**Отчёт**

по результатам написания интеграционных тестов

по дисциплине “Технологии разработки качественного программного обеспечения”

| Выполнил  студент гр. 3530904/80106 | Кудряков Д. В. |
| --- | --- |
|  |  |
|  |  |

**1. Использованные технологии**

Для проведения интеграционного тестирования мы использовали pytest и TravisCI.

**2. Тестовые сценарии**

Суммарно был реализован 21 интеграционный тест.

**Тестовый сценарий 1 Проверка получения информации о пользователе**

Начальное состояние для всех тест кейсов: Имеется схема тестового сервера со следующими эндпоинтами:

1. /me – информация о пользователе
2. /me/connections – информация о соединениях пользователя

| № | Тест кейс | Шаги тестирования | Данные для теста | Ожидаемый результат |
| --- | --- | --- | --- | --- |
| 1 | Получение информации будучи неавторизованным | 1. Запустить приложение 2. Изменить поле authorized на False 3. Отправить GET запрос на эндпоинт /me | - | Код ошибки 401 |
| 2 | Получение информации о пользователе | 1. Запустить приложение 2. Изменить поле authorized на True 3. Отправить GET запрос на эндпоинт /me | Экземпляр класса User с полными данными | Код «успеха» 200 и данные пользователя в формате json |
| 3 | Получение информации о соединениях пользователя будучи неавторизованным | 1. Запустить приложение 2. Изменить поле authorized на False 3. Отправить GET запрос на эндпоинт /me/connections | - | Код ошибки 401 |
| 4 | Получение информации о соединениях пользователя | 1. Запустить приложение 2. Изменить поле authorized на True 3. Отправить GET запрос на эндпоинт /me/connections | Экземпляр класса User с полными данными, экзмепляр класса UserConnection | Код «успеха» 200 и данные соединениях пользователя в формате json |

**Тестовый сценарий 2 Проверка работы возможности взаимодействия со списком гильдий**

Начальное состояние для всех тест кейсов: Имеется схема тестового сервера со следующими эндпоинтами:

1. /me/guilds – информация о гильдиях пользователя
2. /add\_to/<int:guild\_id> – добавление пользователя к гильдии

| № | Тест кейс | Шаги тестирования | Данные для теста | Ожидаемый результат |
| --- | --- | --- | --- | --- |
| 1 | Получение информации о гильдиях будучи неавторизованным | 1. Запустить приложение 2. Изменить поле authorized на False 3. Отправить GET запрос на эндпоинт /me/guilds | - | Ошибка |
| 2 | Получение информации о гильдиях (пользователь является админом гильдии) | 1. Запустить приложение 2. Изменить поле authorized на True 3. Отправить GET запрос на эндпоинт /me/guilds | Экземпляр класса User с полными данными, экзмепляр класса Guild (с указанным пользователем в качестве адимна) | Код «успеха» 200 и строка «[ADMIN] test\_name» |
| 3 | Получение информации о гильдиях (пользователь не является админом гильдии) | 1. Запустить приложение 2. Изменить поле authorized на True 3. Отправить GET запрос на эндпоинт /me/guilds | Экземпляр класса User с полными данными, экзмепляр класса Guild | Код «успеха» 200 и строка «test\_name» |
| 4 | Добавление пользователя в гильдию будучи неавторизованным | 1. Запустить приложение 2. Изменить поле authorized на True 3. Отправить GET запрос на эндпоинт /add\_to/1 | - | Ошибка |
| 5 | Добавление пользователя в гильдию | 1. Запустить приложение 2. Изменить поле authorized на True 3. Отправить GET запрос на эндпоинт /add\_to/1 | Экземпляр класса User с полными данными, экзмепляр класса Guild | Код «успеха» 200 и строка «{}» |

**Тестовый сценарий 3 Проверка работы входа в аккаунт и выхода из него**

Начальное состояние для всех тест кейсов: Имеется схема тестового сервера со следующими экндпоинтами:

1. / – пустой эндпоинт для перенаправления после авторизации
2. /login – старт сессии авторизации без данных
3. /login-prompt – старт сессии авторизации с перезапросом OAuth2
4. /login-data - старт сессии авторизации с данными
5. /invite-bot – пригасить бота пользователем с правами на это
6. /invite-bot-invalid-permissions - пригасить бота пользователем без прав на это
7. /invite-oauth – аналог login-data, но с другими данными
8. /logout – завершение сессии

| № | Тест кейс | Шаги тестирования | Данные для теста | Ожидаемый результат |
| --- | --- | --- | --- | --- |
| 1 | Вход в аккаунт | 1. Запустить приложение 2. Отправить GET запрос на эндпоинт /login | - | Правильная ссылка на переадресации |
| 2 | Вход в аккаунт с данными | 1. Запустить приложение 2. Отправить GET запрос на эндпоинт /login | Данные в формате словаря | Правильная ссылка на переадресации с данными |
| 3 | Приглашение бота с правами на это | 1. Запустить приложение 2. Отправить GET запрос на эндпоинт /invite-bot | Данные бота в формате словаря | Правильная ссылка на переадресации с данными |
| 4 | Приглашение бота без прав на это | 1. Запустить приложение 2. Отправить GET запрос на эндпоинт /invite-bot-invalid-permissions | Данные бота в формате словаря | Ошибка |
| 5 | Завершение сессии | 1. Запустить приложение 2. Отправить GET запрос на эндпоинт /logout | Данные о текущем пользователе | Перенаправление на базовую страницу / |
| 6 | Старт сессии авторизации с перезапросом OAuth2 | 1. Запустить приложение 2. Отправить GET запрос на эндпоинт /login-prompt | Данные в формате словаря, с отсутствующим disable\_guild\_select | Ошибка KeyError |
| 7 | Вход в аккаунт с данными | 1. Запустить приложение 2. Отправить GET запрос на эндпоинт /login-oauth | Данные в формате словаря | Правильная ссылка на переадресации с данными |

**Тестовый сценарий 4 Проверка работы модуля для парсинга json с файлами**

Начальное состояние для всех тест кейсов: Наличие файла в системе в папке test\_data

| № | Тест кейс | Шаги тестирования | Данные для теста | Ожидаемый результат |
| --- | --- | --- | --- | --- |
| 1 | Правильные поля со значением true | 1. Прочитать файл из системы 2. Запустить парсер | Json файл с полями true | “true” |
| 2 | Правильные поля со значением false | 1. Прочитать файл из системы 2. Запустить парсер | Json файл с полями false | “false” |
| 3 | Неправильные поля | 1. Прочитать файл из системы 2. Запустить парсер | Json файл с неправильно написанными полями | Ошибка |

**Тестовый сценарий 5 Тестирование работы декоратора проверки наличия аутентификации**

Начальное состояние для всех тест кейсов: Имеется схема тестового сервера со следующими экндпоинтами:

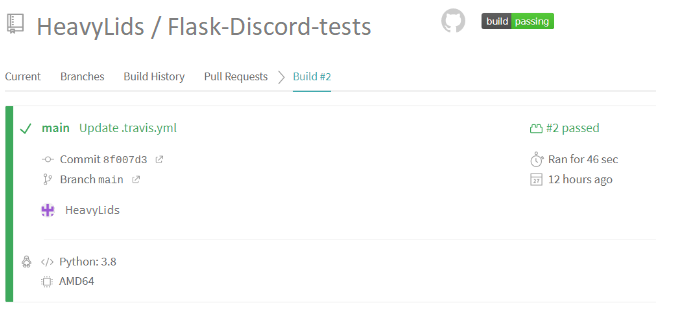
1. / – пустой эндпоинт возвращающий True, если авторизация пройдена

| № | Тест кейс | Шаги тестирования | Данные для теста | Ожидаемый результат |
| --- | --- | --- | --- | --- |
| 1 | Проверка доступа без авторизации | 1. Запустить приложение 2. Изменить поле authorized на False 3. Отправить GET запрос на эндпоинт / | - | Ошибка Unauthorized |
| 2 | Правильные поля со значением false | 1. Запустить приложение 2. Изменить поле authorized на True 3. Отправить GET запрос на эндпоинт / | - | Код «успеха» 200 и параметр authorized = True |

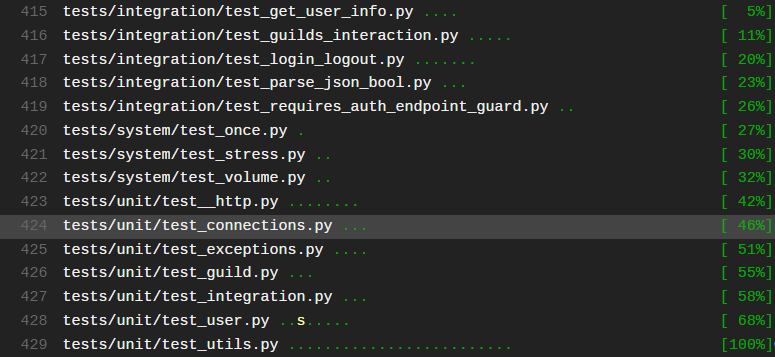
**3. Отчёт о прохождении тестов**

Все тесты выполняются без ошибок. В качестве системы CI/CD было решено использовать Travis CI.

Отчет CI/CD по состоянию билда и прохождению тестов. Все выполнено без ошибок.

****

**Отчет CI/CD по пройденным тестам:**

****

**4. Описание процедуры расширения тестового набора**

В папке интеграционных тестов создаётся новый файл с именем test\_\*название\*.py. В файл импортируются модули, интеграцию которых собираемся тестировать.

Далее согласно сценариям тестирования, пишутся тесты с проверкой ожидаемого и фактического результатов.