**Unit-тесты для Taskbench**

1. TaskAPITests – тесты задач

|  |  |
| --- | --- |
| **Тест** | **Что проверяет** |
| test\_get\_tasks\_list | Получение списка всех задач, привязанных к пользователю, с валидным токеном. |
| test\_get\_tasks\_filter\_by\_date | Фильтрация задач по конкретной дате дедлайна; возвращается только одна соответствующая задача. |
| test\_get\_tasks\_sort\_by\_priority | Сортировка задач по приоритету в убывающем порядке и дополнительная проверка на третичную сортировку по ID при одинаковом приоритете. |
| test\_get\_tasks\_sort\_by\_deadline | Сортировка задач по возрастанию дедлайна, проверка правильного порядка. |
| test\_create\_task | Создание новой задачи с подзадачами, проверка сохранения в базу и увеличения их общего количества. |
| test\_invalid\_date\_filter | Обработка некорректного значения фильтра по дате — ожидается 500. |
| test\_update\_task | Обновление существующей задачи по ID: изменение названия и приоритета, и проверка сохранения. |
| test\_delete\_task | Удаление (фактически завершение) задачи: задача помечается как завершённая, но не удаляется физически. |
| test\_create\_subtask | Создание подзадачи для конкретной задачи — проверка, что новая подзадача сохраняется корректно. |
| test\_update\_subtask | Обновление подзадачи — меняется текст и статус выполнения. |
| test\_delete\_subtask | Удаление подзадачи по ID и проверка её удаления из базы. |
| test\_unauthorized\_access | Попытка получить задачи без авторизации — должен возвращаться статус 401. |
| test\_create\_task\_invalid\_data | Попытка создать задачу с некорректными полями — ожидается ошибка 400. |
| test\_filter\_by\_datetime\_range | Фильтрация задач по диапазону времени (after и before) — задача должна попадать в интервал. |
| test\_conflicting\_filters | Проверяет, что при одновременном указании date и after сервер возвращает ошибку 400 с полем error в JSON-ответе. |
| test\_invalid\_datetime\_filters | Проверяет, что API корректно реагирует на невалидные значения параметров after и before, возвращая ошибку 400 в обоих случаях. |
| test\_exclude\_tasks\_without\_deadline | Создает задачу без дедлайна и убеждается, что при фильтрации по after она не попадает в ответ, даже если есть другие задачи с дедлайнами. Всего задач должно быть 3, но фильтр должен вернуть только 2 с установленными дедлайнами. |
| test\_combined\_after\_before\_filter | Создает три задачи: одну с дедлайном внутри интервала и две — вне. Проверяет, что при фильтрации по after и before возвращается только задача, попадающая в заданный диапазон. |

2. CategoryAPITests – тесты категорий

|  |  |
| --- | --- |
| **Тест** | **Что проверяет** |
| test\_get\_categories | Проверяет, что пользователь может получить список своих категорий. В ответе должны быть только категории, связанные с текущим пользователем. |
| test\_create\_category\_success | Проверяет успешное создание новой категории с корректным именем. Ожидается статус 201 и увеличение количества категорий в базе. |
| test\_create\_category\_missing\_name | Проверяет, что при отсутствии поля name в запросе возвращается ошибка 400 и поле error в JSON-ответе. |
| test\_create\_category\_long\_name | Проверяет, что при попытке создать категорию с именем длиннее 50 символов возвращается ошибка 400 и сообщение об ошибке. |
| test\_create\_category\_duplicate\_name | Проверяет, что при попытке создать категорию с уже существующим именем у текущего пользователя возвращается ошибка 409. |
| test\_unauthenticated\_access | Проверяет, что доступ к эндпоинту /categories без авторизации невозможен: и GET, и POST запросы без токена возвращают статус 401. |
| test\_invalid\_json\_post | Проверяет, что при передаче некорректного JSON (например, строки 'invalid json') в теле запроса возвращается ошибка 400 с описанием ошибки. |

3. UserApiTests – тесты пользователей

|  |  |
| --- | --- |
| **Тест** | **Что проверяет** |
| test\_login\_and\_password\_change | Проверяет успешную авторизацию пользователя по email и паролю, смену пароля с авторизацией через access токен, а также повторный вход с новым паролем. |
| test\_register\_refresh\_and\_delete | Проверяет регистрацию нового пользователя, получение access и refresh токенов, обновление access токена через refresh, а затем удаление пользователя с использованием access токена. |

4. SuggestionServiceTestCase – тесты подсказок

|  |  |
| --- | --- |
| **Тест** | **Что проверяет** |
| test\_deadline\_suggestion | Проверяет, извлекается ли корректная дата и время из текста. |
| test\_subtask\_suggestion | Проверяет, что из текста извлекаются подзадачи. |
| test\_category\_suggestion | Проверяет, что из текста правильно определяется категория. |
| test\_priority\_suggestion | Проверяет, правильно ли определяется приоритет задачи по тексту. |

5. SuggestionApiTestCase – API-подсказки

|  |  |
| --- | --- |
| **Тест** | **Что проверяет** |
| test\_suggestion\_api | Проверяет работу API, которое предлагает дедлайн, подзадачи, приоритет и категорию на основе текста задачи и временной метки, при условии авторизации. |

6. StatisticsAPITests – тесты статистики

|  |  |
| --- | --- |
| **Тест** | **Что проверяет** |
| test\_authentication\_required | Проверяет, что доступ к статистике невозможен без авторизации и возвращается статус 401. |
| test\_empty\_statistics | Проверяет корректный ответ API, если у пользователя нет завершённых задач — значения статистики должны быть нулями. |
| test\_statistics\_calculation | Проверяет, что статистика задач (сегодня, за неделю, максимум) корректно рассчитывается на основе завершённых задач с разными датами. |
| test\_different\_users\_statistics | Проверяет, что статистика задач разделяется по пользователям, и каждый пользователь получает только свои данные. |
| test\_invalid\_token | Проверяет, что при попытке доступа к статистике с недействительным токеном API возвращает статус 401. |

**Возникшие трудности при написании тестов:**

1. **Изменения в API.**  
   Даже незначительные изменения в API, например, в алгоритме сортировки задач, приводили к сбоям ранее написанных тестов. Приходилось довольно часто переписывать часть тестов, подстраивая под новый код.
2. **Отладка неудачных тестов.**  
   В случае провала теста сообщения об ошибке не всегда были информативными. Чтобы найти причину, приходилось добавлять промежуточные print()-вызовы или вручную анализировать тело ответа от сервера.