічної складової програми, була вибрана програма Scene Builder.

Для створення концептуальної моделі бази даних був використаний функціонал сайту dbdesigner.net. І звідти взята фізична модель бази даних. Для роботи з Mysql було вибране середовище DataGrip.

3.2 Структура інформаційної системи

Вхідною точкою в програму є файл Main. У цьому файлі задаються налаштування для першого вікна – вікна авторизації в програмі, і здійснюється перехід на це вікно.

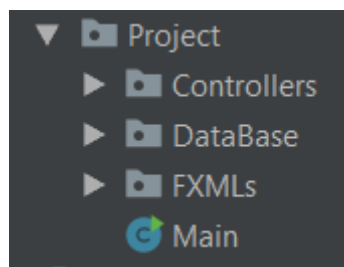


Рисунок 3.1 Файл Main серед пакетів

У пакеті FXMLs розташована графічна оболонка програми. Два файли-вікна розташовані у пакеті Stages. StageEntrance – вікно входу, іншими словами, вікно авторизації у програму. StageMenu – вікно головного меню.

У пакеті Tabs знаходяться 4 вкладки, доступ до яких можна отримати знаходячись у меню. На вкладці TabCert знаходиться таблиця ордерів і функціонал для редагування та пошуку даних у цій таблиці. На вкладці TabDic знаходяться одна таблички, з'єднана в одну. Вона складається з таблиці міст та типів міст. На вкладці також присутні методи редагування даних та фільтр. Вкладки TabTask відповідає таблиця відряджень, TabWorkers – таблиця робітників.

Файл Ico.png – це іконка для програми. Файл Sheet.css коректує зовнішній вигляд декількох об'єктів, створених файлах типу fxml.

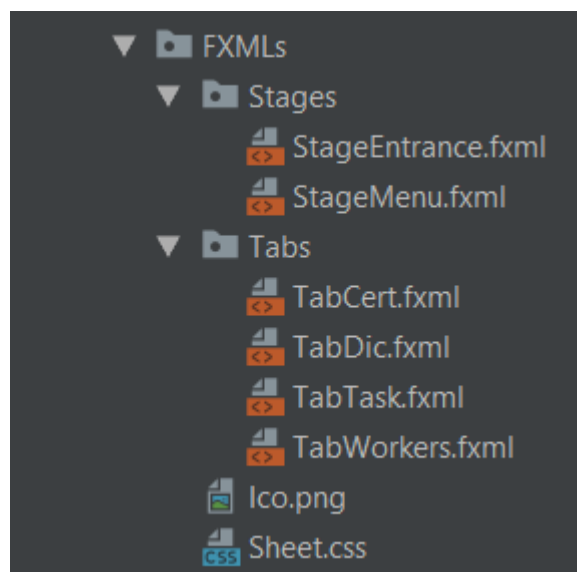


Рисунок 3.2 Пакет FXMLs

У пакеті Controllers знаходяться файли, що відповідають за вивід, редагування, видалення даних з таблиць. Клас ControllerEntrance містить в собі засоби взаємодії з вікном входу. З цього вікна можна перейти у вікно меню, функціонал якого описаний в класі ControllerMenu.

Меню містить в собі 4 вкладки, перехід між якими забезпечений ControllerMenu. Функціонал цих чотирьох вкладок описаний у пакеті TableControllers, а саме у файлах ControllerCert, ControllerDic, ControllerTask, ControllerWorkers. Кожна вкладка містить в собі одну таблицю : Certification, таблиця City і Type_city, Task, Workers.

У пакеті Tables описані класи рядів таблиць, тобто класи які мають певні типи даних для заповнення конкретних таблиць. Кожна таблиця складається з об'єктів свого класу, наприклад, для класу worker – це таблиця Workers , cert – Certification, task – Task , а для таблиць City і Type_city створений один клас City.

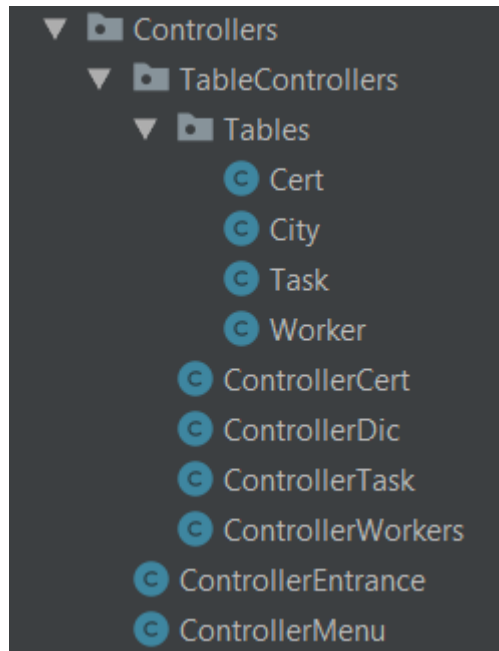


Рисунок 3.3 Пакет Controllers

Пакет DataBase створений для взаємодії з базою даних на локальному сервері. Зв'язок встановлюється через файл DConnector, а у файлі DHandler записані шаблони запитів і команд для взаємодії з базою даних.

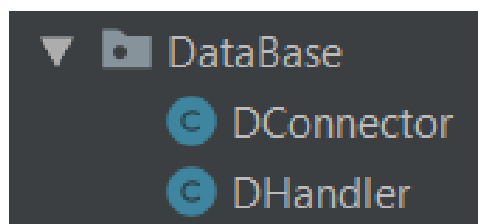


Рисунок 3.4 Пакет DataBase

Увесь проект складається з семи папок та двадцяти одного файлу, серед яких тринадцять – файли-класи. І виглядає проект ось так (див. рисунок 3.5)

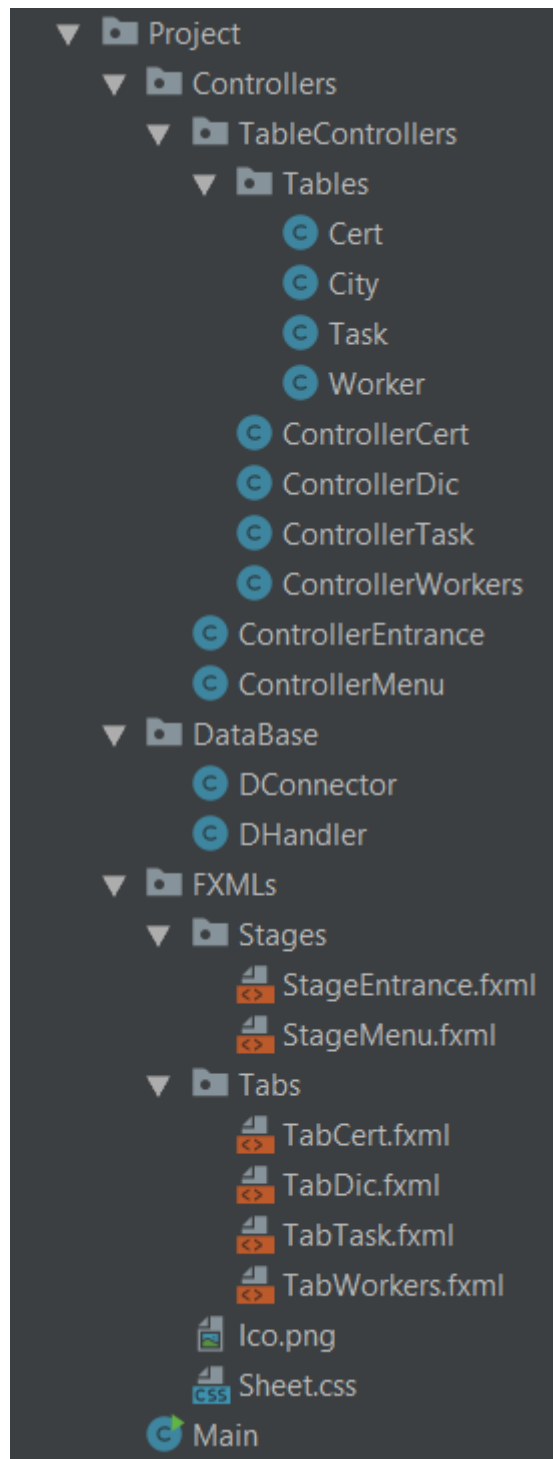


Рисунок 3.5 Увесь проект

На рисунку 3.6 зображена діаграма класів, які в змозі маніпулювати даними з бази даних та відображати ці дані у програмі. Показане відношення між класом ControllerMenu та контролерами ControllerDic, ControllerCert, ControllerTask, ControllerWorkers. Також показані зв'язок між контролером таблиці та рядком (елементом, об'єктом) таблиці.

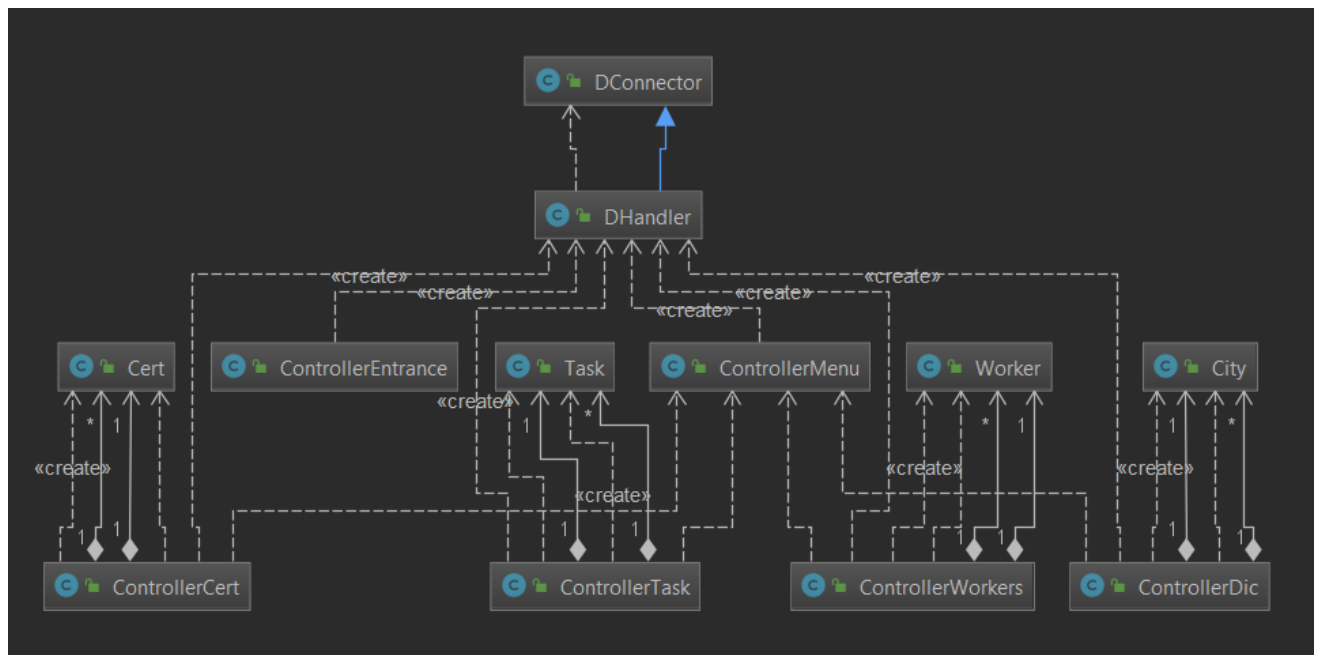


Рисунок 3.6 UML діаграма класів

4 Опис інтерфейсу користувача

При вході в програму впливає вікно авторизації. Для входу в систему потребується ввести правильні логін, пароль та натиснути кнопку Sign in.

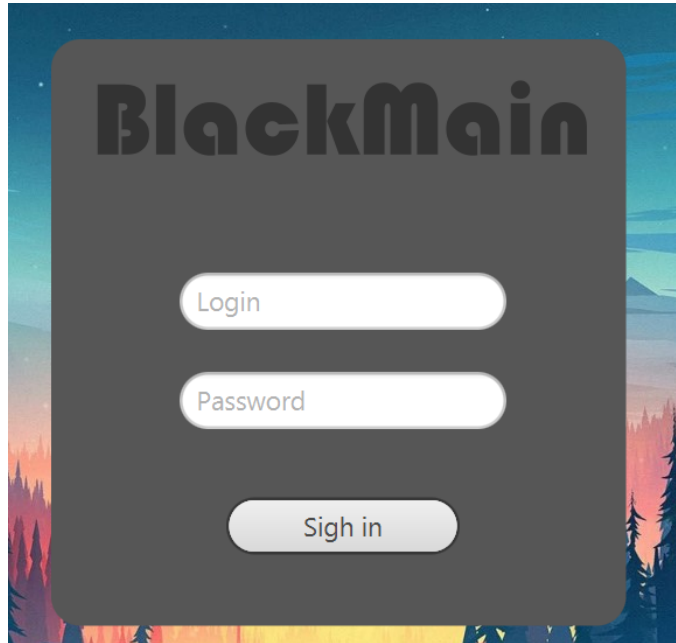


Рисунок 4.1 вхід у систему

При неправильно введеному паролі або логіну, поля Login та Password підсвітяться червоним кольором. Якщо ввести правильні логін та пароль, то відкриється вікно меню.

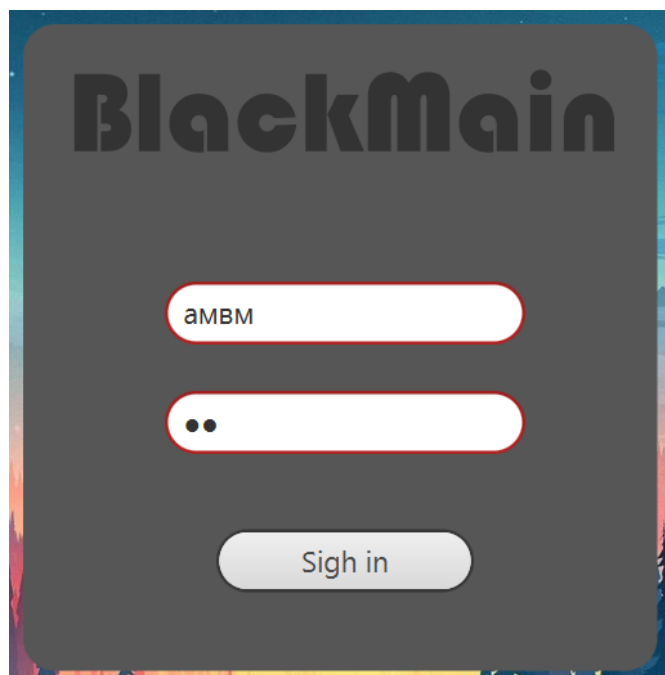


Рисунок 4.2 не правильно введені дані

Перейшовши до меню, можна побачити 4 вкладки, за замовченням відкривається перша вкладка - Workers. На ній знаходиться таблиця робітників, яка пов'язана з таблицею Worker, знаходиться у базі даних. Відповідно, інші вкладки працюють по такій же ідеї.

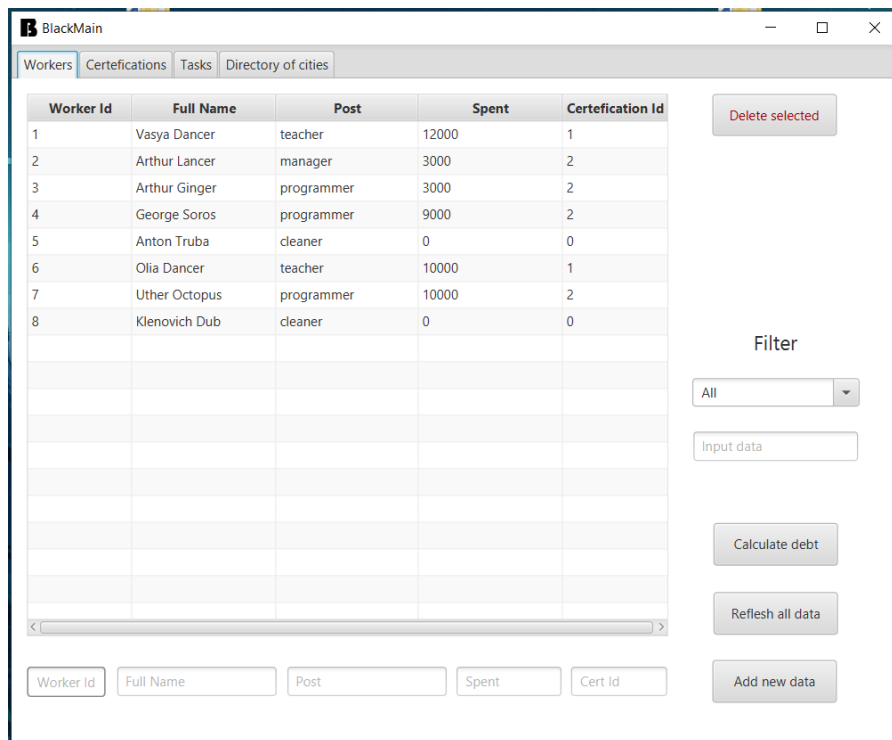


Рисунок 4.3 Вкладка Workers

Наймасивнішим об'єктом на сторінці є таблиця. Вона відображає дані, що отримала від бази даних. При будь-яких змінах цих даних цієї таблиці, відповідні дані на локальному сервері також міняються, але це не стосується фільтру, він міняє лише вигляд таблиці, а дані зберігаються.

Worker Id	Full Name	Post	Spent	Certefication Id
1	Vasya Dancer	teacher	12000	1
2	Arthur Lancer	manager	3000	2
3	Arthur Ginger	programmer	3000	2
4	George Soros	programmer	9000	2
5	Anton Truba	cleaner	0	0
6	Olia Dancer	teacher	10000	1
7	Uther Octopus	programmer	10000	2
8	Klenovich Dub	cleaner	0	0

Рисунок 4.4 Таблиця з робітниками

Розглянемо функціонал. Кнопка Delete selected дозволяє видалити з таблиці, відповідно з бази даних, один рядок. Ця кнопка знаходиться у верхньому правому куті.

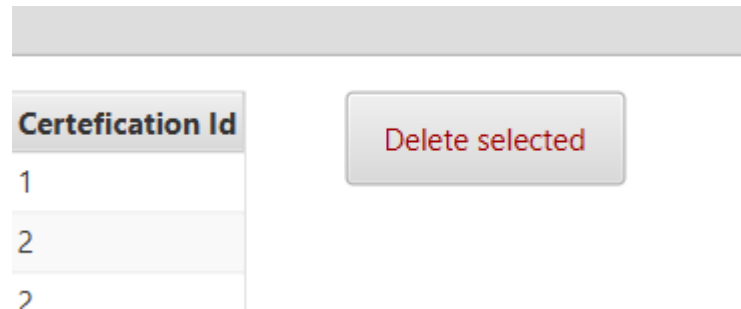


Рисунок 4.5 Кнопка видалення даних

Нижче цієї кнопки знаходиться фільтр. При введенні даних у поле, таблиця з даними виводить ті рядки, у яких міститься букви та цифри, що введені в поле. Більше того, тип даних можна вибрати в select box'i, що знаходиться над полем вводу. Вибравши тип, пошук відбувається тільки по даному типу. За замовченням пошук відбувається по всім даним (All).

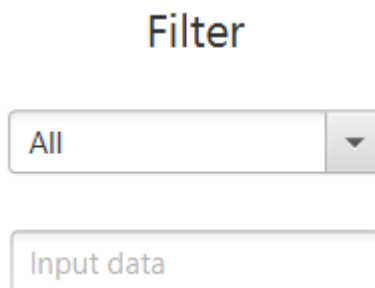


Рисунок 4.6 Фільтр

Оскільки система автоматизована, то при зміні інших таблиць, міняються дані й у таблиці Workers, яку ми зараз розглядаємо. Аби оновити дані усієї таблиці була створена кнопка Refresh all data. Вона знаходиться внизу справа.

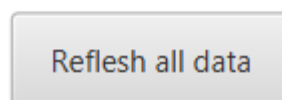


Рисунок 4.7 Кнопка оновлення даних таблиці

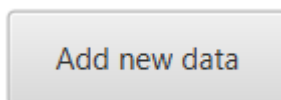
Якщо нажати на один з рядків таблиці, то відповідні дані заповнять поля, а якщо двічі натиснути кнопку мишки на пустий простір, то виділення пропаде. Розташовані ці поля під таблицею з робітниками. Це необхідно для зручного та швидко редагування даних у таблиці. Також можна використати для створення нових рядків у таблиці. Тільки обов'язково треба змінити Worker Id, бо система зрозуміє цю дію як редагування даних, а не створення нових.



Worker Id Full Name Post Spent Cert Id

Рисунок 4.8 Поля даних

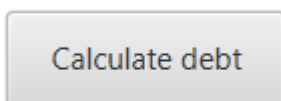
Кнопка для створення та редагування даних у таблиці називається Add new data. Розташована вона в правому нижньому кутку. Дані беруться з полів, що описані вище.



Add new data

Рисунок 4.9 Кнопка введення даних

Кнопка Calculate debt є унікальною для цієї вкладки. Кнопка розташована поруч кнопки оновлення даних таблиці. Calculate debt відкриває вікно, в якому показує обчислення та їх результати, робить висновки хто кому і скільки грошей має дати піприємство чи у випадку робітника повернути кошти, виділенні на відрядження.



Calculate debt

Рисунок 4.10 Кнопка обчислень заборгованості

Інші три таблиці, мають подібний функціонал. Тільки-но Directory of cities не має кнопки оновлення даних, бо на таблицю міст не має інших чинників впливу, які могли б змінити їх дані. Кнопка Calculate debt, як було вище описано, є унікальною

[illegible]

Рисунок 4.11 Вкладка Directory of cities

При виникненні проблеми, на екран виводиться повідомлення про відповідну помилку. Наприклад, при створенні відрядження з від’ємним Id кодом висвітлиться помилка, яка каже що Id має бути не менше нуля і, очевидно проблемна операція не відбудеться.

Error

Id should not less 0

OK

Filter

All

Input data

Refresh all data

Add new data

-1 2020-02-27 2020-03-03 2000 2

Рисунок 4.12 Помилка при створенні відрядження

Висновок

Метою даної курсової роботи було створення інформаційної системи «Відділ бухгалтерського обліку. Облік відряджень». Робота виконана успішно, були реалізовані поставлені задачі, такі як :

1. Створення нових і редагування старих даних про робітників, ордери та завдання, що отримують ці робітники, а також дані про міста, їх типи та ціна проживання в день за фіксованими тарифами.
2. Коректне видалення даних з бази даних. При видаленні міста мають видалятися пов'язані з цим містом завдання, а робітники повинні втрачати ордери на виконання цього завдання.
3. Зручна пошукова система. Ця система повинна виводити потрібні дані орієнтуючись на вказану у фільтрі назву стовпчика або по всій базі даних
4. Система має бути автоматизована, тобто програма сама повинна вираховувати кількість днів поїздки та суму грошей, що підприємство має видати, орієнтуючись на інші дані з інших таблиць.
5. Захист даних. Програма повинна бути захищена принаймні авторизованим входом.
6. Операція обчислення заборгованості

Підчас роботи над цим проектом були засвоєні вагомі знання програмування на мові Java, її бібліотек, та використання реляційної системи управління базами даних Mysql.

Список використаних джерел

1. Java 8. Руководство для начинающих / Герберт Шилдт, 2015. - 1376 с.
2. Системы баз данных. Полный курс / Джеффри Ульман, Дженнифер Уидом, Гектор Гарсія М.: Издательский дом "Вильямс", 2016. - 1088 с.
3. Дейт К. Введение в систему баз данных 8-е издание. : Пер. с англ./ К.Дейт –М.: Издательский дом "Вильямс", 2006. - 1328 с.
4. JavaFX 2.0. Разработка RIA-приложений / Тимур Машніна М.: Издательский дом "БХВ-Петербург", 2012. - 320 с.
5. Бази даних в інформаційних системах / Володимир Гайдаржи, Ігор Ізварін М.: Видавництво Університет "Україна", 2018. – 418 с.
6. Бази даних MySQL / Надія Балик, Віктор Мандзюк М.: Навчальна книга – Богдан, 2010. – 160с.

ДОДАТОК

Лістинг програмного модуля
«Кнопка обчислення заборгованості та шаблони для Mysql запитів, команд»

УКР.НТУУ”КПІ № TI-9253_2020 КР

Листів 5

Київ – 2020

Нижче написаний код надає функціональний сенс кнопці обчислення заборгованості(Calculate debt), яка є унікальною для вкладки Workers.

```
ButtonCalc.setOnAction(actionEvent -> {

    try {

        errorId(FieldId);
        // відбувається перевірка, того що код робітника додатній
        errorId(FieldCert);
        // така ж сама перевірка для коду ордера робітника

        if ("".equals(FieldId.getText()) || FieldId.getText() == null ||
            "0".equals(FieldId.getText()) ||
                "".equals(FieldCert.getText()) || FieldCert.getText() == null ||
            "0".equals(FieldCert.getText())) {

            throw new Exception("Calculation error");

        } // відбувається перевірка того, що поля не пусті – інакше впливає
ПОМИЛКА

        ResultSet rs = new DHandler().selectFromWhere("Sum", "certification",
            "Id_Cert", FieldCert.getText());
        rs.next();

        // За допомогою шаблону з таблиці бази даних Certification дістаємо
значення Sum, де Id_Cert дорівнює значенню з поля FieldCert.

        if ("".equals(FieldSpent.getText()) || FieldSpent.getText() == null )
FieldSpent.setText("0");
```


// Якщо поле FieldSpent пусте , то воно набуває значення 0.

```
int sum = Integer.parseInt(FieldSpent.getText()) - rs.getInt(1);
```

// За допомогою оцієї операції дізнаємося, борг і залежно від того, який знак має число можна зрозуміти хто кому винен гроші.

```
String result;
```

```
if (sum > 0) { // Якщо різниця додатня, гроші має повернути компанія
```

```
    result = "The company should return " + FieldName.getText() + " : ";
```

```
    result += FieldSpent.getText() + " - " + rs.getInt(1);
```

```
    result += " = " + sum + " money " ;
```

```
} else if (sum < 0) { // Інакше кошти вертає співробітник
```

```
    result = FieldName.getText() + " should return the company : ";
```

```
    result += rs.getInt(1) + " - " + FieldSpent.getText();
```

```
    sum = rs.getInt(1) - Integer.parseInt(FieldSpent.getText());
```

```
    result += " = " + sum + " money " ;
```

```
} else result = "no debt "; // Або заборгованості немає
```

```
JOptionPane.showMessageDialog(null,
```

```
    result , // цей рядок-пояснення залежить від значення sum
```

```
    "Calculated debt", JOptionPane.PLAIN_MESSAGE);
```

//На екран виводиться повідомлення, яке вказує на борг особи чи підприємства та показує обчислення

// при невдалому обчисленні появляється вікно, яке оголошує про помилку

```
} catch (Exception e) {
```

```
    e.printStackTrace();
```

```
JOptionPane.showMessageDialog(null,
```

```
    "Calculation error. Select row with certification to calculate.",
```

```

        "Error", JOptionPane.ERROR_MESSAGE);
    }
    } ) ;

```

Наступний код – це шаблони Mysql запитів та команд, які підчас роботи найбільше використовувалися, тому були винесені в окремий файл-клас DHandler.

```

// Шаблон для взяття всіх даних з певної таблиці
public ResultSet getTable(String table) {
    String select = "select * from mydb." + table + " ";
    return executeQuery(select);
}

// Шаблон для видалення конкретного рядку, конкретної таблиці, де
величина на буває конкретного значення
public void deleteRowFromWhere(String table, String where, String value) {
    String select = "delete From mydb." + table + " Where " + where + " = " +
value + " ";
    execute(select);
}

// Шаблон для редагування значень певної таблиці
public void editData(String table, String state) {
    String select = "Update mydb." + table + " set " + state + " ";
    execute(select);
}

// Шаблон для внесення даних у певну таблицю
public void insertData(String table, String state) {
    String select = "Insert into mydb." + table + " values (" + state + ") ";
    execute(select);
}

```

```
}
```

// Шаблон для взяття конкретного рядку, конкретної таблиці, де величина на буває конкретного значення

```
public ResultSet selectFromWhere(String name, String table, String where,
String value) {
    String select = "select " + name + " From mydb." + table + " where " +
where + " = " + value + " ";
    return executeQuery(select);
}
```

// Функція для виконання команд

```
public boolean execute(String select) {
    // Виводить команду в консоль
    System.out.println(select + "\n");
```

/*Команда виконується в Mysql. Якщо вона виконалася успішно, то функція вертає значення true і база даних редагується відповідно до команди. Інакше появляється вікно помилки і функція вертає false */

```
try {
    PreparedStatement prSt;
    prSt = getDConnection().prepareStatement(select);
    prSt.execute();
    return true;
} catch (Exception e) {
    e.printStackTrace();
    JOptionPane.showMessageDialog(null,
        "Error in executing MySQL Command. Problem with data",
        "Error", JOptionPane.ERROR_MESSAGE);
    return false;
```

```

    }
}

// Функція для виконання запитів
public ResultSet executeQuery(String select) {
    // Виводить запит у консоль
    System.out.println(select + "\n");

    /* Mysql виконує запит з бази даних. Якщо він виконався успішно, то
    функція вертає результатний набір (ResultSet), інакше появляється вікно
    помилки і функція вертає null */
    try {

        PreparedStatement prSt;
        prSt = getDConnection().prepareStatement(select);
        return prSt.executeQuery();

    } catch (Exception e) {

        e.printStackTrace();
        JOptionPane.showMessageDialog(null,
            "Error in executing MySQL Query. Problem with data",
            "Error", JOptionPane.ERROR_MESSAGE);
        return null;
    }
}
}

```