

# Система управления подписками

## Оглавление

1. Общее описание системы
2. Модели данных
3. Алгоритмы работы с подписками
4. API интерфейс
5. Варианты использования (Use Cases)
6. Процесс активации и отмены подписок
7. Интеграция с другими системами

## Общее описание системы

Система управления подписками позволяет пользователям запрашивать, активировать и управлять подписками на различные категории контента по определенным локациям. Система поддерживает различные типы тарифов, включая демо-тариф для ознакомления с сервисом.

Ключевые возможности: - Создание подписок на выбранные категории по конкретным локациям - Поддержка различных тарифов с разной продолжительностью и стоимостью - Автоматическая активация демо-подписок без подтверждения администратором - Управление переходом с демо-версии на платные подписки - Отслеживание истории подписок - Автоматическая проверка существования и статусов уже имеющихся подписок - Отмена подписок администратором

## Модели данных

### Основные модели данных

1. **User**
  - Пользователь системы, имеющий подписки
  - Отношения: имеет множество подписок (UserSubscription)
  - Ключевые поля: id, name, is\_trial\_used, role
2. **UserSubscription**
  - Подписка пользователя
  - Отношения: принадлежит User, Tariff, Category, Location
  - Ключевые поля: id, user\_id, tariff\_id, category\_id, location\_id, price\_paid, start\_date, end\_date, status, payment\_method
  - Возможные статусы: "pending", "active", "expired", "cancelled"
  - Дополнительное поле: is\_enabled (boolean) - позволяет пользователю временно отключать активную подписку
3. **Tariff**
  - Тарифный план с определенной длительностью и стоимостью
  - Отношения: имеет многие UserSubscription
  - Ключевые поля: id, name, code, duration\_hours, price, is\_active

- Типы тарифов: “demo”, “premium” и другие
- 4. **Category**
  - Категория контента, на которую оформляется подписка
  - Отношения: имеет многие UserSubscription
  - Ключевые поля: id, name
- 5. **Location**
  - Географическая локация, для которой действует подписка
  - Отношения: имеет многие UserSubscription
  - Ключевые поля: id, city, region
- 6. **SubscriptionHistory**
  - История изменений подписок для аудита
  - Отношения: принадлежит User, UserSubscription
  - Ключевые поля: id, user\_id, subscription\_id, action, tariff\_name, category\_name, location\_name, price\_paid, action\_date, notes

## Алгоритмы работы с подписками

### Запрос подписки

Алгоритм запроса подписки: 1. **Валидация входных данных:** - Проверка наличия и корректности tariff\_id, category\_ids (массив) и location\_id - Конвертация всех ID в целочисленный формат

2. **Проверка существования данных:**
  - Проверка существования тарифа по tariff\_id
  - Проверка существования локации по location\_id
  - Проверка существования всех категорий из массива category\_ids
3. **Проверка специальных условий для демо-тарифа:**
  - Если это демо-тариф:
    - Проверка, что пользователь еще не использовал демо (is\_trial\_used == false)
    - Проверка, что выбрана только одна категория (ограничение для демо)
4. **Создание подписок:**
  - Итерация по всем категориям из массива category\_ids
  - Для каждой категории:
    1. **Проверка особого случая** - платный тариф и существующая демо-подписка:
      - Если это платный тариф и есть активная демо-подписка на ту же категорию и локацию
      - Отмена существующей демо-подписки с пометкой “Автоматическая отмена при переходе на платный тариф”
    2. **Проверка наличия активной подписки:**
      - Если уже есть активная подписка на ту же категорию и локацию (кроме случая апгрейда с демо)
      - Пропуск создания новой подписки для этой категории
    3. **Проверка наличия ожидающей подписки:**
      - Если уже есть подписка со статусом “pending” на ту же категорию и локацию
      - Пропуск создания новой подписки для этой категории
    4. **Создание подписки** через метод user->requestSubscription()
5. **Финализация процесса:**

- Проверка, что был создан хотя бы одна подписка
- Если это демо-тариф, установка флага `is_trial_used = true` для пользователя
- Возврат информации о созданных подписках

## Создание подписки (внутренний процесс)

Внутренний алгоритм `User::requestSubscription()`: 1. Поиск тарифа и проверка его доступности 2. Для демо-тарифа - дополнительные проверки: - Проверка, что пользователь не использовал демо ранее - Проверка, что нет других ожидающих демо-подписок 3. Определение цены подписки (стандартная или специальная для локации) 4. Определение статуса и дат: - Для демо-тарифа: статус "active", даты `start_date = now()`, `end_date = now() + duration_hours` - Для платных тарифов: статус "pending", даты null (будут установлены при активации) 5. Создание записи в `UserSubscription` 6. Создание записи в `SubscriptionHistory` с соответствующим действием 7. Для демо-тарифа - установка флага использования демо

## Алгоритм отмены подписки

1. Проверка существования подписки и принадлежности пользователю
2. Установка статуса "cancelled" в записи подписки
3. Создание записи в истории подписок с действием "cancelled"
4. Возврат результата операции

## Алгоритм активации подписки администратором

1. Проверка, что подписка имеет статус "pending" или "expired"
2. Установка статуса "active"
3. Установка дат начала (текущая дата) и окончания (текущая дата + длительность тарифа)
4. Сохранение информации о методе оплаты, примечаний, администраторе
5. Создание записи в истории подписок с действием "activated"

# API интерфейс

## Создание заявки на подписку

Запрос:

```
POST /subscriptions/request
{
  "tariff_id": 1,
  "category_ids": [2, 3, 5],
  "location_id": 4
}
```

Ответ (успех):

```
{
  "code": 200,
  "status": "success",
  "message": "Заявки на подписку успешно созданы",
}
```

```

    "data": {
      "subscription_count": 3,
      "subscriptions": [
        {
          "id": 101,
          "status": "pending",
          "category_id": 2
        },
        {
          "id": 102,
          "status": "pending",
          "category_id": 3
        },
        {
          "id": 103,
          "status": "pending",
          "category_id": 5
        }
      ]
    }
  }
}

```

#### Ответ (демо-тариф):

```

{
  "code": 200,
  "status": "success",
  "message": "Заявки на подписку успешно созданы",
  "data": {
    "subscription_count": 1,
    "subscriptions": [
      {
        "id": 104,
        "status": "active", // Сразу активна для демо
        "category_id": 3
      }
    ]
  }
}

```

#### Ответ (ошибка):

```

{
  "code": 400,
  "status": "error",
  "message": "Неверный формат запроса. Отсутствует или неверный
формат category_ids. Должен быть массив с идентификаторами
категорий",
  "error_code": "validation_error"
}

```

### Отмена подписки

#### Запрос:

```

DELETE /admin/subscriptions/{id}/cancel
{
  "reason": "Отменено по запросу пользователя"
}

```

#### Ответ (успех):

```
{
  "code": 200,
  "status": "success",
  "message": "Подписка успешно отменена"
}
```

**Примечание:** Этот эндпоинт доступен только пользователям с ролью администратора и требует соответствующего уровня авторизации.

## Получение активных подписок

### Запрос:

GET /subscriptions/active

### Ответ:

```
{
  "code": 200,
  "status": "success",
  "data": {
    "subscriptions": [
      {
        "id": 104,
        "tariff": {
          "id": 2,
          "name": "Демо",
          "code": "demo"
        },
        "category": {
          "id": 3,
          "name": "Спорт"
        },
        "location": {
          "id": 1,
          "name": "Москва, Центральный"
        },
        "price_paid": 0,
        "start_date": "2023-07-01 10:00:00",
        "end_date": "2023-07-08 10:00:00",
        "status": "active",
        "remaining_seconds": 345600
      }
    ]
  }
}
```

## Отключение/включение подписки (временное)

### Запрос:

```
PATCH /subscriptions/{id}/toggle
{
  "is_enabled": false
}
```

### Ответ (успех):

```
{
```

```
"code": 200,  
"status": "success",  
"message": "Статус подписки успешно изменен",  
"data": {  
  "subscription_id": 101,  
  "is_enabled": false  
}  
}
```

**Примечание:** Этот эндпоинт позволяет пользователю временно включать/отключать активную подписку без её отмены. Отключенная подписка не даёт доступа к контенту, но продолжает учитываться в системе и может быть включена в любой момент.

## Варианты использования

### 1. Создание демо-подписки

**Сценарий:** 1. Новый пользователь хочет попробовать сервис 2. Пользователь выбирает демо-тариф, одну категорию и локацию 3. Система создает активную демо-подписку без подтверждения администратором 4. Пользователь получает немедленный доступ к выбранной категории контента 5. Система отмечает, что пользователь использовал демо (is\_trial\_used = true)

**Ограничения:** - Только одна категория для демо-тарифа - Пользователь может использовать демо-тариф только один раз - Нельзя иметь более одной активной демо-подписки одновременно

### 2. Запрос платной подписки

**Сценарий:** 1. Пользователь выбирает платный тариф, несколько категорий и локацию 2. Система создает заявки на подписки со статусом "pending" 3. Администратор проверяет заявки через панель администратора 4. Администратор активирует подписки после подтверждения оплаты 5. Пользователь получает доступ к выбранным категориям контента

### 3. Переход с демо на платную подписку

**Сценарий:** 1. Пользователь имеет активную демо-подписку на категорию X и локацию Y 2. Пользователь запрашивает платную подписку на ту же категорию X и локацию Y 3. Система автоматически отменяет существующую демо-подписку 4. Система создает новую заявку на платную подписку со статусом "pending" 5. Администратор подтверждает подписку после проверки оплаты 6. Пользователь продолжает иметь доступ к категории X, но уже по платной подписке

### 4. Продление существующей подписки

**Сценарий:** 1. У пользователя есть активная или истекающая подписка 2. Администратор выбирает функцию продления подписки 3. Администратор указывает детали продления (сумма, длительность) 4. Система продлевает дату окончания подписки 5. Система создает запись в истории с действием "extended"

## 5. Отмена подписки

**Сценарий:** 1. Администратор выбирает подписку пользователя через админ-панель 2. Администратор нажимает на кнопку отмены и указывает причину 3. Система меняет статус подписки на “cancelled” 4. Система создает запись в истории с действием “cancelled” 5. Доступ к контенту по этой подписке прекращается

**Примечание:** - Только администратор имеет право отменять подписки - Пользователь не может самостоятельно отменить подписку через интерфейс

## 6. Временное отключение подписки пользователем

**Сценарий:** 1. Пользователь имеет активную подписку, которую хочет временно отключить 2. Пользователь выбирает подписку в интерфейсе и нажимает кнопку отключения 3. Система устанавливает флаг `is_enabled = false` для этой подписки 4. Доступ к контенту по этой подписке временно прекращается 5. В любой момент пользователь может снова включить подписку, установив `is_enabled = true`

**Примечание:** - Временное отключение не влияет на статус подписки или срок её действия - Даже при отключении срок действия подписки продолжает истекать - После включения подписка сразу становится активной, если не истёк срок её действия

# Процесс активации и отмены подписок

## Жизненный цикл подписки

### 1. Создание:

- Демо-подписка: сразу получает статус “active” с датами начала и окончания
- Платная подписка: получает статус “pending” без указания дат

### 2. Активация (для платных подписок):

- Администратор подтверждает оплату
- Статус меняется на “active”
- Устанавливаются даты начала и окончания

### 3. Истечение срока:

- Система периодически проверяет даты окончания
- Когда текущая дата > даты окончания, статус меняется на “expired”
- Создается запись в истории с действием “expired”

### 4. Отмена:

- Только администратор может инициировать отмену подписки
- Статус меняется на “cancelled”
- Создается запись в истории с действием “cancelled”
- Пользователь должен связаться с администратором для отмены подписки

### 5. Продление:

- Администратор инициирует продление
- Дата окончания увеличивается на указанную длительность

- Если подписка была “expired”, статус меняется на “active”
  - Создается запись в истории с действием “extended”
6. **Временное отключение/включение:**
- Пользователь может временно отключить активную подписку
  - Статус подписки остаётся “active”, но устанавливается флаг `is_enabled = false`
  - Доступ к контенту временно приостанавливается
  - Пользователь может в любой момент включить подписку, установив `is_enabled = true`
  - Временное отключение не продлевает срок действия подписки

## **Автоматические процессы**

1. **Проверка истечения срока подписок:**
  - Системная задача, выполняемая по расписанию
  - Проверяет все активные подписки с `date_end <` текущая дата
  - Меняет их статус на “expired”
2. **Автоматическая отмена демо при переходе на платную:**
  - При запросе платной подписки проверяется наличие демо на ту же категорию/локацию
  - Демо-подписка автоматически отменяется

## **Интеграция с другими системами**

### **Интеграция с платежными системами**

Текущая реализация не включает прямую интеграцию с платежными системами. Процесс оплаты происходит вне системы, и подтверждение выполняется администратором.

При необходимости, можно расширить систему для интеграции с:

- Платежными шлюзами
- Системами выставления счетов
- Автоматической обработкой оплаты и активации подписок

### **Интеграция с системой уведомлений**

Рекомендуется интегрировать систему подписок с системой уведомлений для:

- Оповещения пользователей о создании и статусе подписок
- Напоминаний об истечении срока действия
- Уведомлений администраторов о новых заявках на подписку

### **Интеграция с системой контента**

Система подписок должна быть интегрирована с системой доступа к контенту, чтобы:

- Предоставлять доступ к контенту только пользователям с активными подписками
- Фильтровать контент по категориям согласно подпискам
- Ограничивать контент по географическим локациям согласно подпискам