Финальный проект: Приложение для поиска фильмов

Цель проекта

Создать **интерактивное консольное приложение** для поиска фильмов по базе данных sakila (MySQL) с возможностью:

1. Искать фильмы:
   - По ключевому слову
     - Поиск по названию фильма
     - Результат ограничен 10 фильмами
     - По запросу пользователя отображаются следующие 10 результатов, или пока не закончатся
   - По жанру и диапазону годов выпуска
     - Перед вводом пользователю показываются:
       - Список всех жанров из таблицы жанров
       - Минимальный и максимальный год выпуска фильмов в базе
     - Указывается нижняя и верхняя граница, например: от 2005 до 2012, или конкретный год
     - По запросу пользователя отображаются следующие 10 результатов, или пока не закончатся
2. Сохранять все поисковые запросы в **MongoDB** в коллекцию `final_project_<your_group>_<your_full_name>`.
3. Выводить список из 5 **популярных запросов** (по частоте или последним поискам).

Технологии

- Python
- MySQL (данные о фильмах)
- MongoDB (логирование запросов)
- `pymysql`, `pymongo`, другие модули (например для вывода в консоль)

Дополнительные требования

1. **Соблюдение стиля PEP8**
   - Код оформлен согласно стандартам PEP8 (отступы, пробелы, длина строк, имена переменных).
   - Используются понятные имена переменных и функций.
2. **Читаемость кода**
   - Каждая функция выполняет **одну понятную задачу**, без лишней вложенности и дублирования.
   - Присутствуют **комментарии** там, где это необходимо (например, описание логики запроса).
   - Блоки кода (меню, логика поиска, логика логирования) **разделены по функциям или файлам**.
3. **Отсутствие дублирования кода**
   - Повторяющиеся действия (например, подключение к базам, формат вывода, обработка ошибок) вынесены в **отдельные функции**.
   - Вместо копирования одного и того же кода в разных частях — используется **повторное** использование функций.
   - Используются **переменные** для повторяемых параметров.

Пример структуры проекта

- `main.py` — точка входа, меню и обработка команд пользователя
- `mysql_connector.py` — подключение к MySQL и функции поиска
- `log_writer.py` — запись поисковых запросов в MongoDB
- `log_stats.py` — получение статистики из MongoDB (частые и последние запросы)
- `formatter.py` — функции форматирования вывода (например, таблицы)

Пример структуры MongoDB логов

Python

```
{
  "timestamp": "2025-05-01T15:34:00",
  "search_type": "keyword",
  "params": {
    "keyword": "matrix"
  },
  "results_count": 3
}
```

IT CAREER HUB

Процесс работы над проектом

Шаг 1: Изучение базы данных

- Подключитесь к MySQL-базе данных и изучите таблицы, связанные с фильмами (названия, жанры, даты выпуска).
- Определите, откуда брать информацию о жанрах и годах.
- Проверьте, какие значения реально присутствуют в базе (например, диапазон годов).

Шаг 2: Написание SQL-запросов

- Реализуйте два основных сценария:
  - Поиск по ключевому слову (поиск по части названия фильма).
  - Поиск по жанру и диапазону годов (перед этим пользователь должен увидеть список всех доступных жанров и диапазон годов).
- Убедитесь, что запросы корректно работают с любыми данными, в том числе с пустыми результатами.

Шаг 3: Настройка MongoDB и логирование

- Подключитесь к MongoDB.
- Реализуйте запись каждого поискового запроса: тип, параметры, количество результатов и время.
- Создайте функции для:
  - вывода самых популярных запросов
  - вывода последних уникальных запросов

Шаг 4: Разработка консольного интерфейса

- Настройте интерактивное меню, где пользователь может:
  - Выполнить поиск по ключевому слову.
  - Выполнить поиск по жанру и диапазону годов.
  - Посмотреть популярные или последние запросы.
  - Выйти из приложения.
- Программа должна обрабатывать ошибки (например, неправильный ввод или сбой подключения) и не завершаться аварийно.

Шаг 5: Структурирование кода и оформление

- При необходимомти разделите код на модули: `mysql_connector.py`, `log_writer.py`, `log_stats.py`, `formatter.py`.

- Соблюдайте читаемость, стиль, и избегайте дублирования.
- Добавьте комментарии к ключевым функциям и структурам данных.

Шаг 6 (по желанию): Улучшения

- Реализуйте дополнительные возможности (например, фильтрация по рейтингу).

IT CAREER HUB

Финальный результат

Оценка за проект

Во время защиты преподаватель даст комментарии по вашей работе, а затем во внеурочное время выставит оценку за проект в LMS.

1 — (отлично) (sehr gut) (very good, excellent!) — 81-100%
2 — (хорошо) (gut) (good, well above average!) — 61-80%
3 — (удовлетворительно) (befriedigend) (satisfactory, average!) — 41-60%
4 — (неудовлетворительно) (ausreichend) (sufficient, borderline!) — 21-40%
5 — (неудовлетворительно) (nicht ausreichend) (not sufficient, failed!) — до 20%

Критерий	1	2	3	4	5
Сценарии поиска	Приложение корректно выполняет все указанные сценарии поиска	Не до конца проработан 1 из сценариев поиска (например, нет ограничения на показ 10 вариантов)	Один из сценариев поиска слабо проработан/оба сценария имеют недоработки, но выдают корректные результаты	Один из сценариев поиска отсутствует или работает с ошибками/оба сценария значительно не доработаны	Не реализован поиск/большинство запросов срабатывают не корректно
Логирование запросов в MongoDB	Все запросы логируются в MongoDB в указанную коллекцию, логи читаемы и понятны	Все запросы логируются, но формат логирования имеет недостатки (например, неполнота информации)	Часть запросов не логируется или при логировании в некоторых редких ситуациях могут возникать ошибки (например, внесение некорректных данных или порча данных)	Имеются серьезные постоянно возникающие ошибки в логировании (например, внесение неправильно отформатированных данных или порча данных)	Логи отсутствуют или полностью нечитаемы

Доступ к статистике	Пользователь может посмотреть статистику: 5 популярных запросов, последние и по частоте. Доступ к статистике удобно организован, статистика корректная	Пользователь может посмотреть статистику: 5 популярных запросов, последние ИЛИ по частоте. Доступ к статистике удобно организован, статистика корректная ИЛИ В программе имеется возможность посмотреть полную статистику, но доступ к ней не очевиден	Пользователь может посмотреть один из вариантов статистики, но доступ не очевидный	Статистика есть, но работает не корректно (неверные данные или при попытке получения возникает ошибка)	Нет функции посмотреть статистику
Стабильность работы программы	Программа работает стабильно, не завершается при ошибках, а выводит понятные сообщения при некорректном вводе или сбоях подключения.	Программа работает стабильно, но некоторые сообщения об ошибках не понятны	В редких случаях программа может завершиться без сообщения об ошибке/ большинство сообщений об ошибках непонятны пользователю	Лишь часть (менее 60%) ошибок обрабатываетс я	Ошибки не обрабатываютс я
Оформление кода	Код структурирован , читаем и разделён на логические части	Код в основном структурирован и читаем, есть небольшие погрешности (например, не хватает нескольких комментариев или они слишком краткие)	Код в основном структурирован и читаем, но есть 1-2 дублирования кода/ непонятные названия некоторых переменных/ отдельные блоки кода не	Общая структура есть, но чтение кода заметно затруднено: нехватка комментариев, есть несколько случаев дублирования кода, большая часть кода в одном файле	Код хаотичен и сложно читаем

			выделены в функции/файлы		
Вопросы преподавателя	Ответ логичный и аргументированный. Упоминаются конкретные аспекты проекта, используются корректные термины, нет неточностей.	Возможны 1–2 несущественные ошибки, упрощения или неполное раскрытие ответа. Студент всё равно показывает хорошее понимание темы.	Ответы неуверенные, студент теряется, но с подсказками выходит на суть. Используются обобщённые формулировки, мало конкретики.	Видно, что студент знает, к какой части проекта относится вопрос, но затрудняется в деталях. Может дать общую догадку, назвать инструмент или этап, но не раскрыть суть.	Ответы типа «не знаю», «этим не я занимался», «не понял, что это». Нет даже общего представления, с чем связан вопрос. Подсказки не помогают.

6 критериев, каждый из которых вносит равный вклад.

Оценка за проект = $\frac{\text{сумма оценок по каждому критерию}}{6}$, округляется до 0.5 включительно в меньшую сторону.

Пример:

Критерий	**Оценка**
Сценарии поиска	1
Логирование запросов в MongoDB	2
Доступ к статистике	2
Стабильность работы программы	3
Оформление кода	1
Вопросы преподавателя	2
Сумма баллов	**11**
Итоговая оценка	**2**

Важно: качественная реализация дополнительных функций может положительно повлиять на оценку, если основной функционал имеет **незначительные недочеты** (на усмотрение преподавателя можно вычесть из оценки до округления **0,1 балл**).

IT CAREER HUB

Презентация результатов

Каждый участник должен **презентовать свой проект**:

- Кратко рассказать об идее и реализованном функционале (что умеет приложение).
- Показать работу консольного приложения:
  - Пример поиска по ключевому слову.
  - Пример поиска по жанру и диапазону годов.
  - Демонстрация вывода популярных запросов.
- Отвечать на вопросы преподавателя по архитектуре, коду, работе с базами данных.

Продолжительность презентации: 5 минут.