# Прикладное программирование. Курсовой проект, 2022/2023

Тема работы: «Разработка клиент-серверного desktop-приложения для поддержки бизнес-процессов компании».

Преподаватель: Харченко Е.А., elenakhaa@yandex.ru.

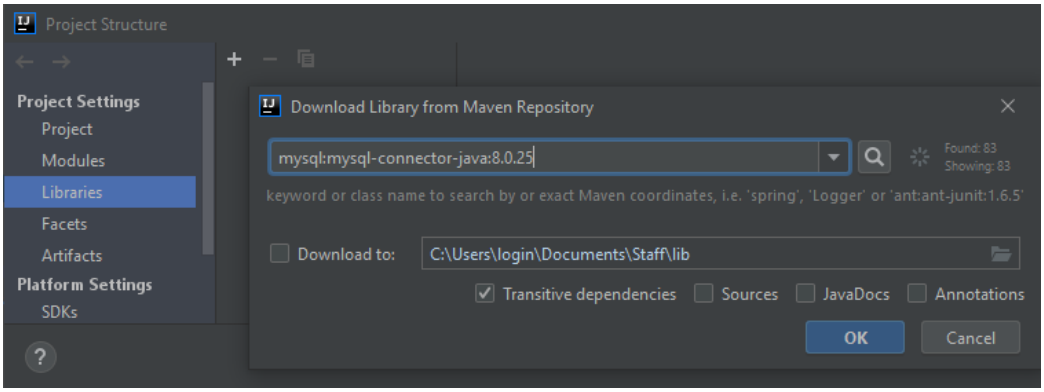
## 1. Постановка задачи

Цель работы – разработать простое оконное *клиент-серверное* приложение, позволяющее пользователю посредством графического интерфейса и согласно предоставляемым ему правам доступа (в ролевой или мандатной модели, см. номер варианта) обрабатывать данные, хранящиеся на сервере и моделирующие некоторую предметную область (см. номер варианта).

## 2. Клиент-серверное взаимодействие

Предварительно ознакомьтесь с инструкцией fit.mospolytech.ru в части установления сетевого соединения с базой данный (через VPN). В качестве тестового задания воссоздайте описанный ниже пример (с корректировкой параметров соединения).

Включите в свой проект (Maven) драйвер JDBC, через меню File в Intellij IDEA:




У каждого на сервере Политеха создана учебная база данных с условным названием «Сотрудники компании» (staff, сквозной пример). Создайте консольное приложение для получения списка должностей компании:

```
import java.sql.*;

public class Main {
    public static void main(String[] args){
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");

            Connection connection = DriverManager.getConnection(
                    "jdbc:mysql://std-mysql.ist.mospolytech.ru:3306/std_1620_staff",
                    "std_1620_staff", "qwerty");

            Statement statement = connection.createStatement();
            String query = "SELECT * FROM posts";
            ResultSet result = statement.executeQuery(query);

            while(result.next()){
                int id = result.getInt("id");
                String name = result.getString("name");
                String short_name = result.getString("short_name");

                System.out.print("Vacant post: ");
                System.out.print("id = " + id);
                System.out.print(", name = \"" + name + "\"");
                System.out.println(", short name = \"" + short_name + "\".");
            }

            connection.close();
        }
        catch(Exception e){
            System.out.println(e);
        }
    }
}
```

Здесь работа с результатами запроса осуществляется непосредственно через поток. В работе же полученные данные при необходимости должно быть возможно многократно «проходить» (с целью сложной обработки данных), не допуская избыточных обращений к базе данных. В работе записи «таблицы» должны быть представлены объектами специально разработанного класса (например, `Post`), а сама «таблица» – объектом специального класса-контейнера (например, `Posts`, т.н. модели в концепции MVC).

## 3. Требования к программному решению

1. Средства разработки – язык программирования Java и система управления базами данных MySQL. *Рекомендуемая* графическая библиотека – JavaFX.

2. Разработка программы должна сопровождаться ведением удаленного репозитория посредством системы контроля версий Git.

3. Структура кода программы должна *строго* соответствовать шаблону проектирования MVC (Model-View-Controller). В программе может быть несколько моделей, для каждой из них следует разработать по одному контроллеру и представлению.

4. База данных должна быть создана на сервере, доступ к которому предоставляется через fit.mospolytech.ru.

5. Каждая таблица базы данных должна удовлетворять нормальной форме Бойса-Кодда.

6. Должны быть предприняты исчерпывающие меры для обеспечения целостности данных при эксплуатации базы данных.

7. Должна быть реализована система регистрации и авторизации пользователей. Пользователь должен иметь возможность изменять свои регистрационные данные.

8. Для пользователей с различными привилегиями должен быть реализован соответствующий функционал в концепции CRUD (Create-Read-Update-Delete), доступный через графический интерфейс пользователя.

9. Запросы к базе данных должны быть *параметрическими* (с целью защиты от SQL-инъекций).

10. Результат запроса должен «оборачиваться» в объект специально разработанного класса согласно шаблону Singleton.

11. Должен быть создан исполняемый JAR-файл программы.

## 4. Требования к оформлению пояснительной записки

Результаты работы должны быть оформлены в виде пояснительной записки в научно-техническом стиле (образец оформления прилагается), содержащей разделы:

- «Введение» – введение в предметную область, выделение проблематики, представление работы (в конце);

- «1 Постановка задачи» – цель работы и задачи (т.е. основные этапы) работы;

- «2 Проектирование и разработка» – полное описание функциональных возможностей приложения, схема базы данных (инфологическая и реляционная модели с пояснениями), структура кода (диаграмма классов с пояснениями), детали и особенности реализации;
- «3 Основные сценарии работы» – демонстрация возможностей приложения на нескольких неодносложных примерах;
- «Заключение» – выводы (достоинтсва, недостатки), перспективы развития проекта, ссылка на репозиторий;
- «Список литературы и интернет-ресурсы».

*Важно*: объем поянительной записки – 16-22 страницы, полный листинг кода не должен включаться в основную часть пояснительной записки, текст должен превалировать над иллюстрациями и таблицами, предложения должны быть развернутыми (не тезисными).

## 5. Критерии оценки

Для получения положительной оценки за курсовой проект необходимо *в полном объеме* выполнить задание, подготовить пояснительную записку и *защитить* работу. Результирующая оценка определяется как среднее значение оценок за программное решение и за пояснительную записку (с учетом результатов защиты работы). Положительная оценка за курсовой проект невозможна, если за программное решение отдельно или за пояснительную записку отдельно получена оценка «неудовлетворительно».

**Вариант 4. Служба доставки [Гладилин, Конопский, Вашкевич, Корнеева, Попов, Чердаков, Макарова, Орлова, Эгамуродзода]**

Владелец небольшой компании по доставке планирует создать информационную систему, которая позволит ему сохранять данные о своих клиентах и доставках:

- У каждого клиента есть уникальный номер, имя, номер телефона и адрес.
- Когда клиент хочет отправить посылку другому клиенту, ему просто нужно войти на веб-сайт компании, выбрать клиента, которому он хочет отправить посылку, ввести вес посылки и указать, является ли доставка обычной или срочной. Затем он получает уникальный идентификационный код, который записывает на упаковке.
- Затем посылка доставляется заказчиком в выбранный им центр доставки. Центр доставки имеет уникальное название и адрес.
- Для каждого клиента ближайший к дому центр доставки. Он выбирается компанией.
- Посылка направляется через внутреннюю систему до тех пор, пока не достигнет центра доставки получателя.
- Из центра доставки посылка доставляется получателю курьером.
- У курьеров есть единый номер, имя и номер телефона. Каждый курьер работает в одном центре доставки.
- Как только пакет вводится в систему, он закрепляется за курьером.