**Документация API «MyHealth»**

Эта документация описывает API (Application Programming Interface) для мобильного приложения «MyHealth».

**Общие сведения**

* **Формат запросов и ответов:** JSON
* **Кодировка:** UTF-8
* **Базовый URL:** (Будет изменен на фактический URL сервера) https://api.myhealth.com/v1
* **Аутентификация:** Большинство endpoint’ов требуют аутентификации пользователя. Аутентификация осуществляется с использованием токенов Bearer, которые необходимо передавать в заголовке Authorization.

**Структура документации**

Для каждого endpoint’а будет указано:

* **URL:** URL endpoint’а.
* **Метод:** HTTP метод (GET, POST, PUT, DELETE).
* **Описание:** Описание endpoint’а.
* **Аутентификация:** Требуется или нет.
* **Параметры запроса:** (Если есть) Описание параметров, которые необходимо передать в запросе (в теле запроса или в URL).
* **Пример запроса:** Пример запроса в формате JSON.
* **Параметры ответа:** Описание полей, которые возвращаются в ответе.
* **Пример ответа (Успех):** Пример успешного ответа в формате JSON (HTTP status code 200 или 201).
* **Возможные ошибки:** Описание возможных ошибок и соответствующие HTTP status code (например, 400, 401, 404, 500).
* **Пример ответа (Ошибка):** Пример ответа в случае ошибки в формате JSON.

**Endpoint’ы**

**1. Аутентификация**

* **1.1. Регистрация пользователя**
  + **URL:** /auth/register
  + **Метод:** POST
  + **Описание:** Регистрация нового пользователя.
  + **Аутентификация:** Нет
  + **Параметры запроса:**
    - login (string, required): Логин пользователя.
    - email (string, required): Адрес электронной почты пользователя.
    - telephone (string, required): Номер телефона пользователя.
    - password (string, required): Пароль пользователя.
    - fio (string, required): Полное имя пользователя.
  + **Пример запроса:**

json

{

«login»: «user123»,

«email»: «user@example.com»,

«telephone»: «+79001234567»,

«password»: «password123»,

«fio»: «Иванов Иван Иванович»

}

* + **Параметры ответа:**
    - success (boolean): true в случае успеха, false в случае ошибки.
    - message (string): Сообщение об успехе или ошибке.
    - user\_id (integer, optional): ID нового пользователя.
  + **Пример ответа (Успех):**

json

{

«success»: **true**,

«message»: «Регистрация прошла успешно»,

«user\_id»: 123

}

* + **Возможные ошибки:**
    - 400 Bad Request: Ошибка валидации данных.
    - 409 Conflict: Логин или email уже зарегистрированы.
  + **Пример ответа (Ошибка - 400 Bad Request):**

json

{

«success»: **false**,

«message»: «Ошибка валидации: Неверный формат email»

}

* **1.2. Авторизация пользователя**
  + **URL:** /auth/login
  + **Метод:** POST
  + **Описание:** Авторизация существующего пользователя.
  + **Аутентификация:** Нет
  + **Параметры запроса:**
    - login (string, required): Логин или email пользователя.
    - password (string, required): Пароль пользователя.
  + **Пример запроса:**

json

{

«login»: «user123»,

«password»: «password123»

}

* + **Параметры ответа:**
    - success (boolean): true в случае успеха, false в случае ошибки.
    - message (string): Сообщение об успехе или ошибке.
    - token (string, optional): Токен Bearer для доступа к защищенным endpoint’ам.
    - user\_id (integer, optional): ID пользователя.
  + **Пример ответа (Успех):**

json

{

«success»: **true**,

«message»: «Авторизация прошла успешно»,

«token»: «eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...»,

«user\_id»: 123

}

* + **Возможные ошибки:**
    - 400 Bad Request: Ошибка валидации данных.
    - 401 Unauthorized: Неверный логин или пароль.
  + **Пример ответа (Ошибка - 401 Unauthorized):**

json

{

«success»: **false**,

«message»: «Неверный логин или пароль»

}

**2. Лекарства (Preparats)**

* **2.1. Получение списка лекарств**
  + **URL:** /preparats
  + **Метод:** GET
  + **Описание:** Получение списка лекарств с возможностью фильтрации и пагинации.
  + **Аутентификация:** Да (требуется токен Bearer)
  + **Параметры запроса:**
    - category\_id (integer, optional): ID категории для фильтрации.
    - search (string, optional): Поисковый запрос по названию лекарства.
    - page (integer, optional, default: 1): Номер страницы.
    - limit (integer, optional, default: 20): Количество элементов на странице.
  + **Пример запроса:**
    - GET /preparats?category\_id=1&search=Аспирин&page=2&limit=10
  + **Параметры ответа:**
    - success (boolean): true в случае успеха, false в случае ошибки.
    - data (array): Массив объектов лекарств.
    - total (integer): Общее количество лекарств, соответствующих запросу.
  + **Пример ответа (Успех):**

json

{

«success»: **true**,

«data»: [

{

«preparat\_id»: 1,

«category\_id»: 1,

«название»: «Аспирин»,

«описание»: «Описание Аспирина...»,

«производитель»: «Bayer»,

«цена»: 100.00,

«срок годности»: «2024-12-31»,

«форма выпуска»: «Таблетки»,

«изображение»: «https://example.com/aspirin.jpg»,

«количество на складе»: 100

},

{...}

],

«total»: 50

}

* + **Возможные ошибки:**
    - 401 Unauthorized: Токен недействителен или отсутствует.
    - 500 Internal Server Error: Ошибка на сервере.
* **2.2. Получение информации о конкретном лекарстве**
  + **URL:** /preparats/{preparat\_id} (где {preparat\_id} - ID лекарства)
  + **Метод:** GET
  + **Описание:** Получение информации о конкретном лекарстве по его ID.
  + **Аутентификация:** Да (требуется токен Bearer)
  + **Параметры ответа:**
    - success (boolean): true в случае успеха, false в случае ошибки.
    - data (object): Объект лекарства.
  + **Пример ответа (Успех):**

json

{

«success»: **true**,

«data»: {

«preparat\_id»: 1,

«category\_id»: 1,

«название»: «Аспирин»,

«описание»: «Описание Аспирина...»,

«производитель»: «Bayer»,

«цена»: 100.00,

«срок годности»: «2024-12-31»,

«форма выпуска»: «Таблетки»,

«изображение»: «https://example.com/aspirin.jpg»,

«количество на складе»: 100

}

}

* + **Возможные ошибки:**
    - 401 Unauthorized: Токен недействителен или отсутствует.
    - 404 Not Found: Лекарство с указанным ID не найдено.
    - 500 Internal Server Error: Ошибка на сервере.

**3. Корзина (Cart)**

* **3.1. Получение содержимого корзины**
  + **URL:** /cart
  + **Метод:** GET
  + **Описание:** Получение содержимого корзины текущего пользователя.
  + **Аутентификация:** Да (требуется токен Bearer)
  + **Параметры ответа:**
    - success (boolean): true в случае успеха, false в случае ошибки.
    - data (array): Массив объектов лекарств в корзине (с указанием количества).
    - total\_price (number): Общая стоимость товаров в корзине.
  + **Пример ответа (Успех):**

json

{

«success»: **true**,

«data»: [

{

«preparat\_id»: 1,

«название»: «Аспирин»,

«цена»: 100.00,

«количество»: 2

},

{

«preparat\_id»: 2,

«название»: «Парацетамол»,

«цена»: 50.00,

«количество»: 1

}

],

«total\_price»: 250.00

}

* + **Возможные ошибки:**
    - 401 Unauthorized: Токен недействителен или отсутствует.
    - 500 Internal Server Error: Ошибка на сервере.
* **3.2. Добавление лекарства в корзину**
  + **URL:** /cart/add
  + **Метод:** POST
  + **Описание:** Добавление лекарства в корзину текущего пользователя.
  + **Аутентификация:** Да (требуется токен Bearer)
  + **Параметры запроса:**
    - preparat\_id (integer, required): ID лекарства, которое необходимо добавить в корзину.
    - количество (integer, required, default: 1): Количество лекарства.
  + **Пример запроса:**

json

{

«preparat\_id»: 1,

«количество»: 2

}

* **3.3. Удаление лекарства из корзины**
  + **URL:** /cart/remove/{preparat\_id} (где {preparat\_id} - ID лекарства)
  + **Метод:** DELETE

**4. Заказы (Orders)**

* **4.1. Создание заказа**
  + **URL:** /orders
  + **Метод:** POST
  + **Описание:** Создание нового заказа на основе содержимого корзины.
  + **Аутентификация:** Да (требуется токен Bearer)
  + **Параметры запроса:**
    - способ\_оплаты (string, required): Способ оплаты (“наличные”, “карта онлайн”).
    - способ\_доставки (string, required): Способ доставки (“самовывоз”, “курьер”).
    - адрес\_доставки (string, optional, required if способ\_доставки == “курьер”): Адрес доставки.
  + **Пример ответа (Успех):**
* **4.2. Получение списка заказов пользователя**
  + **URL:** /orders
  + **Метод:** GET
* **4.3. Получение информации о заказе**
  + **URL:** /orders/{order\_id}
  + **Метод:** GET

**5. Бронирование (Reservations)**

* **5.1. Бронирование лекарства** \* **URL:** /reservations \* **Метод:** POST

**6. Push-Уведомления (Push notifications)**

* **6.1. Получение/Удаление токена устройства**
  + **URL:** /push/token
  + **Метод:** POST/DELETE
    - token (string, required): Токен устройства

**Документация API для Администратора/Фармацевта**

Для администратора/фармацевта добавьте endpoint’ы по управлению заказами, лекарствами, категориями и скидками. Например

* **7.1. Получение списка заказов (Admin/Pharmacist)**
  + **URL:** /admin/orders или /pharmacist/orders
  + **Метод:** GET
  + **Аутентификация:** Да (требуется токен Bearer с соответствующей ролью).
* **7.2. Изменение статуса заказа (Admin/Pharmacist)**
  + **URL:** /admin/orders/{order\_id}/status или /pharmacist/orders/{order\_id}/status
  + **Метод:** PUT
  + **Описание:** Изменение статуса заказа.
  + **Параметры запроса:**
    - status (string, required): Новый статус заказа (“в обработке”, “готов к выдаче”, “доставлен”, “отменен”).
  + **Возможные ошибки:**
    - 403 Forbidden: Нет прав для изменения статуса заказа.

**Обслуживание API и планы на будущее**