Технологический стек:
- FastAPI
- SQLAlchemy 2.0
- AsyncIO
- Aiohttp или Httpx
- JWT для аутентификации

Описание задания:

Создайте асинхронный RESTful API сервис, который позволяет пользователям регистрироваться, аутентифицироваться и управлять списком своих избранных фильмов. Необходимо интегрировать Kinopoisk API Unofficial для получения информации о фильмах.

Интеграция с внешним API:

- Используйте Kinopoisk API Unofficial: https://kinopoiskapiunofficial.tech/
- Для доступа к API необходимо самостоятельно зарегистрироваться и получить бесплатный API ключ.
- Ознакомьтесь с условиями использования и ограничениями API.

Эндпойнты вашего сервиса:

1. Регистрация пользователя:
   - POST /register
   - Регистрация нового пользователя с указанием имени пользователя и пароля.

2. Авторизация пользователя:
   - POST /login
   - Аутентификация пользователя и получение JWT токена.

3. Получение профиля пользователя:
   - GET /profile
   - Получение информации о текущем аутентифицированном пользователе.

4. Поиск фильмов:
   - GET /movies/search?query=НазваниеФильма
   - Требует аутентификации.
   - Ищет фильмы по названию, используя эндпойнт:
   - GET /api/v2.1/films/search-by-keyword
   - Возвращает результаты поиска с основной информацией о фильмах.

5. Получение деталей фильма:
   - GET /movies/{kinopoisk_id}

- Требует аутентификации.
- Получает подробную информацию о фильме по его Kinopoisk ID, используя эндпойнт:
  - GET /api/v2.2/films/{kinopoisk_id}

6. Добавление фильма в избранное:
- POST /movies/favorites
- Требует аутентификации.
- Добавляет фильм в список избранных пользователя по Kinopoisk ID.

7. Удаление фильма из избранного:
- DELETE /movies/favorites/{kinopoisk_id}
- Требует аутентификации.
- Удаляет фильм из списка избранных пользователя.

8. Просмотр списка избранных фильмов:
- GET /movies/favorites
- Требует аутентификации.
- Возвращает список избранных фильмов пользователя с подробной информацией.

Требования к реализации:

1. Аутентификация и авторизация:
- Реализуйте аутентификацию с использованием JWT.
- Защитите эндпойнты /profile и все эндпойнты под /movies/ от неавторизованного доступа.

2. Асинхронность:
- Используйте async/await во всех частях приложения для обеспечения неблокирующих операций ввода-вывода.

3. Работа с базой данных:
- Используйте SQLAlchemy 2.0 для взаимодействия с базой данных.
- Безопасно сохраняйте информацию о пользователях (пароли должны быть захешированы).
- Храните список избранных фильмов каждого пользователя в базе данных.
  - Сохраняйте Kinopoisk ID и необходимую информацию о фильме.

4. Взаимодействие с внешним API:
- Используйте Aiohttp или Httpx для выполнения асинхронных HTTP-запросов напрямую.

5. Обработка данных:
- При поиске фильмов отправляйте запрос к эндпойнту /api/v2.1/films/search-by-keyword и возвращайте результаты.

- При получении деталей фильма используйте эндпойнт /api/v2.2/films/{kinopoisk_id}
- При добавлении фильма в избранное сохраняйте информацию о фильме в базе данных.
- При просмотре списка избранных фильмов возвращайте детальную информацию о каждом фильме.

6. Обработка ошибок:
- Корректно обрабатывайте возможные ошибки (например, фильм не найден, проблемы с внешним API, неверный Kinopoisk ID и т.д.).
- Возвращайте соответствующие HTTP-коды ответов и информативные сообщения об ошибках.

7. Качество кода:
- Пишите чистый, читаемый и хорошо организованный код.
- Соблюдайте правильную структуру проекта.
- Следуйте лучшим практикам разработки на Python и FastAPI.
- Используйте типизацию для повышения качества кода.

8. Документация:
- Используйте автогенерируемую документацию FastAPI (Swagger UI).
- Убедитесь, что все эндпойнты и модели ввода/вывода задокументированы.

9. Безопасность:
- Хэшируйте пароли перед сохранением в базу данных.
- Обрабатывайте исключения и избегайте утечки подробных технических деталей во внешние сообщения об ошибках.

10. Опционально:
- Реализуйте базовые тесты для основных функций вашего API.
- Напишите docker-compose файл для запуска

Ожидаемый результат:

- Исходный код API, соответствующий вышеописанным требованиям.
- README файл с инструкциями по запуску и использованию API.
- Пример файла конфигурации (например, .env), исключая любые конфиденциальные данные (такие как API ключи).

Срок выполнения:

- 24 часа с момента получения задания.

Формат сдачи:

- Исходный код на GitHub/GitLab или в виде архива.
- Убедитесь, что репозиторий содержит инструкцию по настройке и запуску приложения.

Критерии оценки:

- Соответствие функциональным требованиям.
- Качество и организация кода.
- Корректное использование асинхронного программирования.
- Эффективная интеграция с внешним API.
- Обработка ошибок и обеспечение безопасности.
- Наличие и качество документации.
- Следование лучшим практикам разработки.

Примечания:

- Регистрация и получение API ключа:
  - Зарегистрируйтесь на Kinopoisk API Unofficial и получите API ключ.

- Заголовки для запросов к Kinopoisk API:
  - При отправке запросов к Kinopoisk API необходимо указывать API ключ в заголовках:
    - X-API-KEY: ваш API ключ.
    - Content-Type: application/json.

- Рекомендации по работе с базой данных:
  - Продумайте модели данных для пользователей и избранных фильмов.
  - Используйте отношения между таблицами для связывания пользователей с их избранными фильмами.

- Работа с конфигурацией:
  - Используйте файл .env или подобный механизм для хранения конфиденциальных настроек (API ключ, секрет для JWT и т.д.).