HEISENBUG X5 Tech

Корпоративные инструменты спешат на помощь

Мария Снопок
Владислав Григорьев

X5 Group, Москва

Владислав **Григорьев**

Мария **Снопок**

HEISENBUG X5 Tech

ЦЕЛЬ РАБОТЫ

Рассказать
об инструментах и подходах

для автоматизации тестирования в X5 Tech

Рассказать
о полезных фичах

которые мы используем в этих инструментах

Поделиться
нашим опытом

внедрения этих инструментов
(профит и проблемы)

ПРОБЛЕМЫ

Структура проекта

CII/CD и Docker

Selenium

Понятные отчеты и логи после AT

Работа с моделями Swagger

HEISENBUG X5 Tech

Корпоративные **инструменты**

БИБЛИОТЕКА X5QAUTILS

Успешно внедрена на 14 продуктах компании

ШАБЛОННЫЙ ПРОЕКТ

Активно используется при разработке автотестов с нуля

DOCKER IMAGES

Активно используются на проектах для ускорения сборки и выполнения тестов

X5 Tech

X5QaUtils

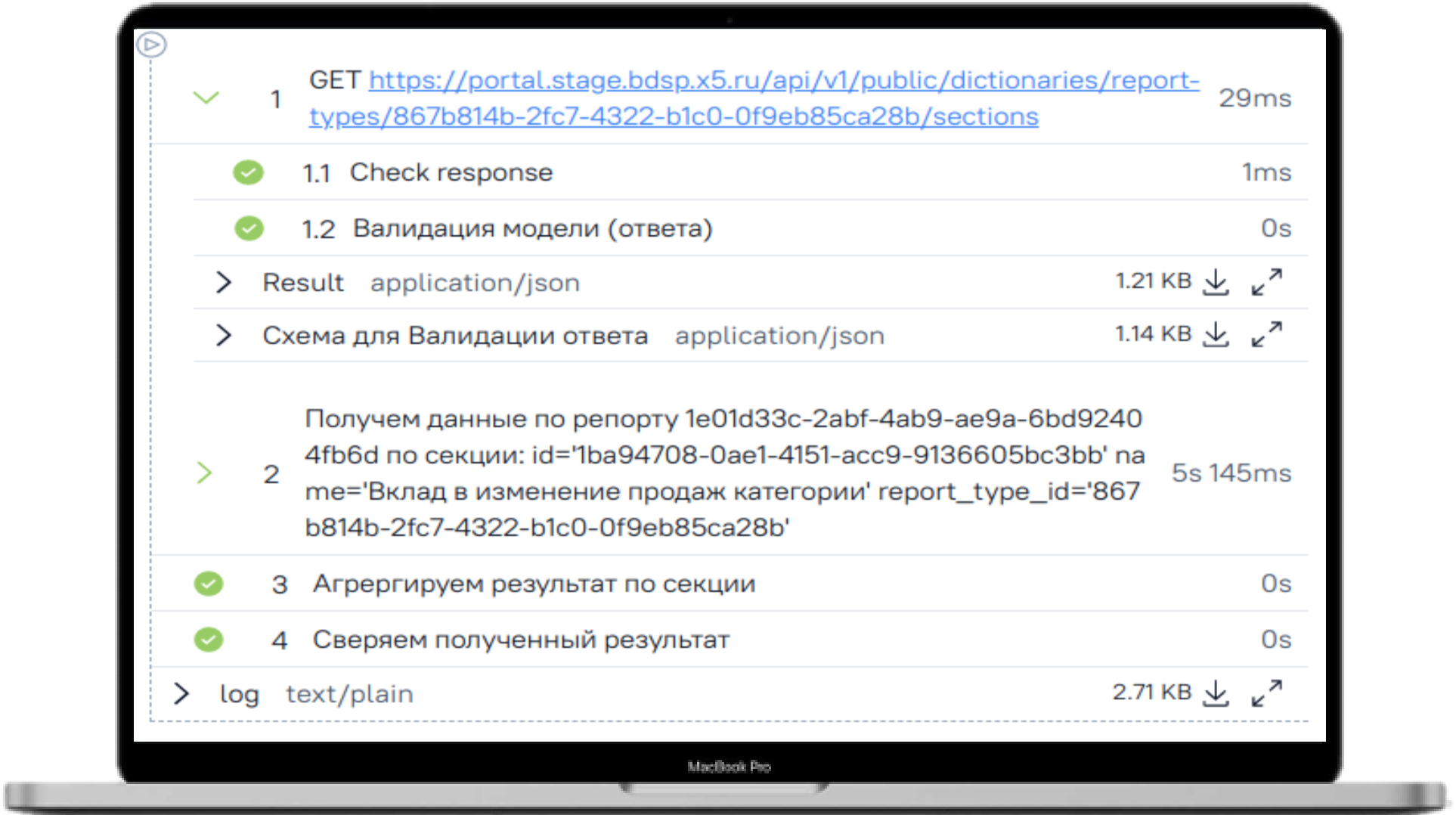
X5 Tech

API

Логирование **запросов**

```
2023-03-01 08:08:37 [API] [INFO] - Auth in keycloak with login
2023-03-01 08:08:37 [API] [INFO] - curl -X GET https://portal.stage.bdsp.x5.ru/api/v1/public/auth/token
2023-03-01 08:08:38 [API] [INFO] - Result status code: 200 time: 0:00:00.249203
{"code":"ok","result":{"token":". [...]
2023-03-01 08:08:38 [API] [INFO] - curl -X GET
https://portal.stage.bdsp.x5.ru/api/reports?...&sortProperty=CREATED_AT
2023-03-01 08:08:38 [API] [INFO] - Result status code: 200 time: 0:00:00.097005
{"code":"ok","result":{"pageNumber":0,"content":[...]
2023-03-01 08:08:38 [testing] [INFO] - Run tests for 1e01d33c-2abf-4ab9-ae9a-6bd92404fb6d
2023-03-01 08:08:38 [API] [INFO] - curl -X GET
https://portal.stage.bdsp.x5.ru/api/v1/public/dictionaries/report-types/.../sections
2023-03-01 08:08:38 [API] [INFO] - Result status code: 200 time: 0:00:00.025905 {"code":"ok","result":[...]
2023-03-01 08:08:38 [API] [INFO] - curl -X GET
https://portal.stage.bdsp.x5.ru/api/v1/public/reports/visualization/.../meta
2023-03-01 08:08:41 [API] [INFO] - Result status code: 200 time: 0:00:03.278510 {"code":"ok","result":[...]
2023-03-01 08:08:41 [API] [INFO] - curl -X GET
https://portal.stage.bdsp.x5.ru/api/v1/public/reports/visualization/...
2023-03-01 08:08:43 [API] [INFO] - Result status code: 200 time: 0:00:01.839927 {"code":"ok","result":...]
```

РАБОТА С ALLURE



Валидация **request/response**

- Pydantic
- Json Validator
- Проверка status code
- Сохранение файла

Магические методы для моделей

HEISENBUG X5 Tech

- **Итерирование**

```
[v.code for v in iter_model_response]
```

- **Доступ к атрибутам через GET**

```
result['name'] || result.name ||
result.get('name', 'Fail'
```

Сравнение **для моделей**

```
def test_contains(self):
    post_create = CreatePost(title='Title',
body='Any Photos')
    new_post = Post(id='1', user_id='user',
**post_create.dict())

    assert post_create in new_post
                        OR
    new_post.equal(post_create)
```

X5 Tech

UI

PageObject/PageElement

main.py

```
class MainPage(BasePage):
    @allure.step("Выбор проекта")
    def choose_project(self, name):
        self.wait(LocatorsGen.project)
        self.app.common.fill_select_area(LocatorsGen.project, name)

    @allure.step("Смена языка")
    def change_lang(self, lang='Русский'):
        self.click(LocatorsGen.lang_icon)
        self.click_parametrize(LocatorsGen.lang_but, param=lang)
```

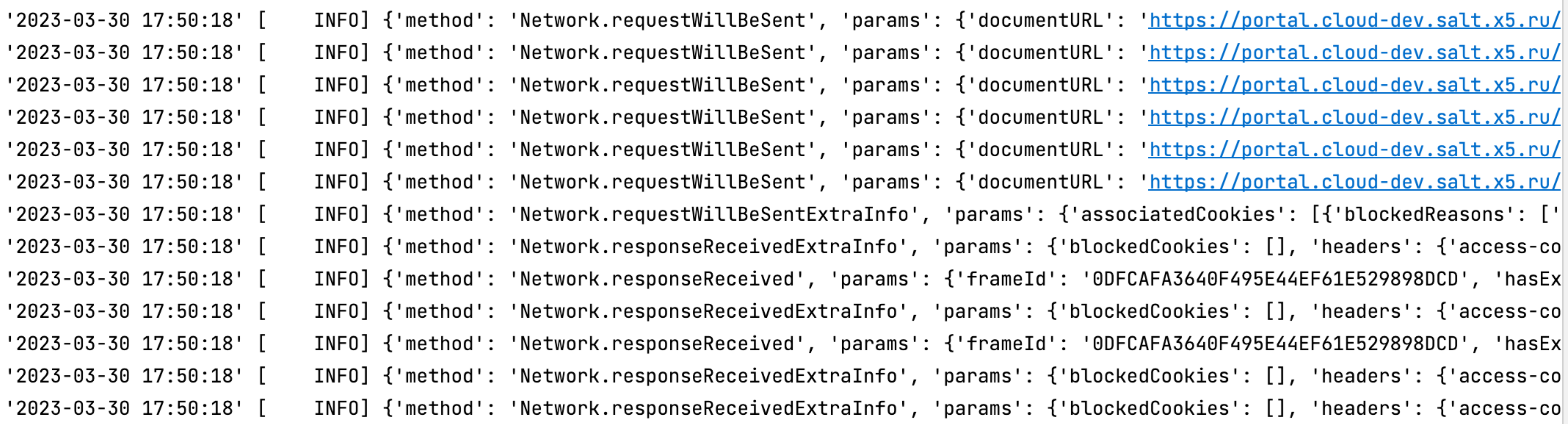
s3.py

```
class S3Page(BasePage):
    @allure.step("Создание бакета")
    def create_bucket(self, bucket):
        self.app.common.press_element_by_text('Создать бакет')
        self.wait(LocatorsGen.popup)
        self.send_keys(LocatorsS3.bucket_name, bucket.name)
        self.wait_for_click(LocatorsGen.create_but)
        self.app.common.press_create_button()
        self.app.navigation.wait_for_line_loader()
```

Работа с браузером

- Запуск на ферме
- Запуск локально

Логирование ошибок и запросов



```
'2023-03-30 17:50:18' [    INFO] {'method': 'Network.requestWillBeSent', 'params': {'documentURL': 'https://portal.cloud-dev.salt.x5.ru/
'2023-03-30 17:50:18' [    INFO] {'method': 'Network.requestWillBeSent', 'params': {'documentURL': 'https://portal.cloud-dev.salt.x5.ru/
'2023-03-30 17:50:18' [    INFO] {'method': 'Network.requestWillBeSent', 'params': {'documentURL': 'https://portal.cloud-dev.salt.x5.ru/
'2023-03-30 17:50:18' [    INFO] {'method': 'Network.requestWillBeSent', 'params': {'documentURL': 'https://portal.cloud-dev.salt.x5.ru/
'2023-03-30 17:50:18' [    INFO] {'method': 'Network.requestWillBeSent', 'params': {'documentURL': 'https://portal.cloud-dev.salt.x5.ru/
'2023-03-30 17:50:18' [    INFO] {'method': 'Network.requestWillBeSent', 'params': {'documentURL': 'https://portal.cloud-dev.salt.x5.ru/
'2023-03-30 17:50:18' [    INFO] {'method': 'Network.requestWillBeSentExtraInfo', 'params': {'associatedCookies': [{'blockedReasons': ['
'2023-03-30 17:50:18' [    INFO] {'method': 'Network.responseReceivedExtraInfo', 'params': {'blockedCookies': [], 'headers': {'access-co
'2023-03-30 17:50:18' [    INFO] {'method': 'Network.responseReceived', 'params': {'frameId': '0DFCAFA3640F495E44EF61E529898DCD', 'hasEx
'2023-03-30 17:50:18' [    INFO] {'method': 'Network.responseReceivedExtraInfo', 'params': {'blockedCookies': [], 'headers': {'access-co
'2023-03-30 17:50:18' [    INFO] {'method': 'Network.responseReceived', 'params': {'frameId': '0DFCAFA3640F495E44EF61E529898DCD', 'hasEx
'2023-03-30 17:50:18' [    INFO] {'method': 'Network.responseReceivedExtraInfo', 'params': {'blockedCookies': [], 'headers': {'access-co
'2023-03-30 17:50:18' [    INFO] {'method': 'Network.responseReceivedExtraInfo', 'params': {'blockedCookies': [], 'headers': {'access-co
```

ОТЧЕТЫ

HEISENBUG X5 Tech

1	Открытие страницы	5s 043ms
suffix	'admin/billing-accounts'	
1.1	Открытие страницы	5s 026ms
2	Клик по первой строке в таблице	512ms
2.1	Клик по элементу	512ms
locator	('xpath', "//*[@data-qa='table-item-link']")	
2.1.1	Ожидание кликабельности элемента	101ms
3	Переход на вкладку	4s 493ms
tab_name	'Возвраты'	
3.1	Клик по элементу	431ms
3.2	Ожидание спиннера	2s 031ms
3.3	Ожидание линейного лоадера	2s 031ms
4	Проверка отсутствия заглушки Что-то пошло не так	43ms
5	Проверка отсутствия заглушки о недостаточности прав	5s 028ms
6	Проверка отсутствия плашки с ошибкой от API	21ms

Data Center:
datapro

Num of selenium workers:
4

PROCESS:
6

Stand:
stage

Suitename:
regress

Fields

Component:
Администратор

Epic:
Библиотека тесткейсов

Feature:
Smoke-тесты

Story:
Отсутствие ошибок и заглуше
к на страницах приложения

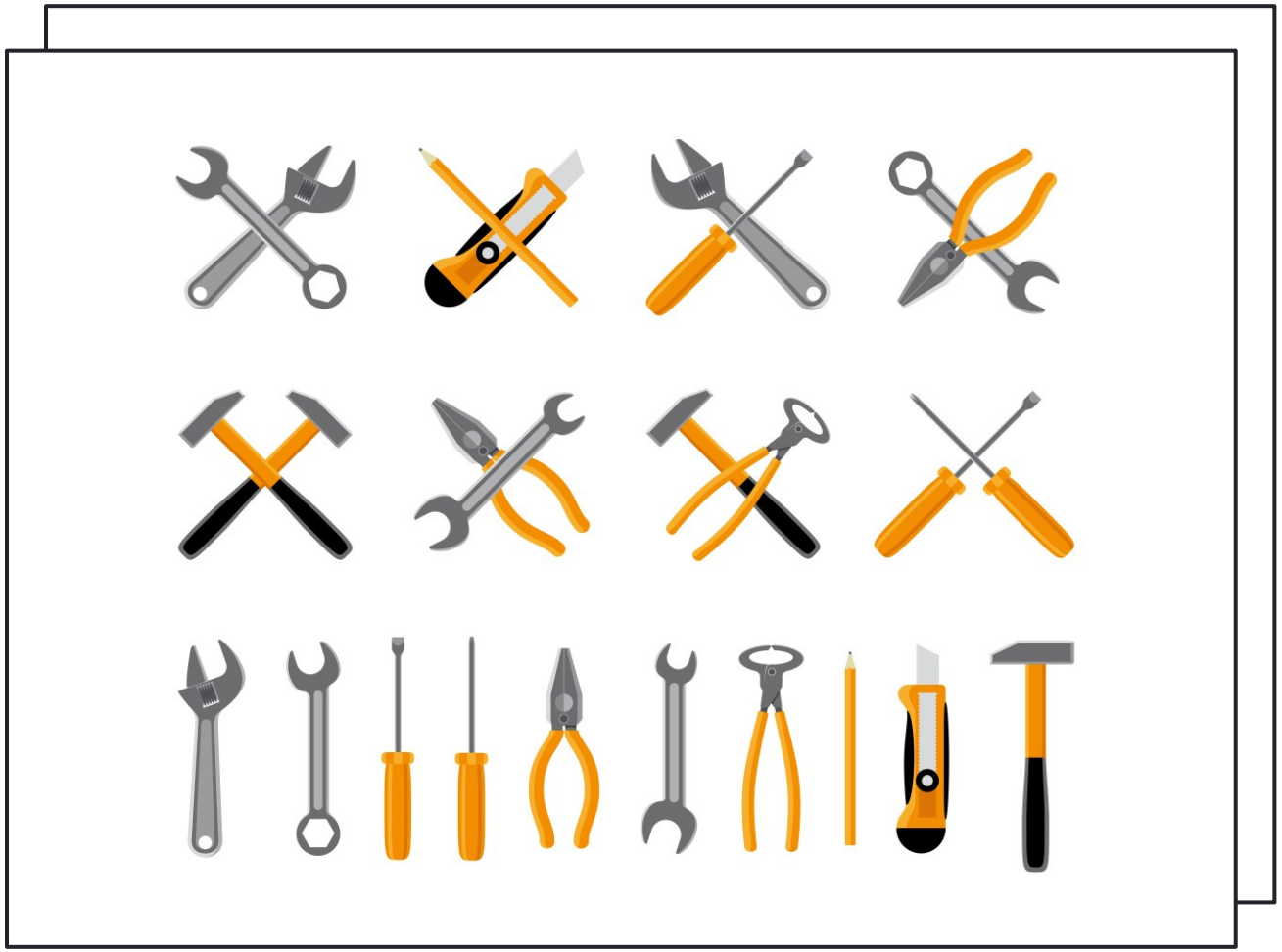
Suite:

X5 Tech

Helpers

Tools

- Keycloak
- Kafka
- SSH
- DB, ClickHouse
- S3



ПЛЮСЫ / МИНУСЫ

- ✓ Универсальные решения
- ✓ Красивая отчетность
- ✓ Снижение времени на разработку

- ✓ Актуализация Pydantic-моделей
- ✓ Много зависимостей
- ✓ Затягивание решения проблем

Шаблонный **проект**

Шаблонный проект

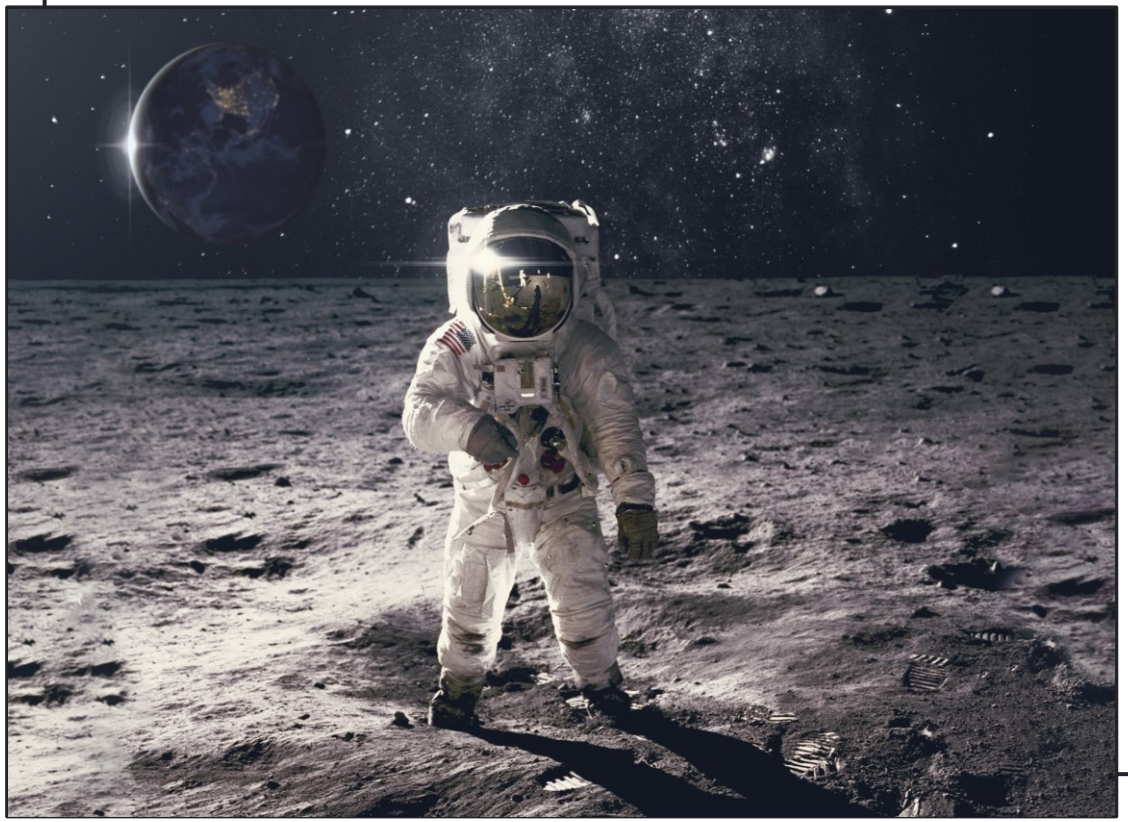
это тестовый проект с целевой архитектурой и примерами, созданный для упрощения внедрения, тиражирования и сопровождения тестовых проектов

Шаблонный **проект**

1

Облегчит организацию первичной структуры проекта

Актуально при внедрении АТ на проекте «с нуля»

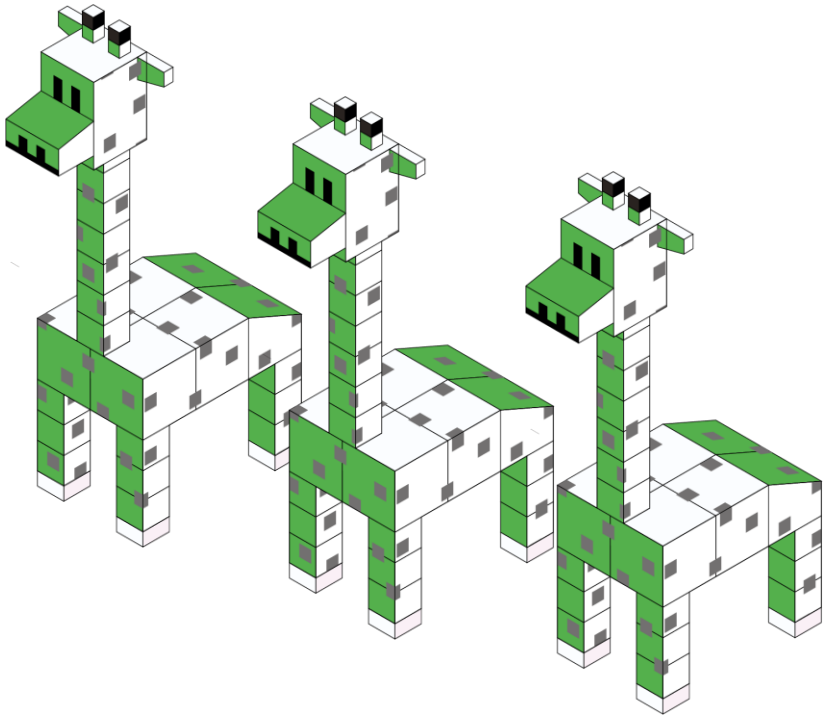


Шаблонный **проект**

2

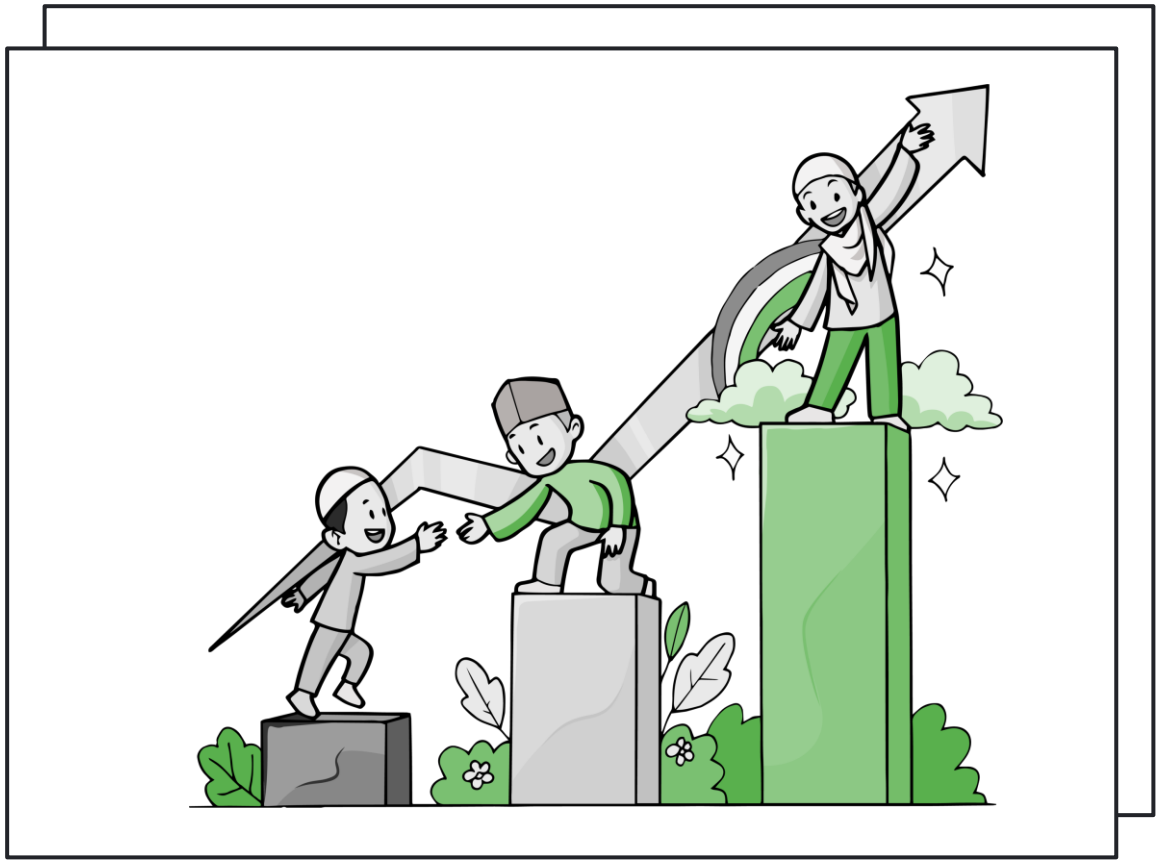
Позволит унифицировать структуру проектов на продуктах компании

Упрощает поддержку проектов в дальнейшем



Шаблонный **проект**

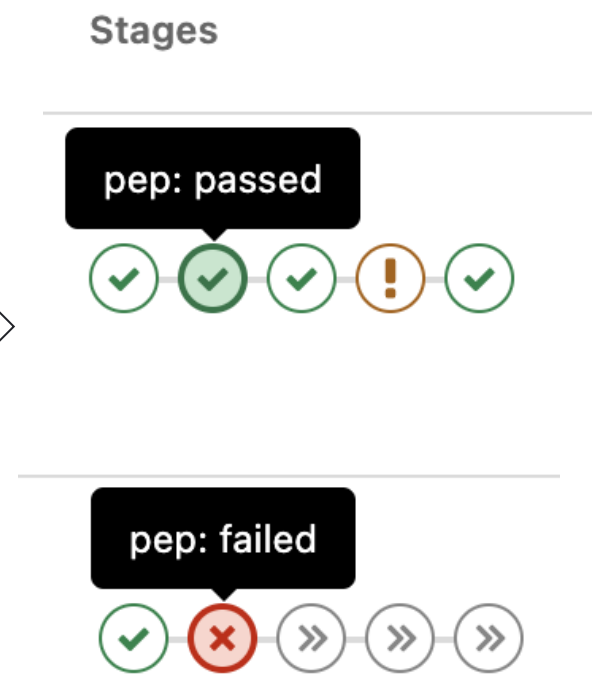
2 **Позволит снизить порог вхождения в АТ**



Шаблонный проект:
статический анализатор кода

flake8
pep8

```
flake8:
  stage: pep
  image: $TEST_PROJECT_IMAGE
  rules:
    - if: ($CI_PIPELINE_SOURCE != "web" &&
$CI_PIPELINE_SOURCE != "pipeline" &&
$CI_PIPELINE_SOURCE != "schedule" &&
$CI_PIPELINE_SOURCE != "trigger" &&
$GITLAB_USER_LOGIN != "allure.service")
  script:
    - python3 -m flake8
```



Шаблонный проект: структура

HEISENBUG X5 Tech

- CI
- models
- test_data
- tests
- utils
- .gitignore
- .gitlab-ci.yml
- README.md
- conf.py
- config_test.json
- conftest.py
- pytest.ini
- requirements.txt
- tox.ini

Типичный сценарий разработки автотестов для проверки API:

1. скачиваем из сваггера json-схему модели и сохраняем ее в файл api-docs.json
2. конвертируем json-схему API в python-модель следующей командой:

```
datamodel-codegen --input api-docs.json --input-file-type auto --output models/model.py --snake-case-field --base-class BaseAPI
```

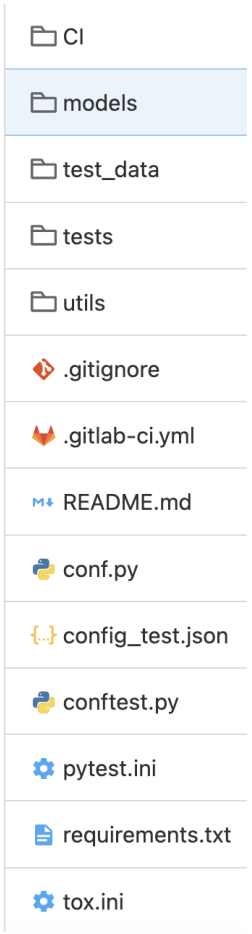
где `api-docs.json` - json-схема API, `models/model.py` - выходной файл модели после конвертации

4. в файле models/model.py импортируем класс BaseAPI из x5qautils: from x5qautils.api import BaseAPI
5. из models/model.py удаляем from **future** import annotations
6. удаляем неиспользуемые и системные методы из полученной модели
7. разбиваем модель на файлы по смыслу (по желанию)
8. пишем контрактные тесты на валидацию модели (пример: test_get_user, test_get_users)
9. дополняем тесты логическими/бизнесовыми проверками (пример: test_create_user)
10. пишем интеграционные апи-тесты (пример: test_create_user_and_get_users_list)

Шаблонный проект: структура

HEISENBUG X5 Tech

x5qautils/api/model.py




```python
class BaseAPI(ABC, BasePydantic):
    def get(self, key, default=None):
        return self.__dict__.get(to_snake_case(key), default)

    def __getitem__(self, item):
        if getattr(self, '__root__', None):
            return self.__root__[item]
        return self.__dict__[to_snake_case(item)]

    def __iter__(self) -> str:
        yield from [key for key in self.__dict__ if not key.startswith('_')]

    def values(self) -> Tuple['BaseAPI']:
        yield from [self.__dict__[key] for key in self]

    def items(self) -> Tuple[str, 'BaseAPI']:
        yield from [(key, self.__dict__[key]) for key in self]
```

Шаблонный проект:
типовые примеры api-тестов

HEISENBUG X5 Tech

tests/api/test_users.py

```
@allure.feature('api/user')
@allure.story('Контрактные тесты')
@pytest.mark.regress
@pytest.mark.user
class TestUsers:
    def test_get_users(self, session):
        session.get('api/users', Users)

    @pytest.mark.asyncio
    async def test_get_users_async(self, session_2):
        await gather_with_concurrency(*(
         session_2.get('api/users/2', User)  for _ in range(10)), n=5)

    def test_create_user(self, session):
        user = UserCreateRequest(
            name='Ivanov',
            job='X5')
        result = session.post('api/users', UserCreateResponce, json=user)
        result.equal(user)
```

Шаблонный проект:
типовые примеры api-тестов

HEISENBUG X5 Tech

tests/api/test_users.py

```
@allure.feature('api/user')
@allure.story('Интеграционные тесты')
@pytest.mark.regress
@pytest.mark.user
class TestUsersIntegration:
    @allure.title("Прверка наличия пользователя в списке
пользователей после создания")
    def test_create_user_and_get_users_list(self, session):
        user = UserCreateRequest(
            name='Holt',
            job='X5')
        session.post('api/users', UserCreateResponce, json=user)
        result = session.get('api/users', Users)
        is_username_in_responce_list(name=user.name, result=result)
```

Шаблонный проект:
типовые примеры ui-тестов

HEISENBUG X5 Tech

tests/ui/test_login.py

```
@allure.feature('Авторизация')
@pytest.mark.regress
@pytest.mark.authorization
class TestLogin:
    @allure.id('99990')
    @allure.title("Авторизация с валидными данными")
    @pytest.mark.parametrize(('login', 'password'), [(APP_USER, APP_PASSWORD)])
    def test_login_valid_user(self, app, login, password):
        app.session.ensure_logout()
        app.navigation.open_homepage()
        app.auth.login(APP_USER, APP_PASSWORD)
        with allure.step('Проверка отображения иконки пользователя'):
            assert app.session.is_user_icon_visible(), "Иконка пользователя не отображается"
```

Шаблонный проект:
типовые фикстуры (общие)

HEISENBUG X5 Tech

conftest.py

```python
@allure.title('Авторизация через api (keycloak)')
@pytest.fixture(scope="session")
def keycloak_api_auth():
    ka = KeycloakAuth(login=APP_USER,
                      passwd=APP_PASSWORD,
                      client_id=CLIENT_ID,
                       client_secret=CLIENT_SECRET,
                      url=BASE_URL)
    return ka.auth_with_secret()
```

```python
# хук для добавления слоеев и кастомных атрибутов в Allure
def pytest_collection_modifyitems(session, config, items):
    for test in items:
        if 'api' in test.fspath.strpath:
            test.add_marker(allure.label("layer", "API"))
        else:
            test.add_marker(allure.label("layer", "UI"))
        test.add_marker(allure.epic("Библиотека тесткейсов"))
        test.add_marker(pytest.mark.regress)
```

Шаблонный проект:
типовые фикстуры (общие)

HEISENBUG X5 Tech

conftest.py

```
def test_param(f):
    def wrap(item):
        full_name = f(item)
        return f'{full_name}_{item.name.split("[", 1)[-1][:-1]}'

    return wrap


# monkey patch для создания параметризированных тестов
отдельными кейсами в Allure
allure_pytest.listener.allure_full_name =
test_param(allure_pytest.listener.allure_full_name)
```

Шаблонный проект: типовые фикстуры (UI)

HEISENBUG X5 Tech

conftest.py

```
@allure.title('Браузер')
@pytest.fixture(scope="session")
def browser(request):
    with allure.step('Запуск браузера'):
        _browser =
Application(browser=request.config.getoption('--browser'),
base_url=APP_URL)
    yield _browser
    with allure.step('Закрытие браузера'):
        _browser.destroy()
```

conftest.py

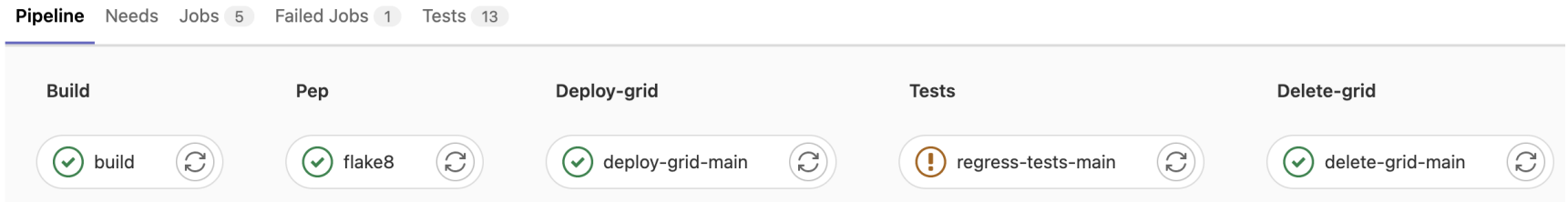
```
@allure.title('Логин')
@pytest.fixture()
def app(browser):
    with allure.step('Логин под юзером по умолчанию'):
        browser.session.ensure_logout()
        browser.session.ensure_login(APP_USER,
APP_PASSWORD)
    return browser

# хук для создания сриншотов при падении тестов
@pytest.mark.tryfirst
@pytest.hookimpl(hookwrapper=True)
def pytest_runtest_makereport(item):
    outcome = yield
    rep = outcome.get_result()
    if rep.failed and 'browser' in item.funcargs:
        item.funcargs['browser'].base.screenshot()
    return rep
```

Шаблонный проект: CI

HEISENBUG X5 Tech

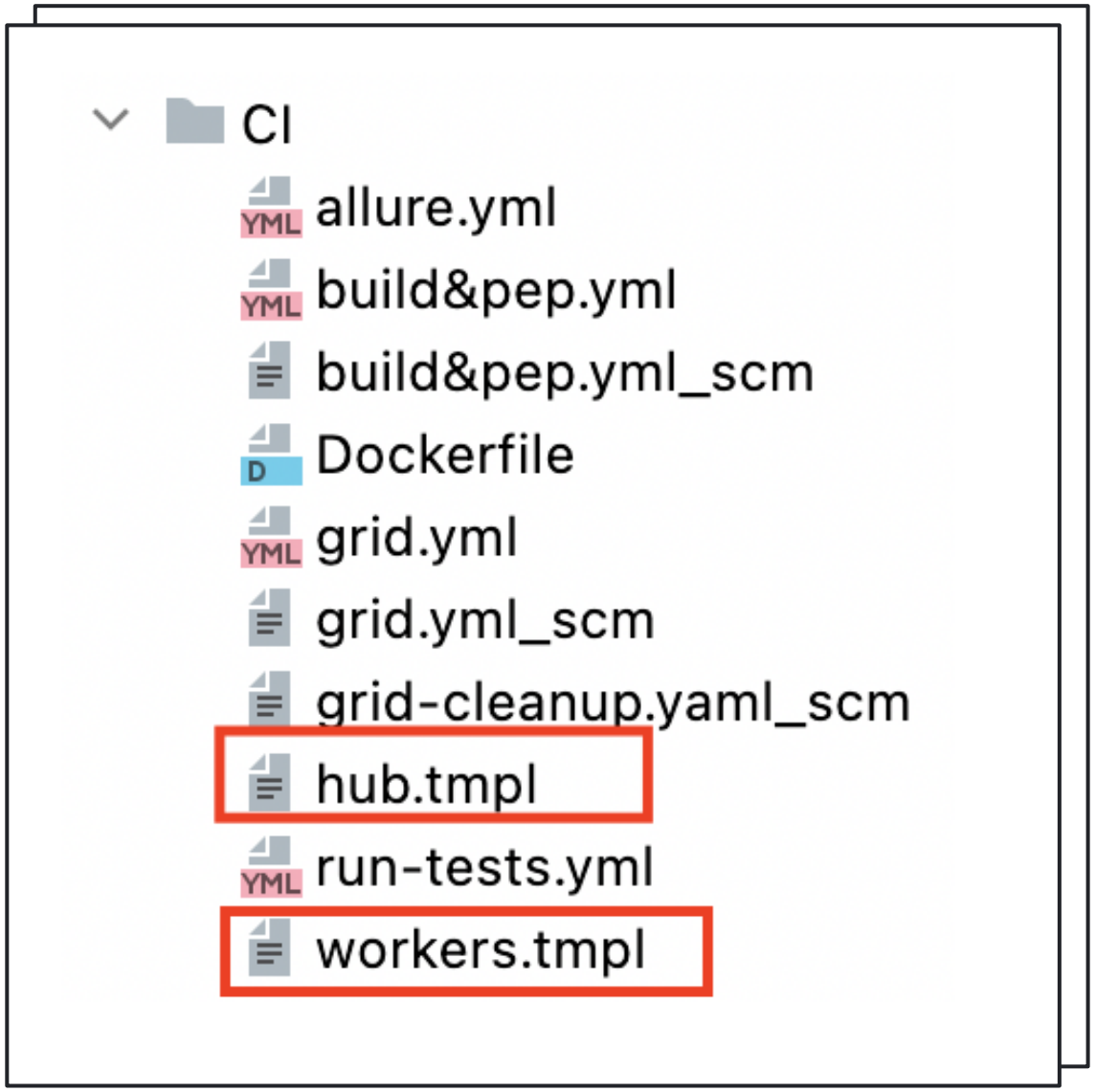
```
variables:
  TEST_PROJECT_IMAGE: ${CI_REGISTRY}/${CI_PROJECT_PATH}:latest
  ALLURE_LAUNCH_NAME: "${CI_PROJECT_NAME} -
${CI_COMMIT_SHORT_SHA} - ${CI_PIPELINE_ID}"
  ALLURE_LAUNCH_TAGS: "${CI_COMMIT_REF_NAME},
${TEST_LABEL}, ${ENV}, ${TAG_NAME},
${CI_RUNNER_SHORT_TOKEN}"
  ALLURE_RESULTS: ${CI_PROJECT_DIR}/allure-results-${TEST_LABEL}
  ALLURE_TESTPLAN_PATH: ./testplan.json
  RERUN_COUNT: 0
#
stages:
  - build
  - pep
  - deploy-grid
  - tests
  - delete-grid
  - allure-export
#
include: 'CI/*.yml'
```

- CI
  - allure.yml
  - build&pep.yml
  - build&pep.yml_scm
  - Dockerfile
  - grid.yml
  - grid.yml_scm
  - grid-cleanup.yaml_scm
  - hub.tmpl
  - run-tests.yml
  - workers.tmpl



Шаблонный проект: Selenium Grid

HEISENBUG X5 Tech



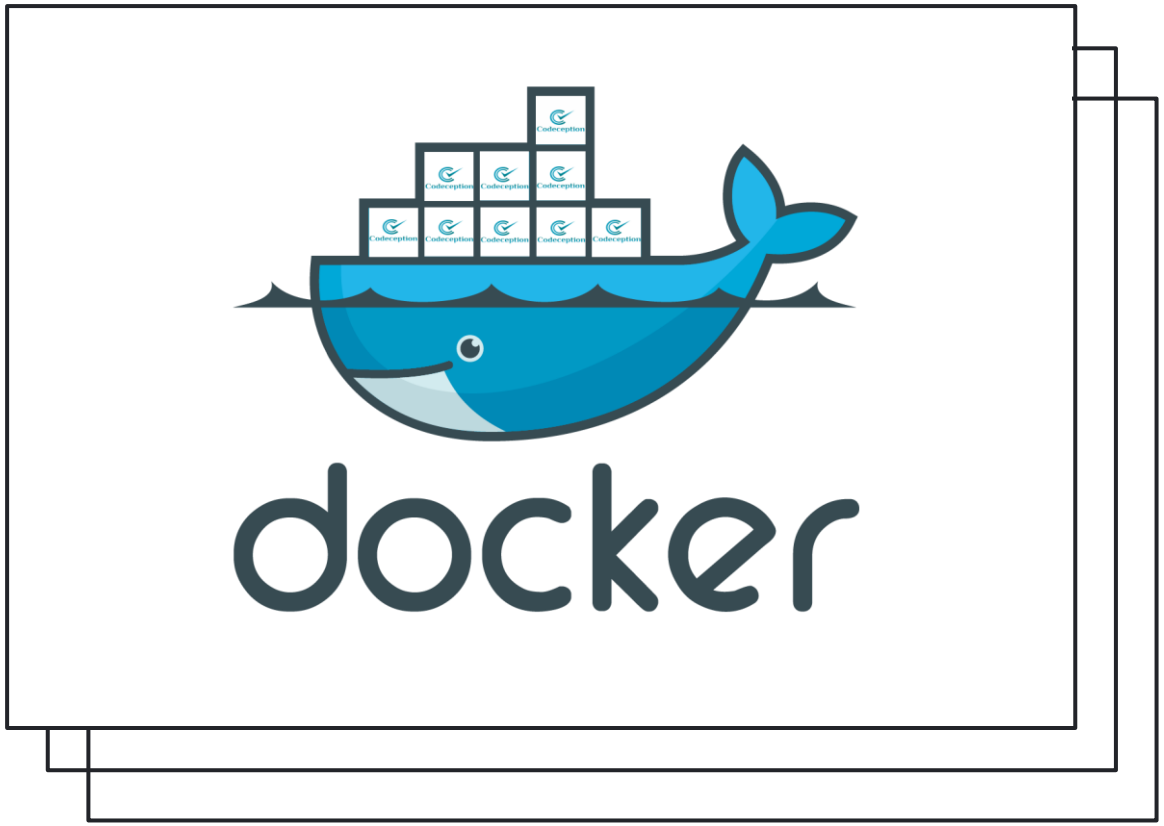
ПЛЮСЫ / МИНУСЫ

- ✓ Идеально для АТ «с нуля»
- ✓ Меньше времени на аудит и поддержку
- ✓ Снижение порога вхождения в АТ

- ✓ Доп. затраты на внедрение не с нуля
- ✓ Может быть много лишнего

Базовые **докер-образы**

- python-tests-base-image
- selenium-hub/workers



Базовые **докер-образы**

python-tests-base-image

- Легкий образ на базе alpine
- Предустановленные утилиты и сертификаты
- Установлены зависимости, включая свежую версию x5qautils

Dockerfile 1.62 KB

```
FROM registry.do.x5.ru/shared/base-containers/alpine/3.17:latest

ARG ALLURECTL_VERSION
ENV ALLURECTL_VERSION=${ALLURECTL_VERSION}

ARG ARTIFACTORY_PYPI_LOGIN
ARG ARTIFACTORY_PYPI_PASS

# install allurectl
RUN wget https://github.com/allure-framework/allurectl/releases/download/${ALLURE
    && chmod +x /usr/bin/allurectl \
    && echo "allurectl version - $ALLURECTL_VERSION" \
    && /usr/bin/allurectl -v

RUN apk add --update nss \
    unzip \
    gcc \
    g++ \
    libxslt-dev \
    libffi-dev \
    jpeg-dev \
    zlib-dev \
    libjpeg \
    musl-dev \
```

Базовые **докер-образы**

Selenium hub/workers

- Zalenuim
- Selenoid
- Moon

ПЛЮСЫ / МИНУСЫ

- ✓ Универсальные решения
- ✓ Требует меньше хард-скилов
- ✓ Стабилизация CI/CD

- ✓ Версионирование и зоопарк браузеров
- ✓ Условно динамический Grid

X5 Tech

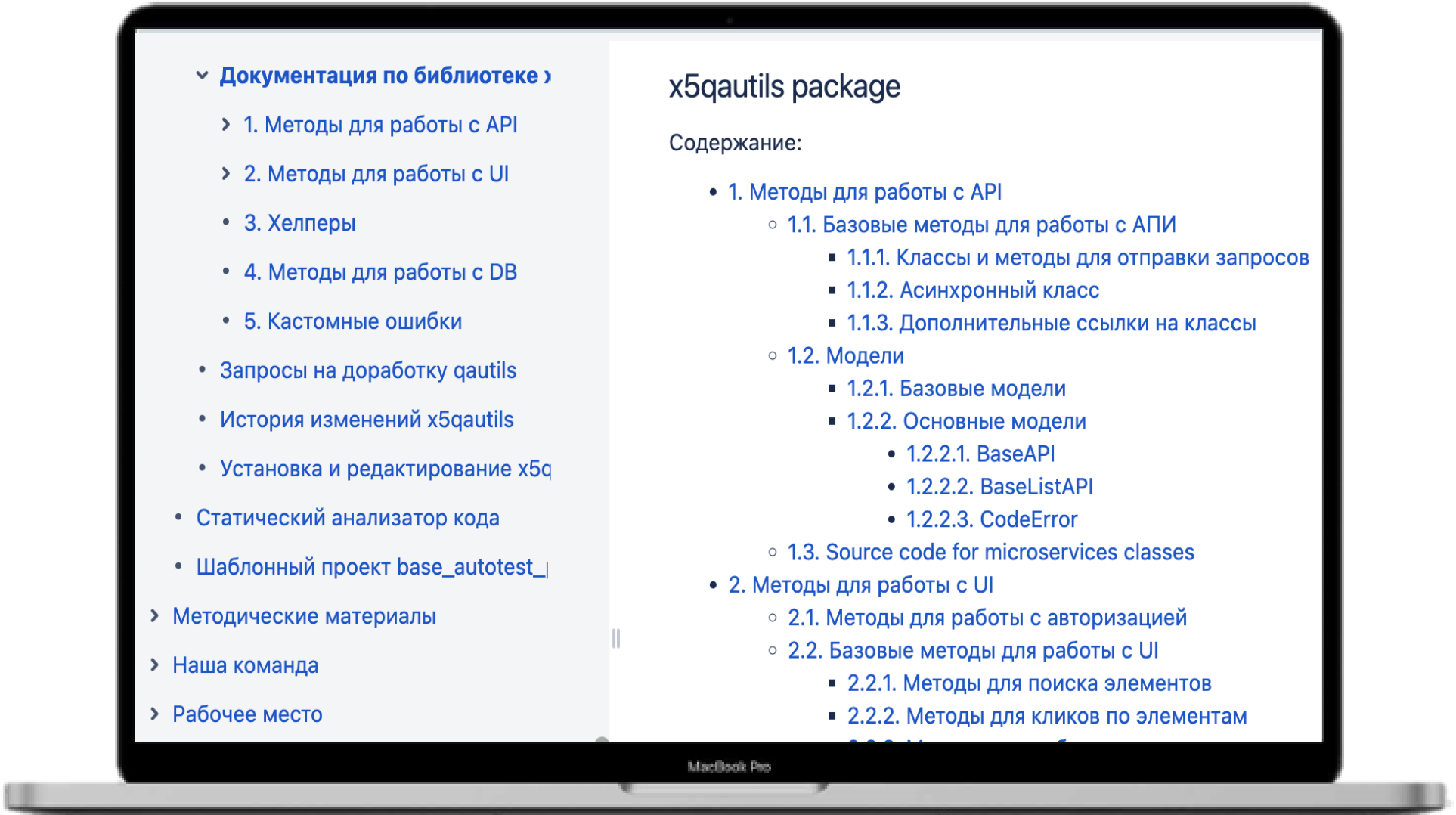
И как этим пользоваться?

Документация

- ☑ **Wiki**
- ☑ **Readme**

```
Классы и методы для отправки запросов
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

:py:mod:`x5qautils.api.microservices`


Класс для отправки запросов:
:py:class:`.MicroservicesTemplate` (or
:py:class:`~x5qautils.api.QaSession`)
    :Parameters:
        :base_url: Урл …

    :Types:
        :base_url: str or furl

    :Методы:

:py:meth:`~.MicroservicesTemplate.get`

   :py:meth:`~.MicroservicesTemplate.post`
```

Документация: wiki



Документация: wiki

Хранение образа тестового проекта в artifactory

Создатель Snopok, Mariya, отредактировано только что

ЗАЧЕМ ЭТО НУЖНО?

1. для возможности встраивания автотестов в пайплайны на разных доменах (scm.x5.ru и git.do.x5.ru, например)
2. если нет возможности хранить image проекта с автотестами в gitlab ci registry (пользователи https://scm.x5.ru , например)

ЧТО НУЖНО СДЕЛАТЬ?

1. для авторизации в докер-репозиторий добавить CI/CD переменную DOCKER_AUTH_CONFIG

DOCKER_AUTH_CONFIG

```
{
    "auths": {
      "docker-qa-at-prod.x5.ru": {
        "auth": "${username}:${password} in base64"
      }
    }
  }
```

Документация: readme

README.md

Базовый проект-пример для разработки автотестов

Проект подходит для разработки автотестов с нуля: нужно склонировать проект и использовать его в качестве шаблона с целевой структурой и настроенным CI. Склонированный проект нужно наполнять автотестами по аналогии с примерами из проекта.

Структура проекта:

`CI/Dockerfile` - файл для создания image

HEISENBUG X5 Tech

Профит

- Универсальные решения

HEISENBUG X5 Tech

Профит

- Универсальные решения
- Красивая отчетность

HEISENBUG X5 Tech

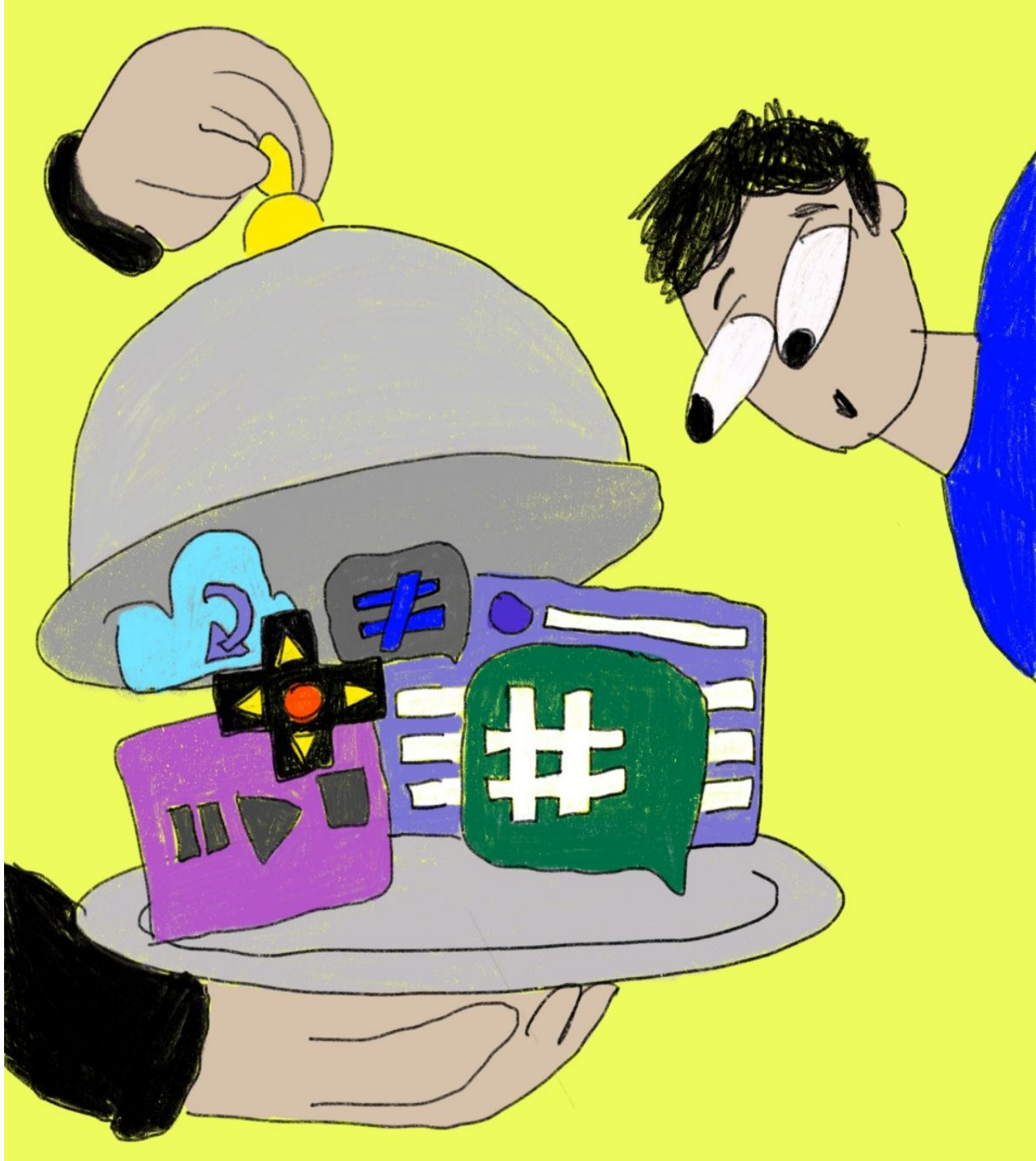
Профит

- Универсальные решения
- Красивая отчетность
- Экономия времени на внедрение и поддержку автотестов

HEISENBUG X5 Tech

Профит

- Универсальные решения
- Красивая отчетность
- Экономия времени на внедрение и поддержку автотестов
- Снижение порога вхождения в АТ



Проблемы **внедрения**

Не обязательно – можно забить

Проблемы **внедрения**

Багаж автотестов

Проблемы **внедрения**

**Есть свое решение,
не хочу ничего менять**

HEISENBUG X5 Tech

Проблемы **поддержки**

Мейнтейнер

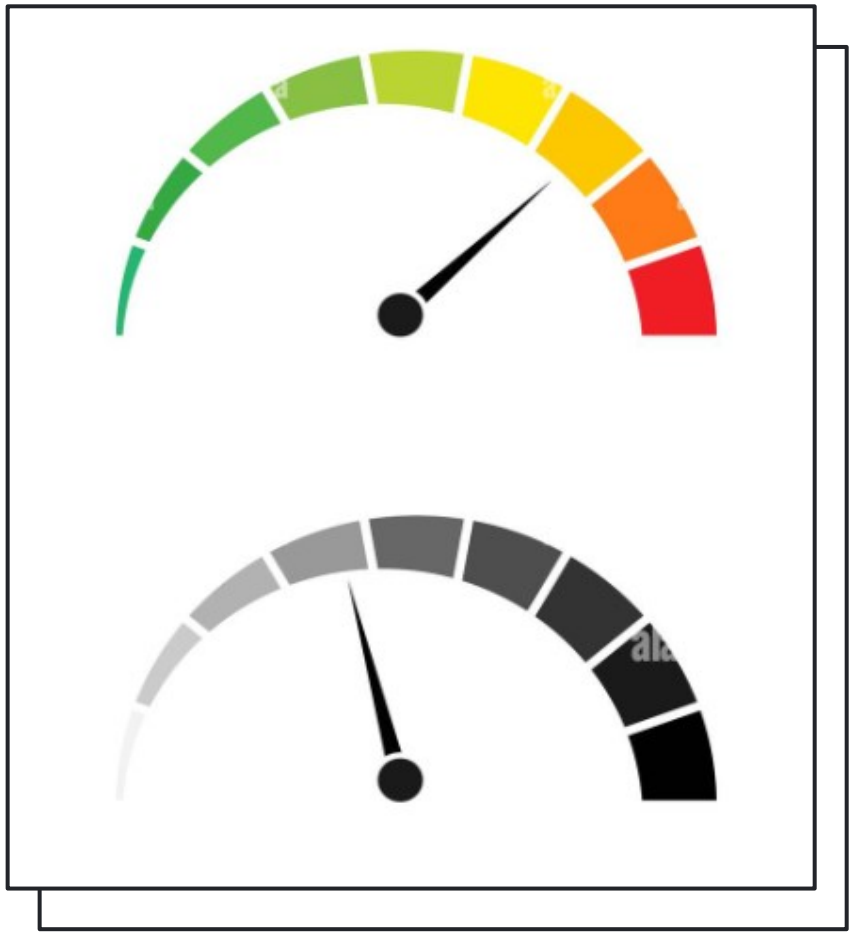
Что дальше? Развиваем x5qautils

1. **Автогенерация тестов из сваггер-документации**

Что дальше? Развиваем x5qautils

1. Автогенерация тестов из сваггер-документации

2. НТ (Первичное)

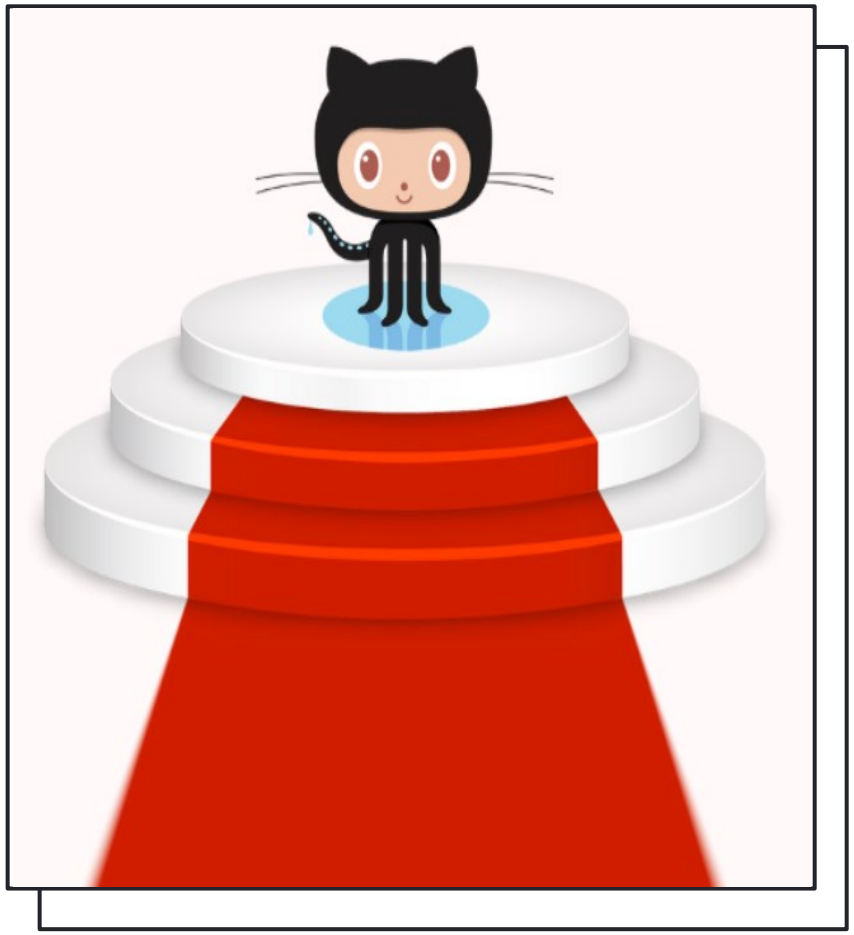


Что дальше? Развиваем x5qautils

1. Автогенерация тестов из сваггер-документации
2. НТ (Первичное)
3. Мобилки

Что дальше? Развиваем x5qautils

1. Автогенерация тестов из сваггер-документации
2. НТ (Первичное)
3. Мобилки
4. Выход в публичку



Что дальше? Развиваем x5qautils

1. Автогенерация тестов из сваггер-документации
2. НТ (Первичное)
3. Мобилки
4. Выход в публичку
5. Универсализация для других фреймов

ВЫВОД

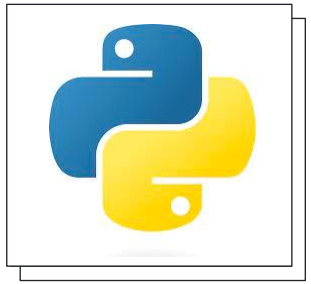
HEISENBUG X5 Tech

- ✓ Крупная компания
- ✓ > 6 проектов
- ✓ Профессиональная поддержка
- ✓ > 3-х мейнтейнеров

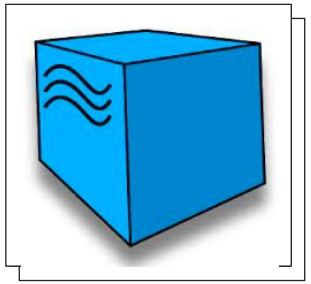
- ✓ Небольшая компания
- ✓ <= 6 проектов
- ✓ Нет профессиональной поддержки
- ✓ 1-2 мейнтейнера

Может быть **полезно**

HEISENBUG X5 Tech



Selene



Selenoid/Moon

Schemathesis

Pydantic

Спасибо за внимание!

Контакты

Maria.Snopok@x5.ru

Vladislav.Grigorev@x5.ru

@maryiasnapok

@Vaindante