#### ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ «ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет компьютерных наук

Департамент программной инженерии

|  |  |
| --- | --- |
| СОГЛАСОВАНО  Руководитель проекта, старший преподаватель факультета компьютерных наук департамента «Программная инженерия» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ М. Д. Шадрин  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2021 г. | УТВЕРЖДАЮ  Академический руководитель образовательной программы «Программная инженерия», канд. техн. наук, профессор  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.В. Шилов  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2021 г. |
| |  |  | | --- | --- | | Подп. и дата |  | | Инв. № дубл. |  | | Взам. Инв. № |  | | Подп. и дата |  | | Инв. № подл. |  | | **Серверная часть мобильного приложения для аниме-сообщества**  **Пояснительная записка**  **ЛИСТ УТВЕРЖДЕНИЯ**  **RU.17701729.02.06-01 01-1-ЛУ**  **Исполнитель**  Студент группы БПИ 199  \_\_\_\_\_\_\_\_\_\_\_/В. А. Федченко/  «\_\_\_»\_\_\_\_\_\_\_\_\_\_ 2021 г. | |  |

**УТВЕРЖДЕН**

**RU.17701729.02.06-01 81 01-1 ЛУ**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Подп. и дата |  | | Инв. № дубл. |  | | Взам. Инв. № |  | | Подп. и дата |  | | Инв. № подл. |  | | **Серверная часть приложения для аниме-сообщества  Пояснительная записка  RU.17701729.02.06-01 01-1**  **Листов 19** |  |

СОДЕРЖАНИЕ

[1. ВВЕДЕНИЕ 3](#_Toc72110265)

[1.1. Наименование программы 3](#_Toc72110266)

[1.2. Документы, на основании которых ведется разработка 3](#_Toc72110267)

[2. НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ 4](#_Toc72110268)

[2.1. Функциональное назначение 4](#_Toc72110269)

[2.2. Эксплуатационное назначение 4](#_Toc72110270)

[3. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ 5](#_Toc72110271)

[3.1. Постановка задачи на разработку программы 5](#_Toc72110272)

[3.1.1. Требования к составу выполняемых функций 5](#_Toc72110273)

[3.1.2. Требования к организации входных данных 5](#_Toc72110274)

[3.1.3. Требования к организации выходных данных 5](#_Toc72110275)

[3.2. Описание алгоритмов и функционирования программы 5](#_Toc72110276)

[3.2.1. Алгоритм программы 5](#_Toc72110277)

[3.2.2. Архитектура программы 6](#_Toc72110278)

[3.2.2. API сервера 7](#_Toc72110279)

[3.3. Описание и обоснование выбора метода организации входных и выходных данных 9](#_Toc72110280)

[3.3.1. Описание метода организации входных и выходных данных 9](#_Toc72110281)

[3.4. Требования к составу и параметрам технических средств 9](#_Toc72110282)

[4. ОЖИДАЕМЫЕ ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ 10](#_Toc72110283)

[4.1. Предполагаемая потребность 10](#_Toc72110284)

[4.2. Экономические преимущества разработки по сравнению с отечественными и зарубежными аналогами 10](#_Toc72110285)

[5. ИСТОЧНИКИ, ИСПОЛЬЗОВАННЫЕ ПРИ РАЗРАБОТКЕ 11](#_Toc72110286)

[Приложение 1 12](#_Toc72110287)

[Приложение 2 14](#_Toc72110288)

[ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ 19](#_Toc72110289)

# 1. ВВЕДЕНИЕ

## 1.1. Наименование программы

Наименование программы на русском языке – «Серверная часть приложения для аниме-сообщества»

Наименование программы на английском языке – «Server Part of Mobile Application For Anime Community»

## 1.2. Документы, на основании которых ведется разработка

Приказ декана факультета компьютерных наук И.В. Аржанцева "Об утверждении тем, руководителей курсовых работ студентов образовательной программы «Программная инженерия» факультета компьютерных наук" № 2.3-02/1112-04 от 11.12.2019

# 2. НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ

## 2.1. Функциональное назначение

Приложение предназначено для удобного просмотра новостей, связанных с аниме, а комфортного чтения манги, манхвы, маньхуы в одном мобильном приложении. Все манги, манхвы, маньхуы взяты из открытых источников.

Серверная составляющая приложения предназначена для хранения базы данных пользователей и файлов с мангой, реализации логики приложения, работы со сторонними сайтами, с которых берется необходимая информация, предоставления клиенту API для взаимодействия с сервером

## 2.2. Эксплуатационное назначение

Приложение предназначено для предоставления людям, увлекающимся аниме и мангой, возможности в одном приложении читать мангу и одновременного получения актуальных новостей.

# 3. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

## 3.1. Постановка задачи на разработку программы

### 3.1.1. Требования к составу выполняемых функций

Программа должна иметь API, при помощи которого клиент сможет получать информацию о манге и ее главы, получать необходимую информацию о пользователе, вызывать функции для взаимодействия с пользователем. API должен содержать:

* POST запрос для регистрации пользователя
* POST запрос аутентификации пользователя
* POST запрос для смены пароля пользователя
* POST запрос для установления пользователю любимой манги
* POST запрос для получения любимой манги пользователя
* GET запрос для получения обложки и основной информации о манге.
* GET запрос для глав манги
* GET запрос для получения новой манги
* GET запрос для получения популярной манги

**3.1.2. Требования к организации входных данных**

Для корректной работы сервера входные данные должны приходить с помощью HTTP-запроса и иметь формат JSON.

**3.1.3. Требования к организации выходных данных**

В качестве выходных данных сервер должен давать HTTP-ответ в формате JSON.

## 3.2. Описание алгоритмов и функционирования программы

### 3.2.1. Алгоритм программы

После запуска сервер разворачивается на указанном пользователем локальном ip адресе и сетевом порту и начинает ждать запросы от клиента. Запросы от клиента делятся на 2 кластера: запросы о пользователе и запросы о манге. Разберем обе группы запросов.

Запросы о пользователе:

К запросам о пользователе можно причислить

* Запрос на регистрацию нового пользователя
* Запрос на аутентификацию пользователя
* Запрос на смену пароля пользователя
* Запрос на установку пользователю любимой манги
* Запрос на получение любимой манги пользователя

При запросе на регистрацию сервер проверяет корректность введенной почты (обертка над встроенной в DRF функцией) и уникальность никнейма и email адреса пользователя. Если данные email адреса некорректны, сервер отлавливает ошибку, возвращаемую встроенной функцией, после чего возвращает сообщение об ошибке клиенту и продолжает работать дальше. При успешном прохождении всех проверок сервер регистрирует пользователя в базе данных и возвращает HTTP-ответ с положительным статусом. В случае, если какая-либо из проверок была не пройдена, сервер возвращает HTTP-ответ с ошибкой и описанием ошибки. Для аутентификации пользователя, смены его пароля или получения/задания любимой манги, необходимо прислать данные пользователя (логин и пароль), после чего будет произведена попытка авторизовать пользователя, и в случае успеха возвращен HTTP-ответ с соответствующим содержанием.

Запросы о манге:

К запросам о пользователе можно причислить

* Запрос на получение информации о манге
* Запрос на получение страницы манги
* Запрос на получение главы манги
* Запрос на получение популярной манги
* Запрос на получение новой манги

При получении любого из этих запросов в случае, если манга (вся популярная, новая манга) не была скачана до этого, серверу необходимо обратиться к стороннему ресурсу, размещающему у себя мангу. Во время разработки возникла проблема со сканированием сайта и получения информации, связанная с вычислением сервисом Google выполняющегося скрипта и выдачи ему CAPTCH’и вместо необходимой информации. Для обхода CAPTCH’и пришлось имитировать браузер Google Chrome, а также после каждого запроса закрывать и заново открывать браузер с новым идентификатором пользователя. Данное решение успешно решает проблему выдачи CAPTCH’и сервисом Google, однако является весьма долгим из-за физической имитации браузера. Для ускорения работы сервера, мною используется файловая система хранения информации о манге, а также использование многопоточности для одновременного скачивания нескольких страниц.

В случае, если манга уже была скачана до этого, сервер незамедлительно отдает файл с информацией, минуя этап скачивания.

### 3.2.2. Архитектура программы

Сервер написан на языке Python версии 3.9.5 с использованием Django framework и базы данных Sqlite3. Проект имеет стандартную авто сгенерированную структуру DRF проекта, а именно:

admin.py – файл, в котором описана модель администратора. В нее входит информация об элементах, необходимых к отображению в панели администратора, фильтрах, применяемых к базе пользователей и т.п.

apps.py – конфигурация приложения

forms.py – файл, содержащий формы, используемые в проекте. Содержит в себе форму для регистрации пользователя через браузер. В проекте использовалась исключительно для удобства разработки на начальных этапах.

models.py – файл, содержащий все модели. Хранит в себе класс пользователя, унаследованный от django.contrib.auth.models.AbstractUser и содержащий в себе дополнительное поле любимой манги.

views.py – файл, содержащий логику работы сервера со сторонними источниками и клиентом через API.

urls.py – файл, содержащий в себе связь между эндпоинтами и методами, обрабатывающими запросы по данным ссылкам.

settings.py – файл с настройками базы данных, моделей, установленных приложений и т.д

### 3.2.2. API сервера

**Пользователь:**

/register/post/

POST – регистрирует пользователя. На вход подается JSON с никнеймом, email адресом и паролем. Ответом является JSON со статусом запроса. Если пользователь успешно зарегистрирован, то статус будет “OK”, в противном случае “ERROR”. ERROR может возникать в 3 случаях:

* Неправильная почта
* Пользователь с таким никнеймом уже существует
* Пользователь с таким email уже существует

Более детально изучить информацию об ошибке можно в поле “description” JSON файла.

/authenticate/

POST – авторизовать пользователя. На вход подается JSON с никнеймом и паролем. Ответом является JSON со статусом запроса. Если пользователь успешно авторизован, то статус будет “OK”, в противном случае “ERROR”. ERROR может возникнуть только в том случае, если данные для авторизации были указаны неверно.

/change\_password/

POST – сменить пароль пользователя. На вход подается JSON с никнеймом, старым паролем и новым паролем. Ответом является JSON со статусом запроса. Если пользователю был успешно сменен пароль, то статус будет “OK”, в противном случае “ERROR”. ERROR может возникнуть только в том случае, если данные для авторизации были указаны неверно.

/set\_favourite\_manga/

POST – Задать пользователю список из любимых манг. На вход подается JSON с никнеймом, паролем и листом манги, которую необходимо задать пользователю как любимую. Ответом является JSON со статусом запроса. Если пользователю была успешно задана любимая манга, то статус будет “OK”, в противном случае “ERROR”. ERROR может возникнуть только в том случае, если данные для авторизации были указаны неверно.

/get\_favourite\_manga/

POST – Получить список из любимых манг пользователя. На вход подается JSON с никнеймом и паролем. Ответом является JSON со статусом запроса и списком любимой манги пользователя. Если пользователю была успешно задана любимая манга, то статус будет “OK”, в противном случае “ERROR”. ERROR может возникнуть только в том случае, если данные для авторизации были указаны неверно.

**Манга:**

/getmanga/

Получить основную информацию о манге. На вход подается JSON с кодом манги. Ответом является JSON с названием манги, описанием, жанрами, картинкой обложки и списком всех глав с названиями.

/getmanga/page/

Получить страницу главы манги. На вход подается JSON с кодом манги, номером тома, главы и страницы. Ответом является JSON со страницей манги, закодированной в base64 и общим количеством страниц в главе.

/getmanga/chapter/

Получить главу манги. На вход подается JSON с кодом манги, номером тома и главы. Ответом является JSON со ссылками на все страницы главы данной маги.

/ongoings/

Получить популярную мангу. Ответом является JSON со списком популярных манг и краткой информации о них.

/new/

Получить новую мангу. Ответом является JSON со списком популярных манг и краткой информации о них.

## 3.3. Описание и обоснование выбора метода организации входных и выходных данных

### 3.3.1. Описание метода организации входных и выходных данных

Входными данными для сервера являются HTTP-запросы типа POST, содержащие в себе тело в формате JSON

Выходными данными для сервера являются HTTP-ответы, содержащие в себе тело в формате JSON

### 3.3.2. Обоснование выбора метода организации входных и выходных данных

В качестве формата для передачи данных был выбран JSON, так как он легко читается человеком, а также очень распространен, благодаря чему существует множество библиотек, позволяющих удобно работать с этим форматом данных.

## 3.4. Требования к составу и параметрам технических средств

Для корректного запуска сервера необходим компьютер с операционной системой Windows 10 или выше, интерпретатор языка Python версии не ниже 3.9.5, база данных Sqlite3, установленный браузер Google Chrome, веб-драйвер браузера Google Chrome совпадающий с версией установленного на ПК вышеназванного браузера, а также следующие дополнительно подключенные библиотеки:

* Selenium 3.141.0
* Requests 2.25.1
* BeautifulSoup4 4.9.3
* Fake-useragent 0.1.11
* Pathlib 1.0.1
* Drf-yasg 1.20.0
* Django-rest-swagger 2.2.0
* Urllib3 1.26.4

В качестве альтернативы можно в терминале прописать команду *pip install -r requirements.txt* для установки всех необходимых библиотек

## 3.5. Требования к отказоустойчивости системы

В случае возникновения ошибки сервер выдает html страницу с описанием ошибки и продолжает свое нормальное функционирование. В случае возникновения критической ошибки, после которой дальнейшая работа сервера невозможна, достаточно перезапустить сервер.

# 4. ОЖИДАЕМЫЕ ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ

## 4.1. Предполагаемая потребность

В последнее десятилетие наблюдается явная тенденция к увеличению популярности Японских, Китайских и Корейских комиксов и аниме. Некоторые произведения японской мультипликации наравне с фильмами показывают в кинотеатрах, а медиа гиганты, такие как Netflix, выкупают права для показа данных произведений на своих платформах. Данная программа нацелена на растущую аудиторию любителей аниме и манги, которым необходимо удобное приложение для чтения и просмотра новостей.

## 4.2. Экономические преимущества разработки по сравнению с отечественными и зарубежными аналогами

Аналогами данного приложения являются такие приложения как:

* Мобильное приложение “Манга-чан”
* Мобильное приложение “MangaL”

В сравнении с аналогами приложение обладает следующими преимуществами:

* минималистичный, интуитивно понятный интерфейс
* возможность создания списка избранной манги
* новостная лента
* возможность загрузить мангу и читать ее без доступа к интернету
* Бесплатность, отсутствие рекламы.

# 5. ИСТОЧНИКИ, ИСПОЛЬЗОВАННЫЕ ПРИ РАЗРАБОТКЕ

1. ГОСТ 19.101-77 Виды программ и программных документов. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
2. ГОСТ 19.301-79 Пояснительная записка. Требования к содержанию и оформлению. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
3. ГОСТ 19.102-77 Стадии разработки. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
4. ГОСТ 19.103-77 Обозначения программ и программных документов. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
5. ГОСТ 19.105-78 Общие требования к программным документам. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
6. ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
7. ГОСТ 19.604-78 Правила внесения изменений в программные документы, выполненные печатным способом. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
8. Документация по Django-rest-framework [Электронный ресурс] -URL: https://www.django-rest-framework.org (Дата обращения:15.05.2021, режим доступа: свободный).
9. Документация по Django Framework 3.2 [Электронный ресурс] -URL https://docs.djangoproject.com/en/3.2/ (Дата обращения: 15.05.2021, режим доступа: свободный)
10. Документация по Selenium [Электронный ресурс] -URL https://www.selenium.dev/documentation/en/ (Дата обращения: 15.05.2021, режим доступа: свободный)
11. Документация по BeautifulSoup4 [Электронный ресурс] -URL https://www.crummy.com/software/BeautifulSoup/bs4/doc/ (Дата обращения: 15.05.2021, режим доступа: свободный)
12. Документация по Requests [Электронный ресурс] -URL https://docs.python-requests.org/en/master/ (Дата обращения: 15.05.2021, режим доступа: свободный)
13. Документация по Fake-useragent [Электронный ресурс] -URL https://fake-useragent.readthedocs.io/en/latest/ (Дата обращения: 15.05.2021, режим доступа: свободный)
14. Документация по Pathlib [Электронный ресурс] -URL https://docs.python.org/3/library/pathlib.html (Дата обращения: 15.05.2021, режим доступа: свободный)
15. Документация по Drf-yasg [Электронный ресурс] -URL https://drf-yasg.readthedocs.io/en/stable/ (Дата обращения: 15.05.2021, режим доступа: свободный)
16. Документация по Django-rest-swagger [Электронный ресурс] -URL https://django-rest-swagger.readthedocs.io/en/latest/ (Дата обращения: 15.05.2021, режим доступа: свободный)
17. Документация по Urllib3 [Электронный ресурс] -URL https://urllib3.readthedocs.io/en/latest/ (Дата обращения: 15.05.2021, режим доступа: свободный)

# Приложение 1

**Основные определения**

**Манга** –японские комиксы.

**Манхва** – корейские комиксы

**Маньхуа** – китайские комиксы

**Аниме** – японская мультипликация.

**Тайтл** – название аниме.

**Никнейм** – псевдоним, который указывает пользователь при регистрации.

**Запрос** – удаленная часть приложения, которая содержит в себе логику приложения и обрабатывает запросы от клиента посредством API

**Сервер** – программа, обрабатывающая запросы клиента.

**Клиент** – часть приложения, непосредственно взаимодействующая с пользователем, обрабатывающая и отсылающая на сервер через API поступающие данные и команды.

**API (Application programming interface)** – это составляющая часть сервера, которая получает запросы и отправляет ответы.

**DRF (Django Rest Framework)** – фреймворк для быстрой разработки интерфейса на языке python

**JSON** – текстовый формат обмена данными, основанный на JavaScript.

**IP-адрес** (Internet Protocol) – уникальный числовой [идентификатор](https://ru.wikipedia.org/wiki/%D0%98%D0%B4%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%82%D0%BE%D1%80) устройства в компьютерной [сети](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D1%81%D0%B5%D1%82%D1%8C)

**Сетевой порт** – целое неотрицательное число, записываемое в заголовках [протоколов](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB_%D0%BF%D0%B5%D1%80%D0%B5%D0%B4%D0%B0%D1%87%D0%B8_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) [транспортного уровня](https://ru.wikipedia.org/wiki/%D0%A2%D1%80%D0%B0%D0%BD%D1%81%D0%BF%D0%BE%D1%80%D1%82%D0%BD%D1%8B%D0%B9_%D1%83%D1%80%D0%BE%D0%B2%D0%B5%D0%BD%D1%8C) сетевой [модели OSI](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%82%D0%B5%D0%B2%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_OSI)

**HTTP** – протокол передачи данных.

**Эндпоинт** **(от англ. Endpoint)** – Конечная точка связи, ссылка, обращаясь по которой клиент может получить определенную информацию.

**CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart - полностью автоматизированный публичный** [**тест Тьюринга**](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D1%81%D1%82_%D0%A2%D1%8C%D1%8E%D1%80%D0%B8%D0%BD%D0%B3%D0%B0) **для различения компьютеров и людей)** – компьютерный тест, используемый для того, чтобы определить, кем является пользователь системы: человеком или компьютером.

# Приложение 2

**Описание и функциональное назначение классов**

|  |  |
| --- | --- |
| **Класс** | **Назначение** |
| User | Класс, описывающий пользователя. Является моделью для таблицы в базе данных, в которой хранятся данные пользователей. |

# 

# Приложение 3

**Описание и функциональное назначение полей и методов**

В таблице указаны только те методы и поля, которые отсутствовали в базовом классе User или их логика как-либо была изменена.

|  |  |  |
| --- | --- | --- |
| **User** | | |
| **Поля** | | |
| **Название** | **Назначение** | **Значение по умолчанию** |
| favourite\_manga | Поле типа JSON, хранящее в себе информацию о любимой манге пользователя | [] |
| email | Обязательное поле email адреса пользователя. Должно быть уникальным | **-** |

|  |  |  |  |
| --- | --- | --- | --- |
| **Методы** | | | |
| **Название** | **Входные данные** | **Выходные данные** | **Назначение** |
| register | Request – HTTP запрос типа POST в формате JSON | Render – отрисовка формы регистрации пользователя | Форма для регистрации пользователя через браузер. Используется исключительно для разработки и тестирования |
| post\_register | Request – HTTP запрос типа POST в формате JSON | JsonResponse – HTTP ответ в формате JSON | Проверяет данные, пришедшие из request на корректность, и регистрирует пользователя. Если данные некорректны – возвращает сообщение об ошибке |
| post\_auth | Request – HTTP запрос типа POST в формате JSON | JsonResponse – HTTP ответ в формате JSON | Пытается авторизовать пользователя по данным, пришедшим из request. |
| post\_change\_password | Request – HTTP запрос типа POST в формате JSON | JsonResponse – HTTP ответ в формате JSON | Пытается авторизовать и сменить пароль пользователя по данным, пришедшим из request. |
| set\_favourite\_manga | Request – HTTP запрос типа POST в формате JSON | JsonResponse – HTTP ответ в формате JSON | Пытается авторизовать пользователя по данным, пришедшим из request, после чего задать список любимой манги. |
| get\_favourite\_manga | Request – HTTP запрос типа POST в формате JSON | JsonResponse – HTTP ответ в формате JSON | Пытается авторизовать пользователя по данным, пришедшим из request, после чего получить список любимой манги. |
| username\_present | username, email | Bool – логическая переменная | Проверяет, существует ли пользователь с таким никнеймом или email адресом в базе данных |
| validate\_post\_mail | email | Bool – логическая переменная | Проверяет, корректно ли введена почта |
| get\_manga | Request – HTTP запрос типа POST в формате JSON | JsonResponse – HTTP ответ в формате JSON | Обертка над методом get\_manga\_logic. Получает кодовое название манги, после чего передает его в вышеназванную функцию, а ответ от нее возвращает в качестве JsonResponse |
| get\_manga\_logic | code – кодовое название манги | Dict – словарь, содержащий информацию о манге | Скачивает и выдает информацию по запрошенной манге (Обложка, описание, жанры, количество томов, глав, названия глав, рейтинг) |
| get\_manga\_page | Request – HTTP запрос типа POST в формате JSON | JsonResponse – HTTP ответ в формате JSON | Выдает картинку, запрошенную в request, в формате base64 вместе с общим количеством страниц в главе. |
| get\_manga\_page\_urls | Request – HTTP запрос типа POST в формате JSON | JsonResponse – HTTP ответ в формате JSON | Выдает список ссылок на все картинки главы вместе с общим количеством страниц в главе. |
| get\_chapter | code, volume, chapter – кодовое название манги, номер тома и главы | - | Скачивает все страницы главы запрошенной манги |
| get\_ongoings | Request – HTTP запрос типа POST в формате JSON | JsonResponse – HTTP ответ в формате JSON | Возвращает список популярной манги |
| get\_new\_manga | Request – HTTP запрос типа POST в формате JSON | JsonResponse – HTTP ответ в формате JSON | Возвращает список новой манги |
| create\_folder | Path – путь до места, где необходимо создать папку | - | Создает папку по указанному пути |
| download\_image | url, path – ссылка на фотографию и путь до места, где необходимо сохранить фотографию | - | Запрашивает фотографию по ссылке и скачивает ее |
| download\_cover | url, path – ссылка на фотографию и путь до места, где необходимо сохранить фотографию | - | Имитируя браузер, переходит по ссылке на фотографию, после чего сохраняет ее на компьютере |
| concat\_url | code, volume, chapter – кодовое название манги, номер тома и главы | Path – путь до файла, где лежит глава | Возвращает путь по которой лежит глава, конкатенируя его из кодового названия манги, номера тома и главы |
| get\_image\_bytes | Path - путь до файла, где лежит фотография | String – фотография в base64 | Кодирует фотографию, находящуюся по указанному пути в base64 |
| get\_driver\_with\_random\_ua | - | Driver – драйвер браузера Google Chrome | Имитирует браузер Google Chrome с рандомным параметром user\_agent |
| worker\_downloader | i, code, volume, chapter, urls – счетчик, кодовое название манги, номер тома, главы и ссылка на результирующий массив | - | Функция, запускающаяся параллельно в нескольких потоках. Скачивает фотографии и добавляет в результирующий список ссылку на данную картинку. |
| get\_image\_path | code, volume, chapter, page – кодовое название манги, номер тома, главы и страницы | Path – путь до картинки | Конкатенирует из кодового названия, номера тома, главы и страницы путь до картинки |
| is\_image\_downloaded | code, volume, chapter, page – кодовое название манги, номер тома, главы и страницы | Bool – логическая переменная | Проверяет, существует ли фотография по данному пути (путь собирается с помощью функции get\_image\_path) |
| parse\_chapter\_info | String – строка, полученная из тега html документа | Dict | Парсит информацию о главе и возвращает словарь содержащий номер тома, номер главы и название главы |
| Crop\_url | url | String | Получает кодовое название манги из ссылки на мангу |

# ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Изм. | Номера листов (страниц) | | | | Всего листов (страниц) в документе | № документа | Входящий № сопроводительного документа и дата | Подпись | Дата |
| измененных | замененных | новых | аннулированных |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |