Министерство образования и науки Российской Федерации Федеральное государственное автономное образовательное учреждение высшего образования «Уральский федеральный университет имени первого Президента России Б. Н. Ельцина»

Отчёт по лабораторной работе №4

«Тестирование бэкенд приложения»

Выполнил: Тринц Илья Олегович

Группа: РИМ-150950

**Цель работы**

Познакомится со способами тестирования приложения.

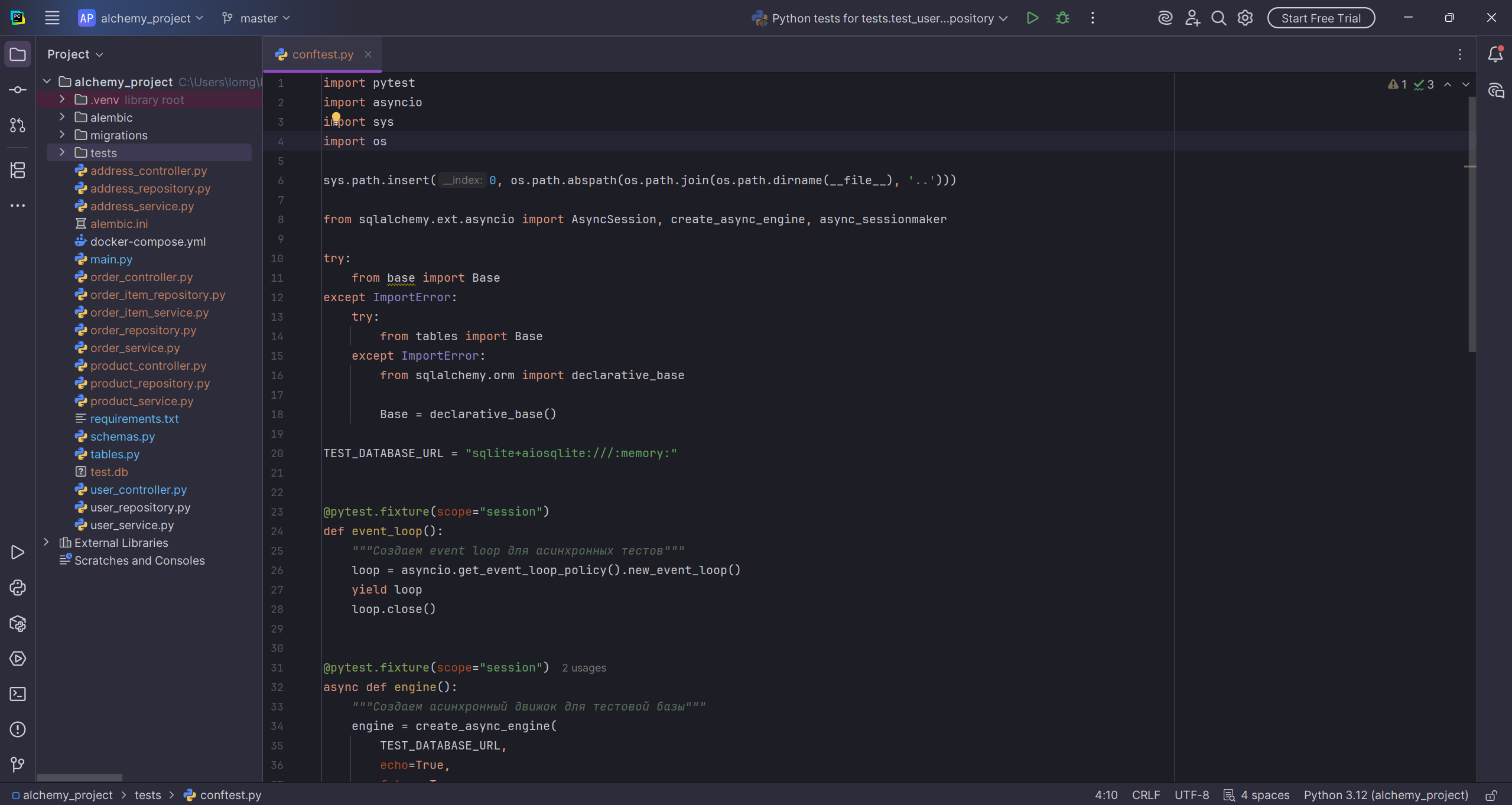
**Задачи**

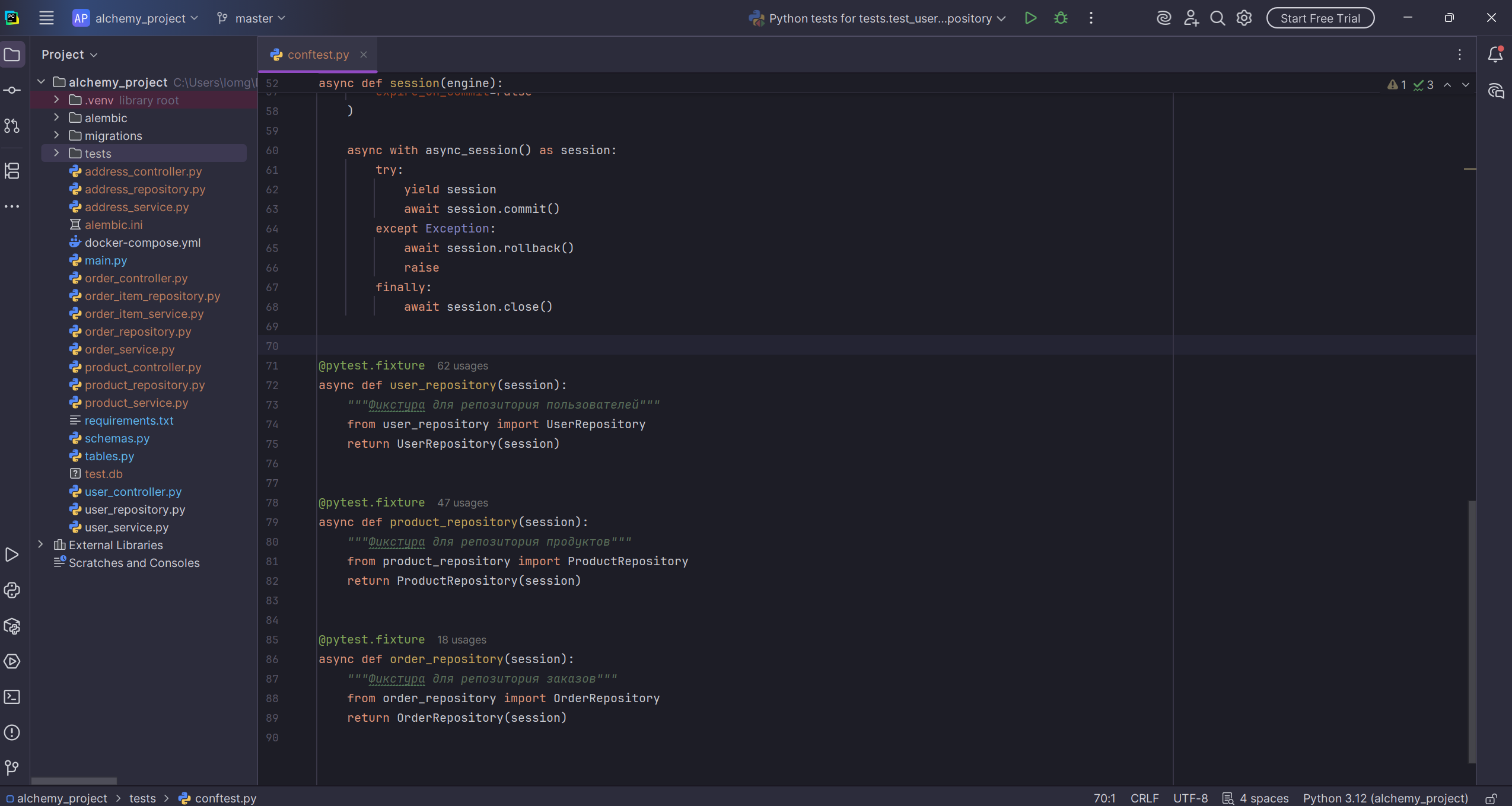
1. Создать тестовую базу и подключиться к ней;
2. Протестировать создание пользователя;
3. Протестировать сервисной слой приложения;
4. Протестировать API;
5. Ответить на вопросы.

**Ход работы**

1. Создание тестовой базы.

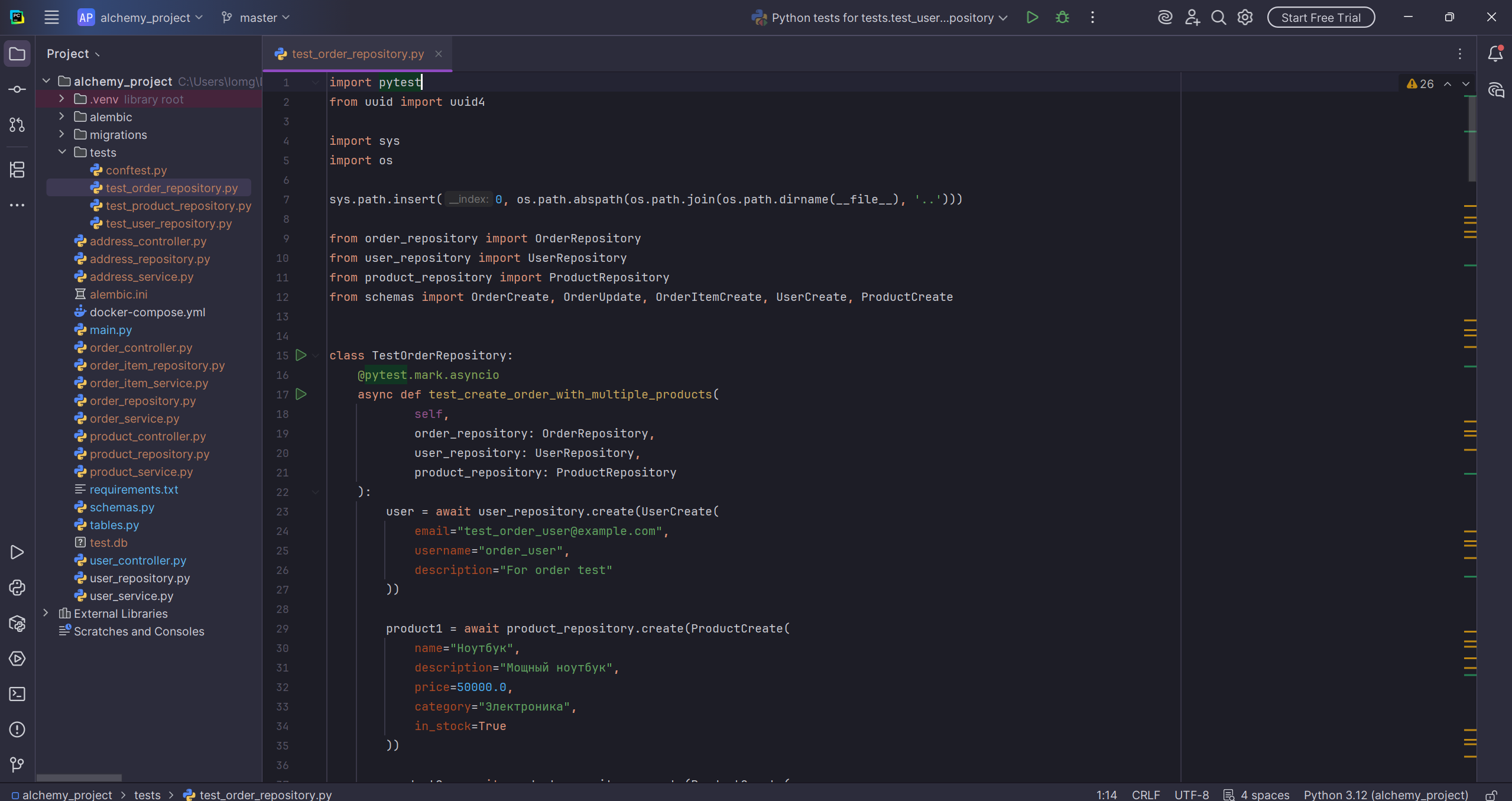
Создадим репозитории по недостающим объектам, реализуем подключение к тестовой БД и подключим наши репозитории.

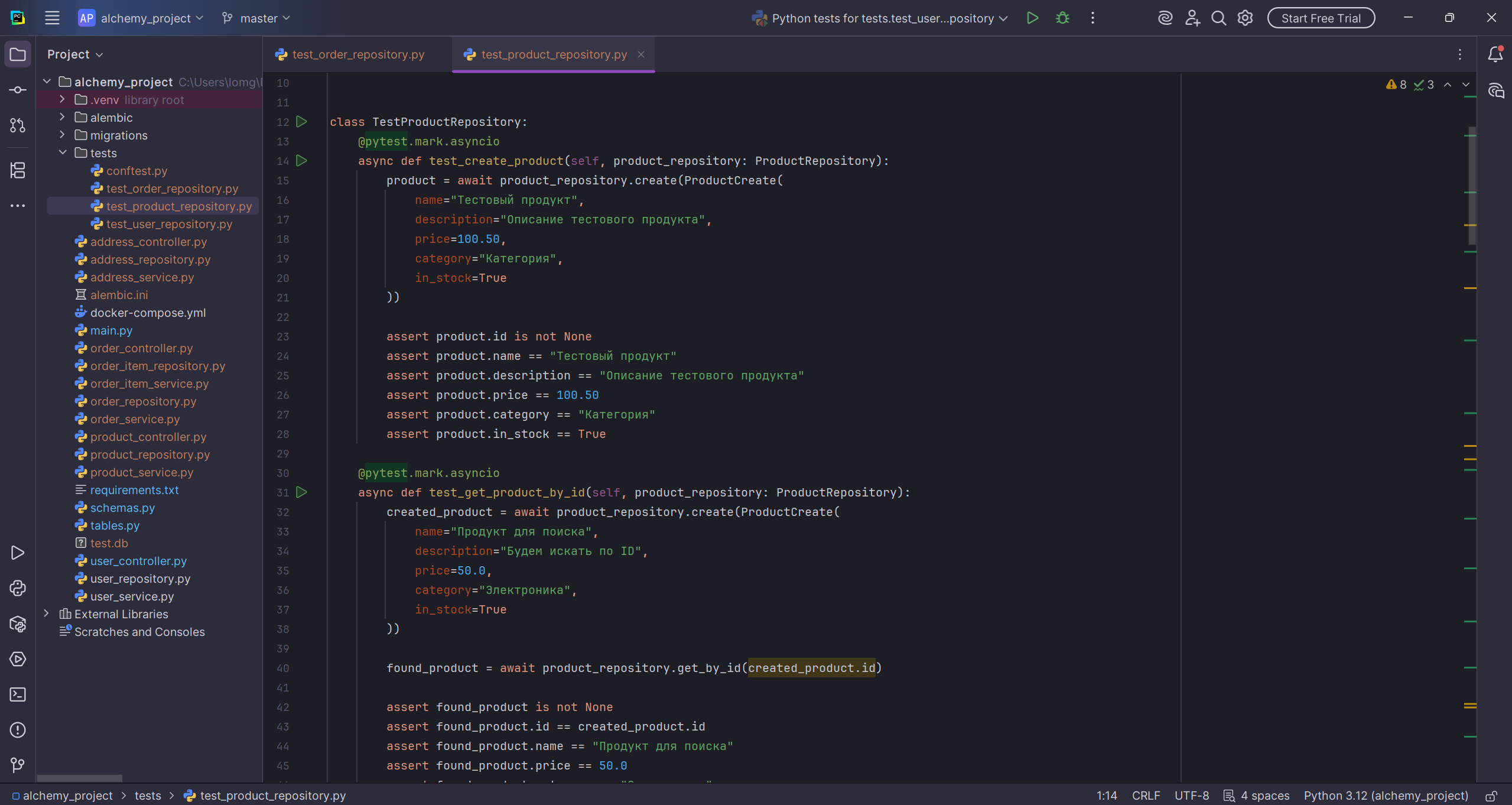


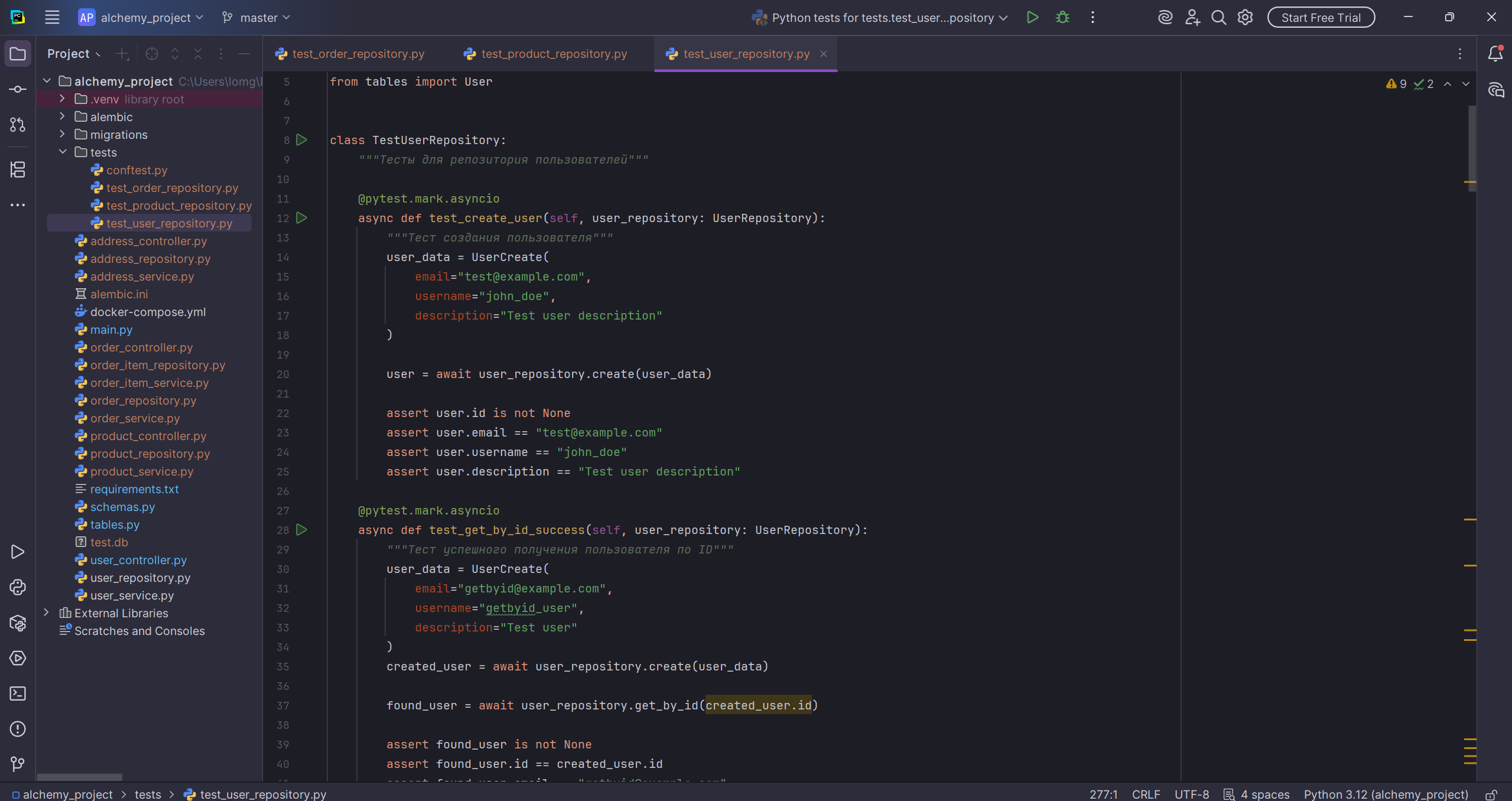


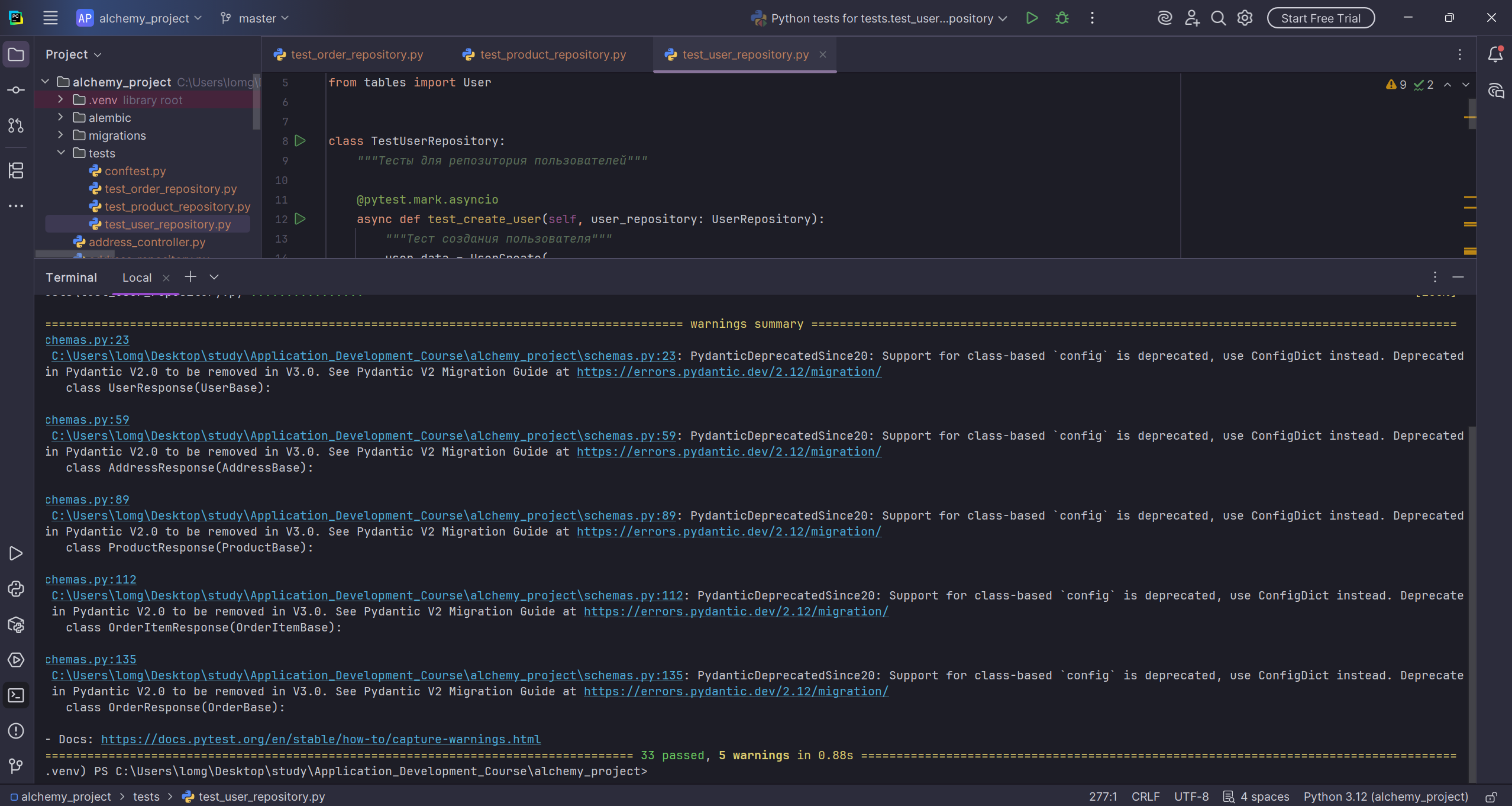
2. Тестирование создания пользователя.

Напишем тесты и запустим их на репозиторий пользователей, заказов и продуктов.



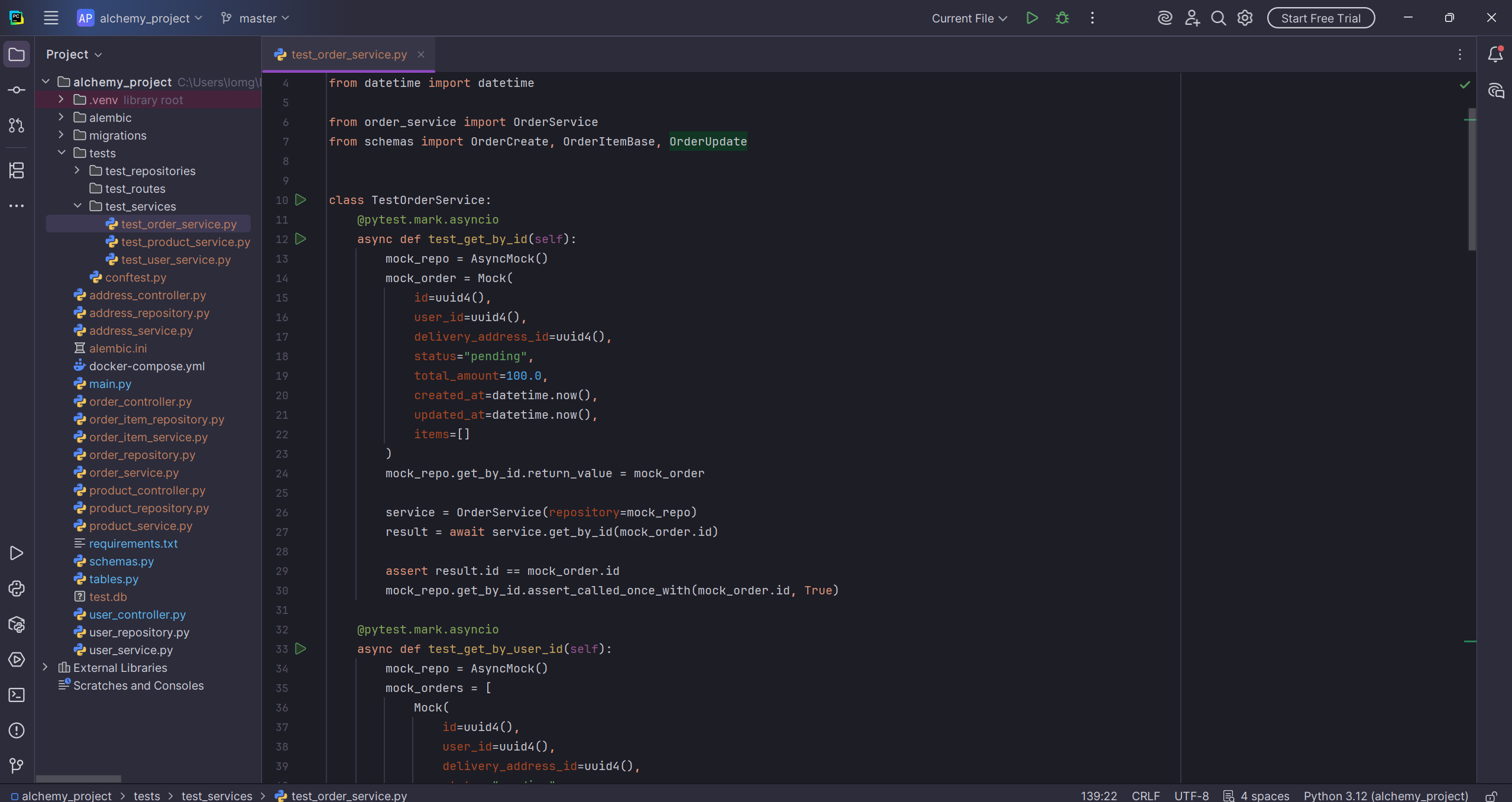


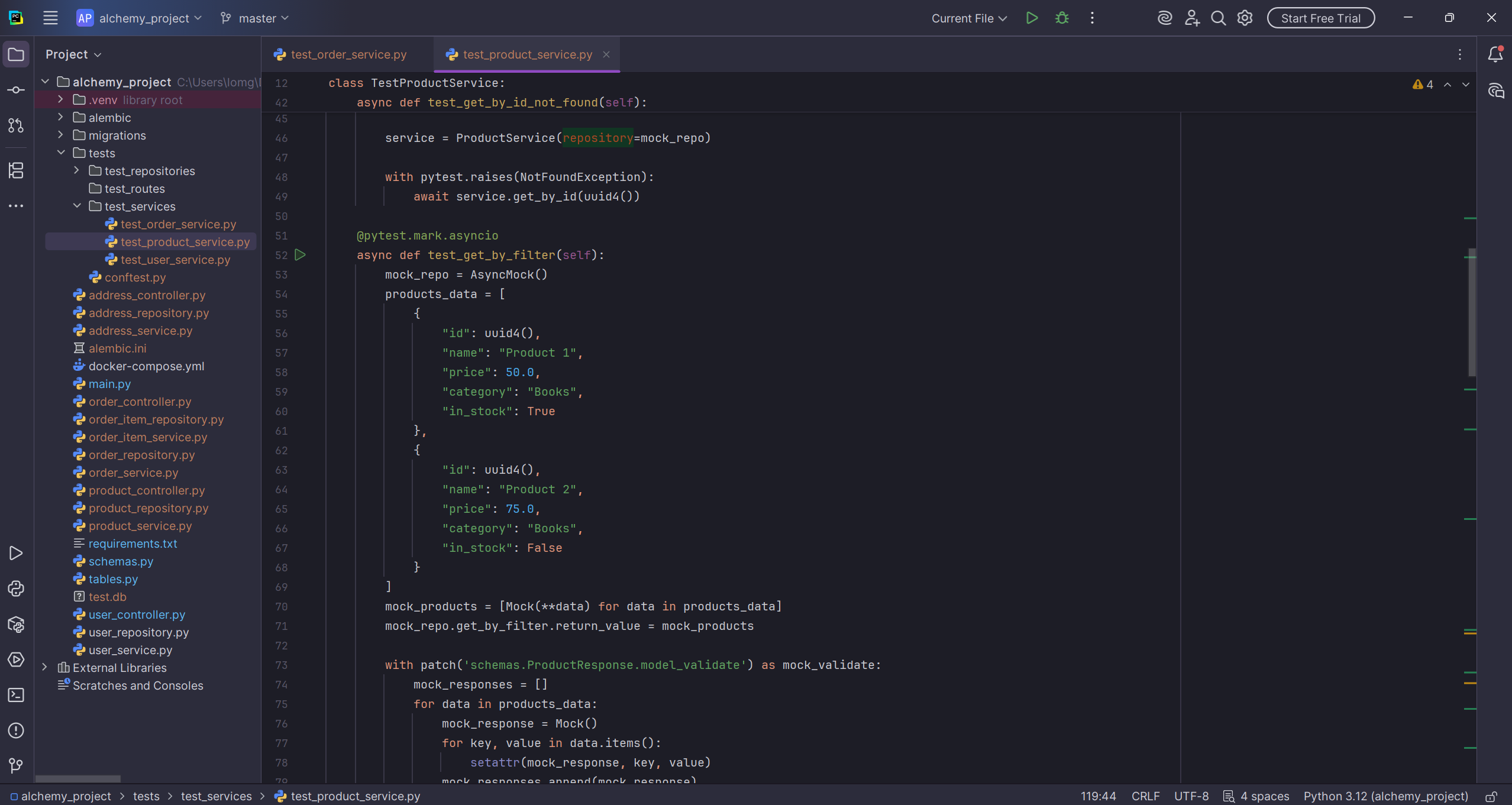


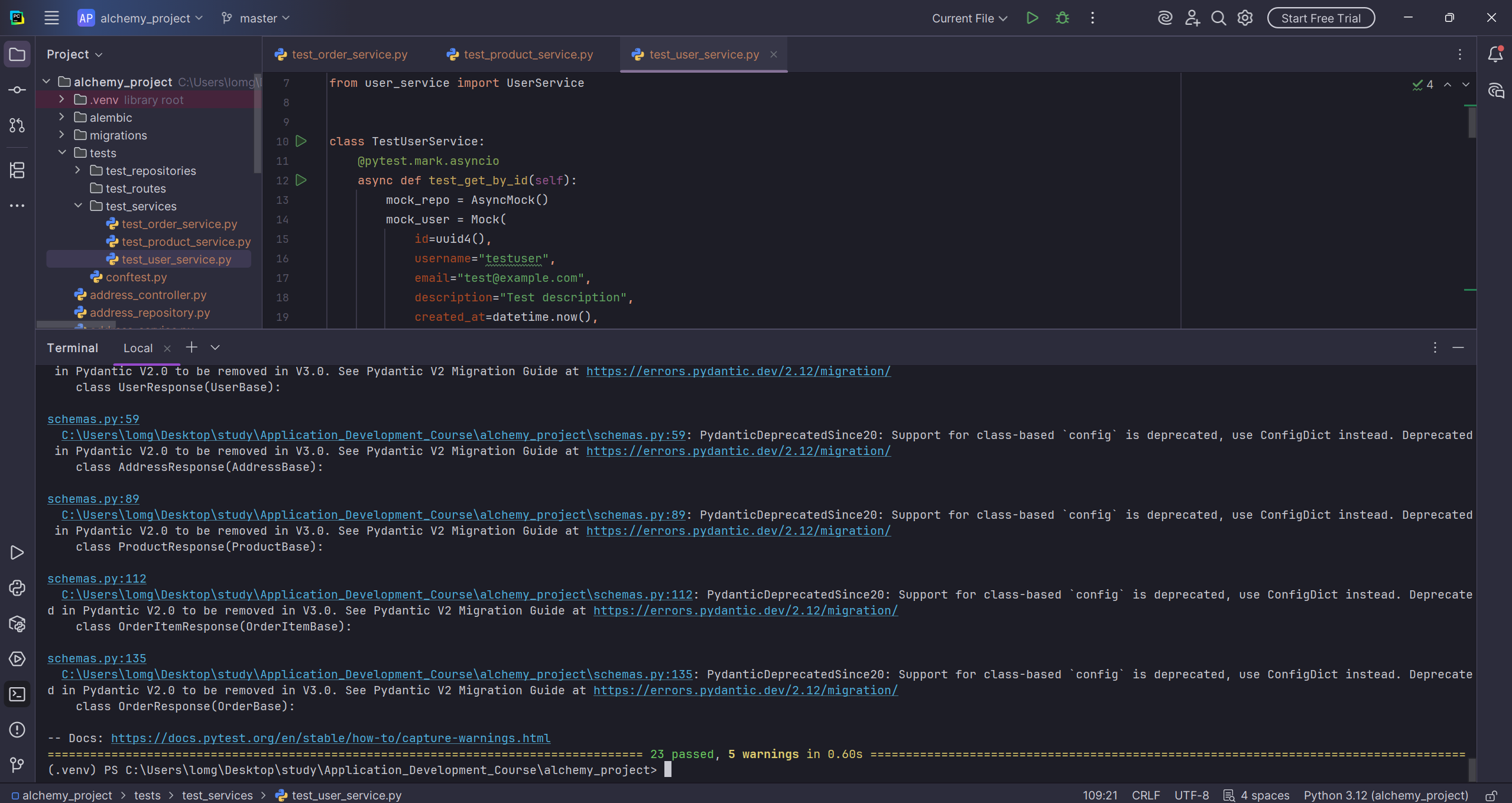
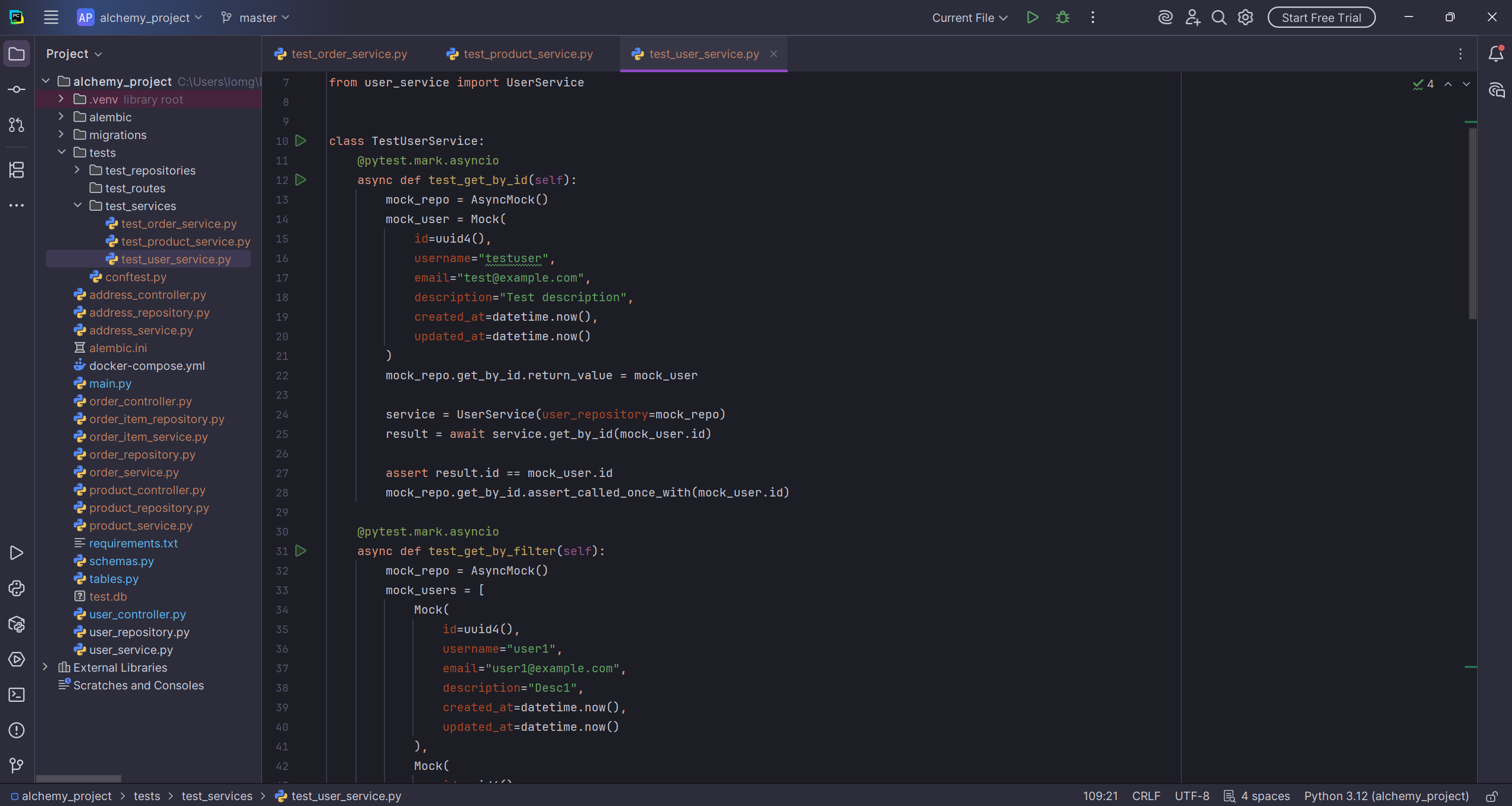


3. Тестирование сервисного слоя приложения.

Напишем тесты на сервисный слой с помощью моков и просмотрим результат тестов.

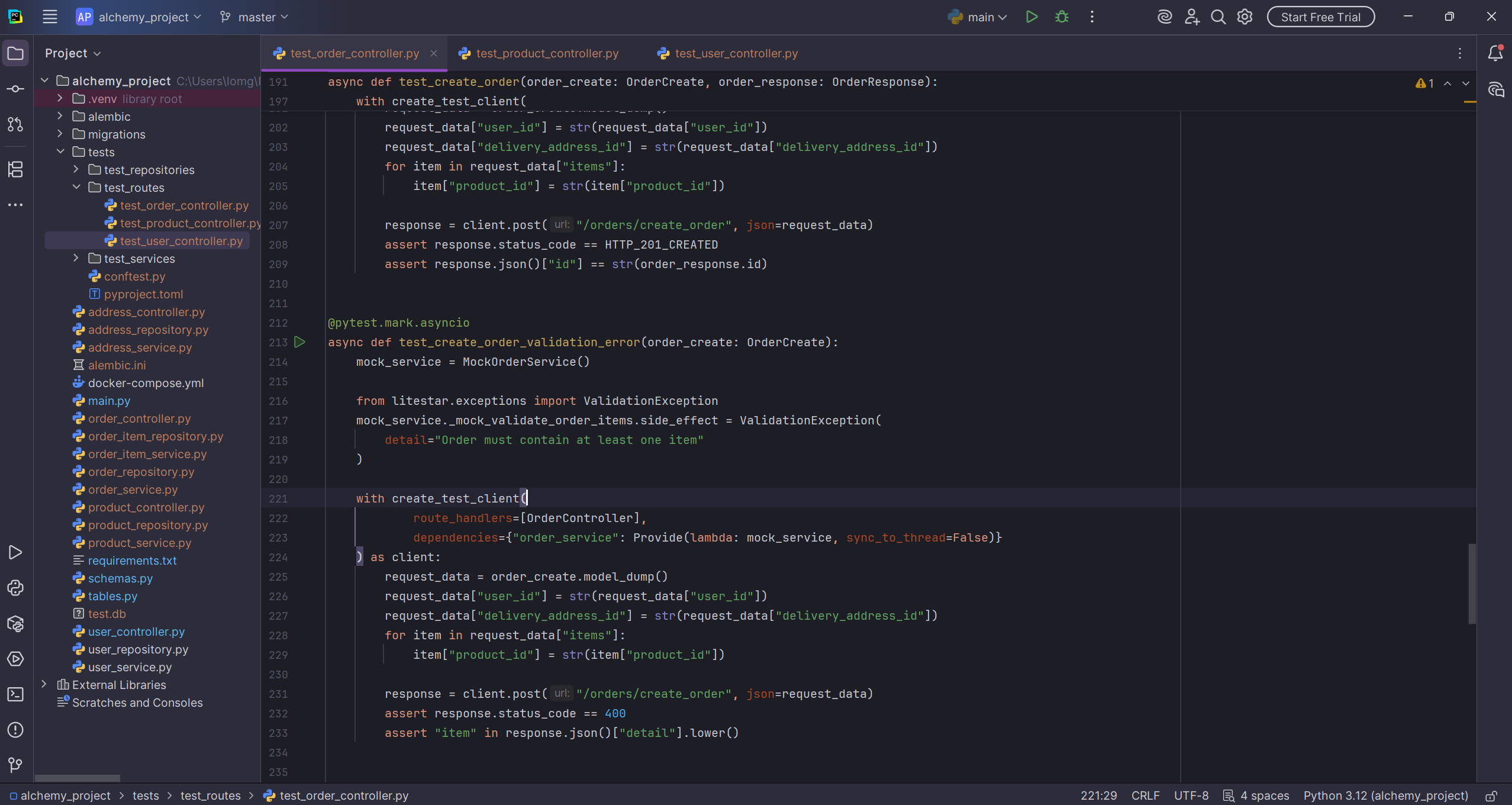




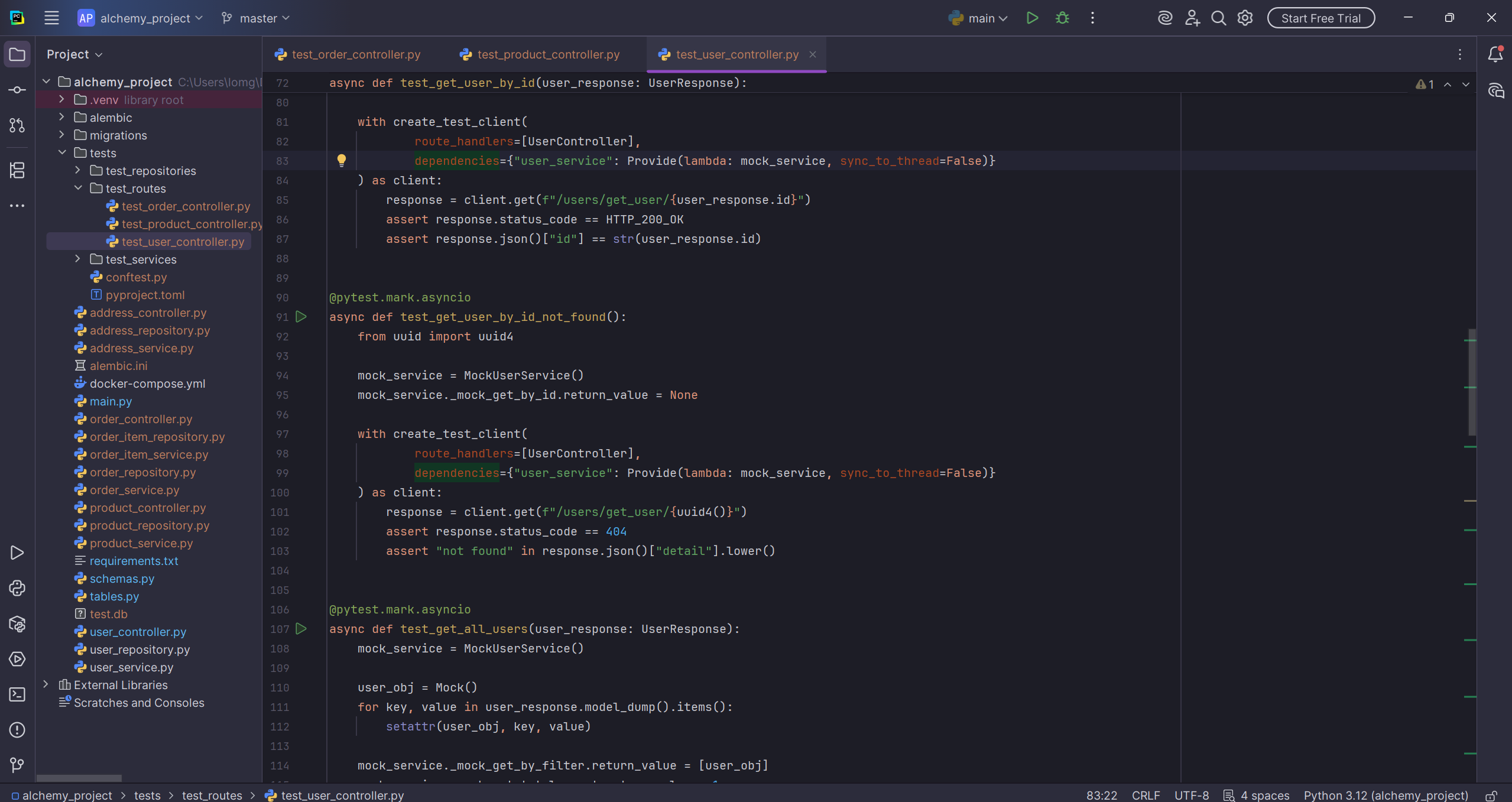


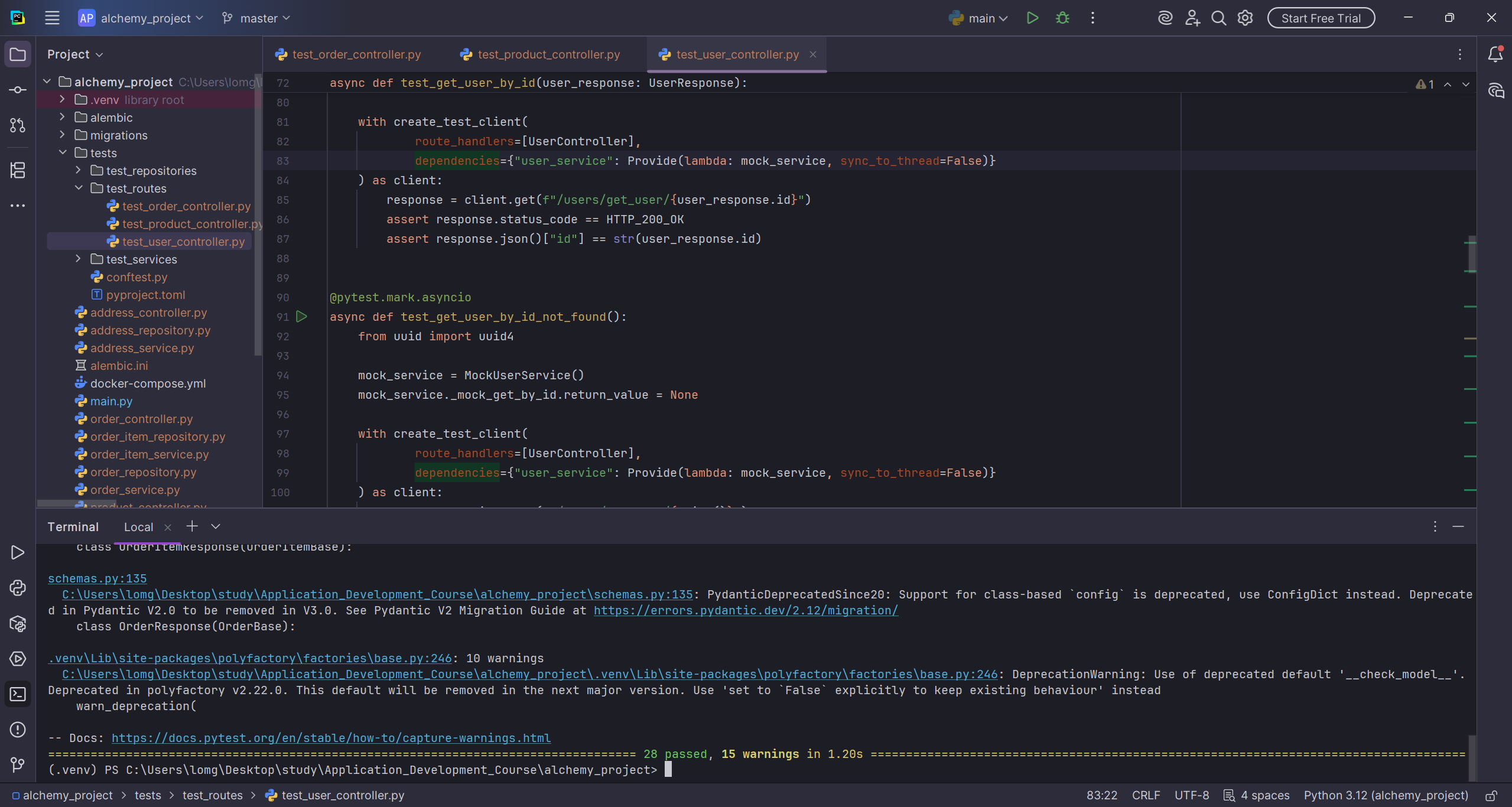
4. Тестирование API.

Напишем тесты на контроллеры и проведём их запуск.









5. Ответы на вопросы.

1. Почему в тестах мы используем отдельную тестовую базу данных (SQLite in-memory)? Какие проблемы могут возникнуть при использовании production базы данных для тестирования?

- Используем тестовую бд для простоты тестирования: не нужно отдельно поднимать базу с постоянным хранилищем данных, не попортим данные на реальной базе, если будем подключаться к ней. Если использовать production базу, то проблема что можем потерять или попортить реальные данные, а также можем случайно раскрыть креды этой базы.

2. Как работает TestClient в Litestar? Какие преимущества он дает по сравнению с обычными HTTP-запросами?

- Обходит сетевое взаимодействие, преобразуя запросы прямо в вызовы приложения. Преимущества: быстрее, проще настройка, легче деббажить, нет затрат ресурсов на http-client.

3. При тестировании сервиса заказов, какие edge cases (граничные случаи) нужно учитывать? Напишите к ним тесты

- Нужно учитывать есть ли товары и какое кол-во, работу с пользователем и взаимодействие с другими заказами.

Тесты: создать заказ с 0 товаров, создать заказ с 1\_000\_000 товаров, создать заказ с несуществующим пользователем, создать заказ с существующим пользователем, создать заказ без пользователя, создать заказ на пользователя без заказов, создать заказ на пользователями с уже существующими заказами.

4. Как бы вы протестировали метод, который должен отправлять email при смене статуса заказа на "shipped"?

- Подменил бы реализацию отправки метода на удобный мне, или же использовал тестовый email сервер и сделал проверку с отправкой письма туда.

5. Напишите тест для проверки пагинации товаров. Какие параметры должны проверяться?

- Должны проверяться параметры страниц и кол-во отображения на этих страницах.

Тесты: показать товары на странице -1 с кол-во -100, показать товары на странице 1 с 0, на странице 0 с 0, на странице 0 с 1, на странице 2 с 0, на странице 2 с 2 (по сути все возможные граничные случаи пагинации).

6. Как обеспечить изоляцию тестов друг от друга? Почему это важно?

- Изоляция важна для предсказуемости тестов и их результата и независимости от порядка их запуска.

Обеспечить можно с помощью отдельной тестовой бд, транзакций, а самое главное – чистой реализации тестов, чтобы результат выполнения какого-либо теста не менял результат выполнения любого другого теста.

Ссылка на репозиторий: <https://github.com/lomonoga/Application_Development_Course>

**Вывод:**

В ходе этой работы мы вернули и дополнили код со 2 ЛР, создали тестовую реализация бд и протестированы разные слои приложения, а в частности уровень репозитория, уровень сервиса и уровень API.