ООО «Мелвад»
390026, г. Рязань, ул. Ленинского Комсомола, д. 106, кв. 18
ИНН 6234193052, КПП 623401001
ОГРН 1206200011907
тел. 8 (920) 961-27-77
support@melvad.com

Руководство для проведения проверки эксплуатации Melvad DMP

Оглавление

1. Руководство для работы....4
1.1. Авторизация....4
1.2. Контакты техподдержки....4
2. Общее описание....5
2.1. Назначение DMP....5
2.2. Дашборд....5
2.3. Пользователи и роли....6
2.4. SSP....6
2.5. DSP....7
2.6. DMP....8
2.7. Сегменты....8
2.8. Синхронизация кук....9
2.9. Мониторинг....10
3. Руководство для проведения проверки эксплуатации ПО....11
3.1. Авторизация....11
3.2. Создание основных сущностей....11
3.2.1. Создание рекламодателя....11
3.2.2. Создание SSP....12
3.2.3. Создание DSP....13
3.2.4. Создание пиксельного сегмента....14
3.3. Проверка синхронизации кук и сегментов....15
3.3.1. Входящий матчинг SSP....15
3.3.2. Пиксель сегмента....15
3.3.3. Возврат от DSP....16
3.3.4. Сегмент по идентификаторам SSP....16
3.3.5. Получение сегментов пользователя (запрос партнёра)....16
3.3.6. Идентификатор пользователя у DSP (запрос партнёра)....16
3.3.7. Исходящий матчинг....17
3.3.8. Проверка на дашборде....17
4. Описание развернутого экземпляра инфраструктуры ПО....18
4.1. Общее описание....18
4.2. Подробное описание контейнеров сервисов, их процессов и компонентов....19
4.2.1. UI....19
4.2.2. API....19
4.2.3. Matching....20
4.2.4. PostgreSQL....20
4.2.5. ScyllaDB....20

4.2.6. Мониторинг.....21
4.3. Настройка удаленного доступа к развернутому экземпляру ПО 21
4.4. Расположение файлов приложения 22
4.4.1. Список volumes.....23
4.4.2. Смонтированные каталоги хоста 23
Приложение 1. Общая схема контейнеров и движения данных между ними.....24

1. Руководство для работы

1.1. Авторизация

Для ознакомления с работой DMP необходимо авторизоваться в личном кабинете, для этого перейдите по ссылке **https://ui.dmp.melvad.com** и:

- Укажите электронный адрес: **review@dmp.melvad.com**
- Введите пароль: **YckmffWrsDK4ssgV**

Учётная запись имеет роль «Администратор» и доступ ко всем разделам. Сессия действует 24 часа, после чего вход нужно повторить. После 5 неудачных попыток входа за 15 минут вход для этого адреса временно блокируется.

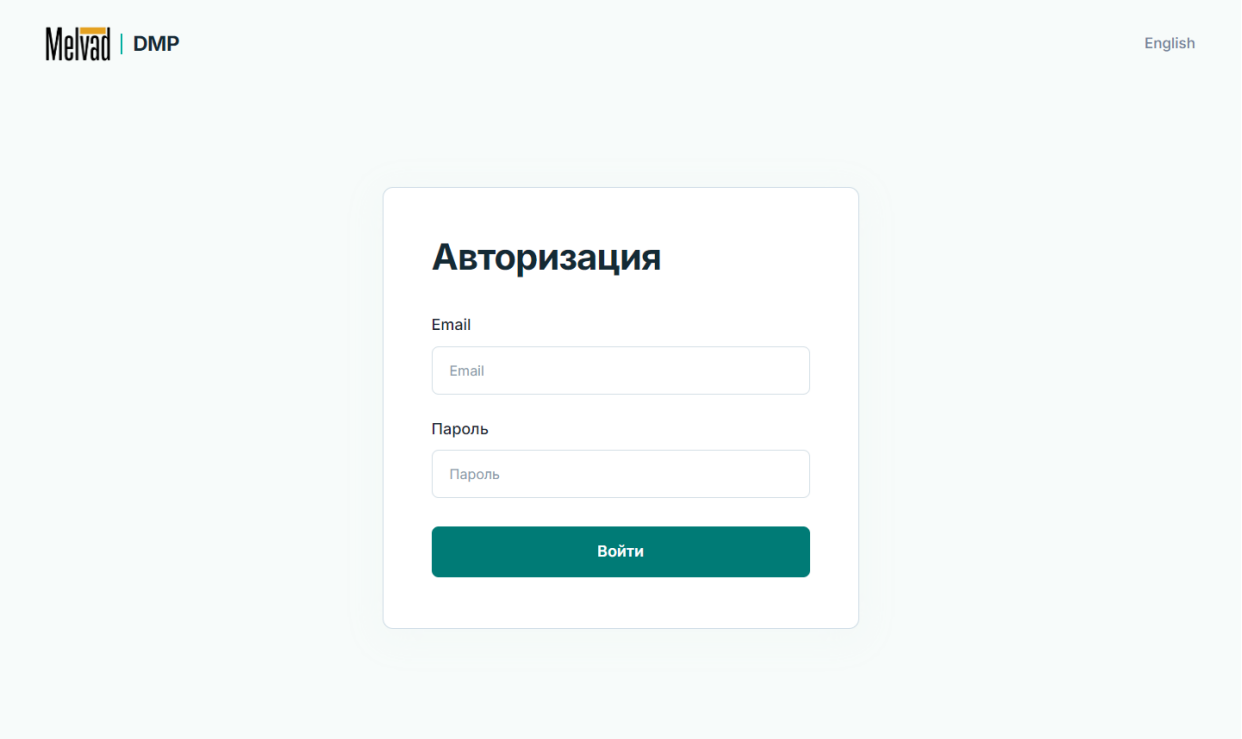


Рисунок 1 — Экран входа

1.2. Контакты техподдержки

Стуколкин Александр

e-mail: a.stukolkin@melvad.com

Телефон: +7(920) 961-27-77

2. Общее описание

2.1. Назначение DMP

DMP (Data Management Platform) — платформа управления данными об аудитории. Она решает две задачи:

- **синхронизация кук (матчинг)** — связывает идентификаторы пользователя у партнёров (SSP, DSP, DMP) с собственным идентификатором DMP (`oui`), чтобы партнёры могли обмениваться данными об одном и том же пользователе;
- **сегменты аудитории** — списки пользователей, которые рекламодатель собирает пикселем на своём сайте, загружает файлом (идентификаторы SSP, мобильные идентификаторы IDFA/GAID) или получает от сторонних DMP. Партнёры запрашивают сегменты пользователя и используют их для таргетинга рекламы.

ПО состоит из трёх частей: веб-интерфейс (UI), API для управления данными и сервис матчинга (matching), который принимает запросы браузеров пользователей и серверов партнёров.

2.2. Дашборд

Дашборд показывает запросы синхронизации кук, которые сервис матчинга получил от SSP и DSP:

- Периоды: сегодня, вчера, последние 7 дней, последние 30 дней или произвольный;
- График запросов SSP и DSP: до 2 дней — по часам, дольше — по дням;
- Итоги за период по SSP и DSP;
- Для администратора — диаграмма синхронизации кук по каждому партнёру (сколько запросов записали новую или продлённую куку).

Данные хранятся 90 дней.

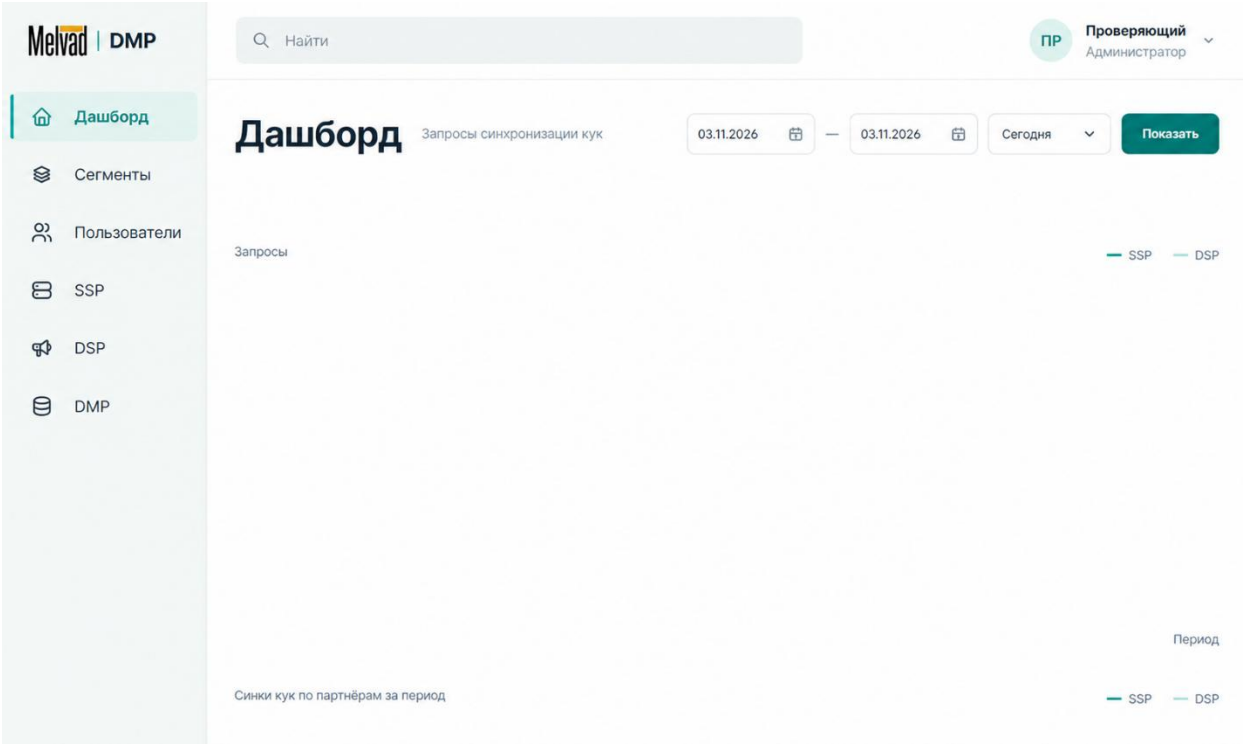


Рисунок 2 — Дашборд

2.3. Пользователи и роли

В DMP две роли — администратор и рекламодатель.

Администратор управляет всеми сущностями: пользователями, справочниками SSP, DSP и DMP, всеми сегментами, может создавать общие сегменты с ценой (CPM) и запускать импорт сегментов DMP.

Рекламодатель — пользователь, который создаёт и ведёт свои сегменты аудитории. Ему также видны (только для чтения) общие сегменты и сегменты сторонних DMP.

Пользователи входят в UI по email и паролю. Внешние сервисы работают с API по API-ключу пользователя (заголовок `X-Auth-Token`). Список пользователей можно посмотреть на вкладке «Пользователи»: вкладки Все / Активные / Неактивные / Удалённые, число сегментов, дата последнего входа, роль и статус.

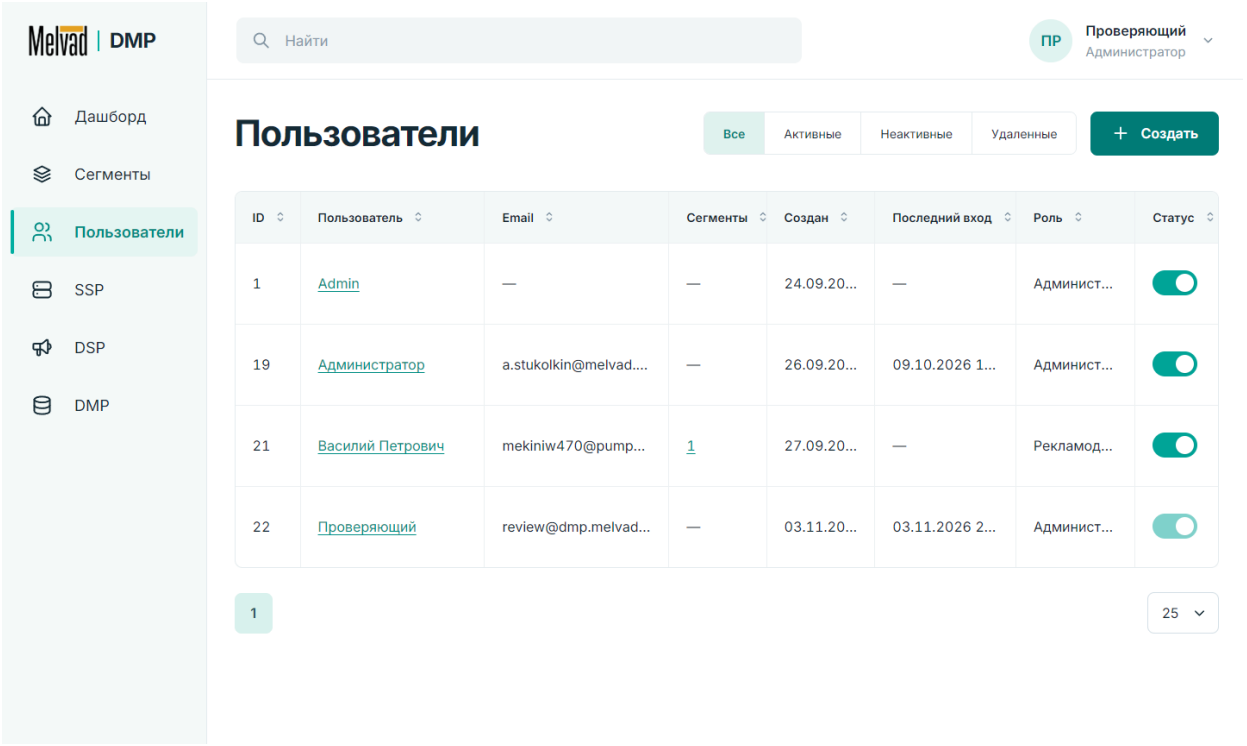


Рисунок 3 — Пользователи

2.4. SSP

SSP — партнёры (рекламные биржи), которые отправляют пользователей на синхронизацию кук в DMP. Для каждой SSP DMP выдаёт синк-ссылку вида:

```
https://match.dmp.melvad.com/match/<название SSP>?id={user_id}&rt={REDIRECT_TO_ENCODE_URL}
```

SSP подставляет вместо `{user_id}` идентификатор пользователя у себя, вместо `{REDIRECT_TO_ENCODE_URL}` — свою ссылку возврата, в которой на место идентификатора DMP ставит макрос `{{oui}}`. Синк-ссылка открывается кнопкой `</>` в колонке «Действия». Список SSP — на вкладке «SSP».

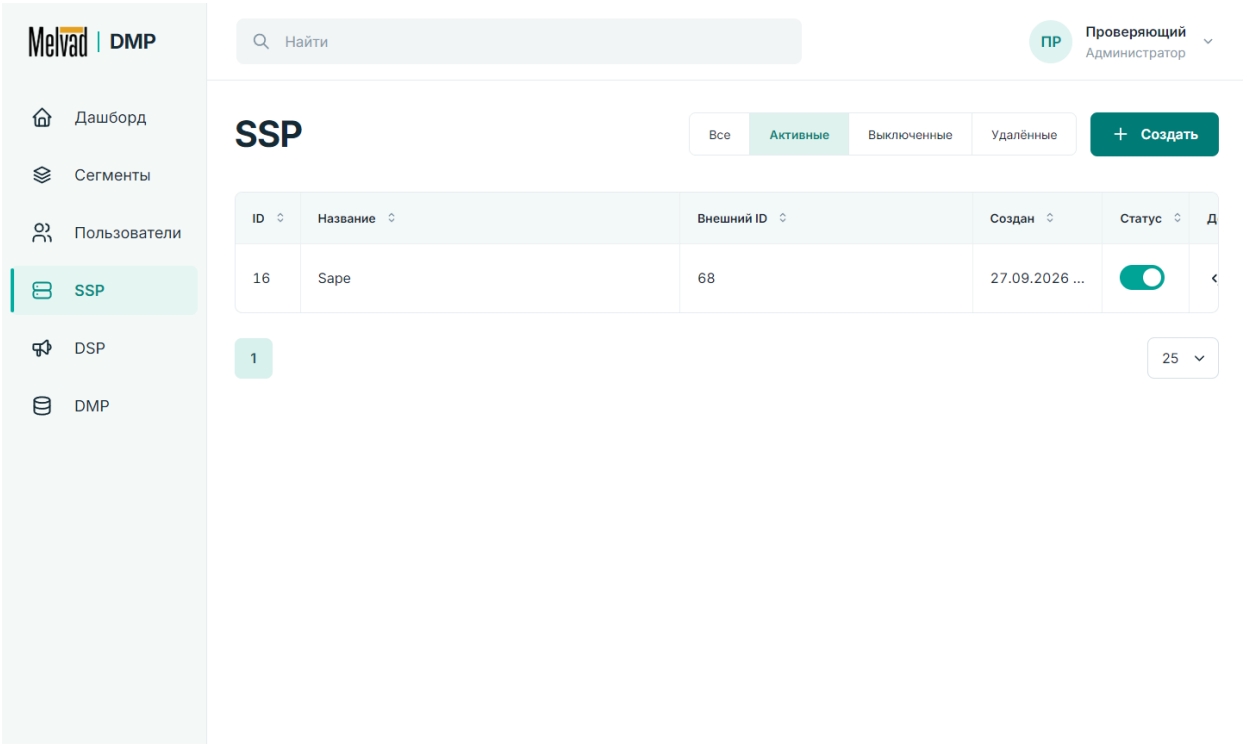


Рисунок 4 — SSP

2.5. DSP

DSP — партнёры, которые покупают рекламу и хотят знать пользователя DMP. Сайт, подключивший скрипт `/usermatch`, загружает синк-ссылки всех активных DSP (картинкой-пикселем или скрытым iframe), DSP возвращает пользователя на ссылку возврата со своим идентификатором. После этого сервер партнёра может получить идентификатор пользователя у DSP по методу `/match/ubu`. У DSP может быть несколько внешних ID (алиасов). Ссылка возврата открывается кнопкой `</>`. Список DSP — на вкладке «DSP».

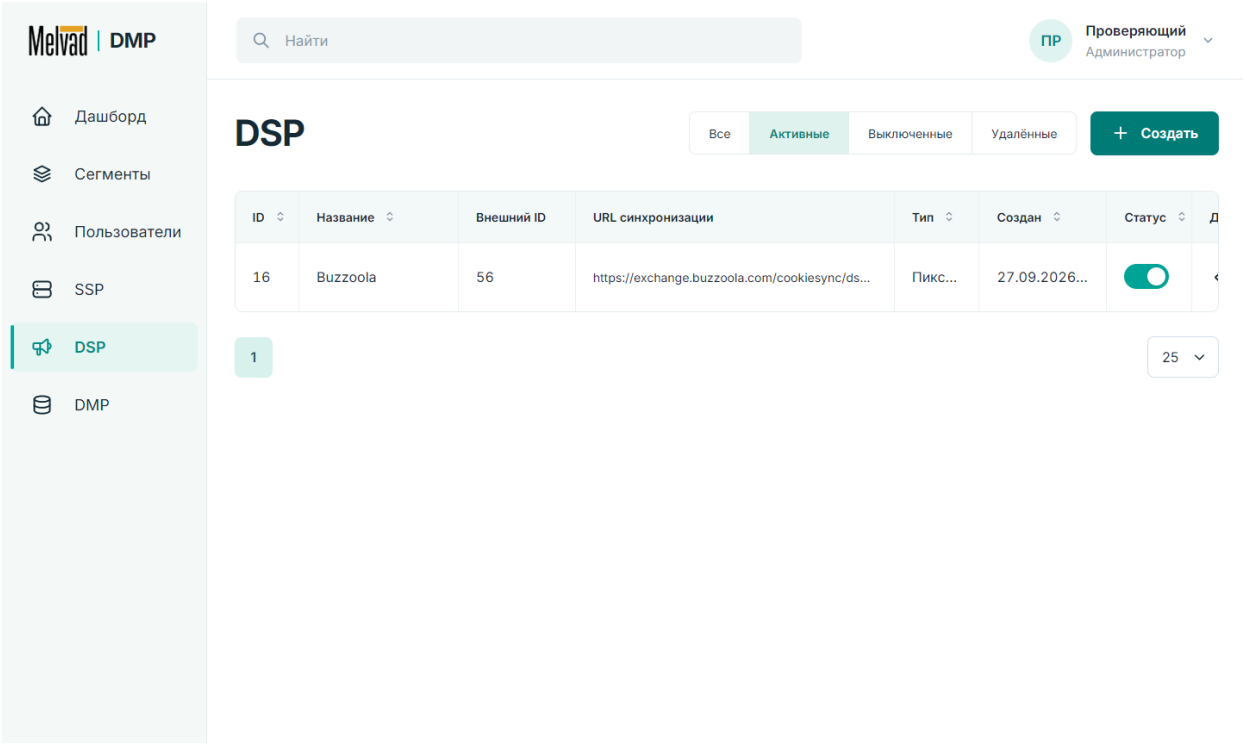


Рисунок 5 — DSP

2.6. DMP

DMP — сторонние платформы данных. После синхронизации с SSP пользователь перенаправляется на синк-ссылку случайной активной DMP с идентификатором DMP вместо макроса `{{oui}}`. Сегменты DMP (таксономия и списки пользователей) импортируются из папки DMP в S3-бакете автоматически по расписанию (каждый час) или вручную кнопкой запуска импорта; время последнего импорта и его ошибка показываются в списке. Список DMP — на вкладке «DMP».

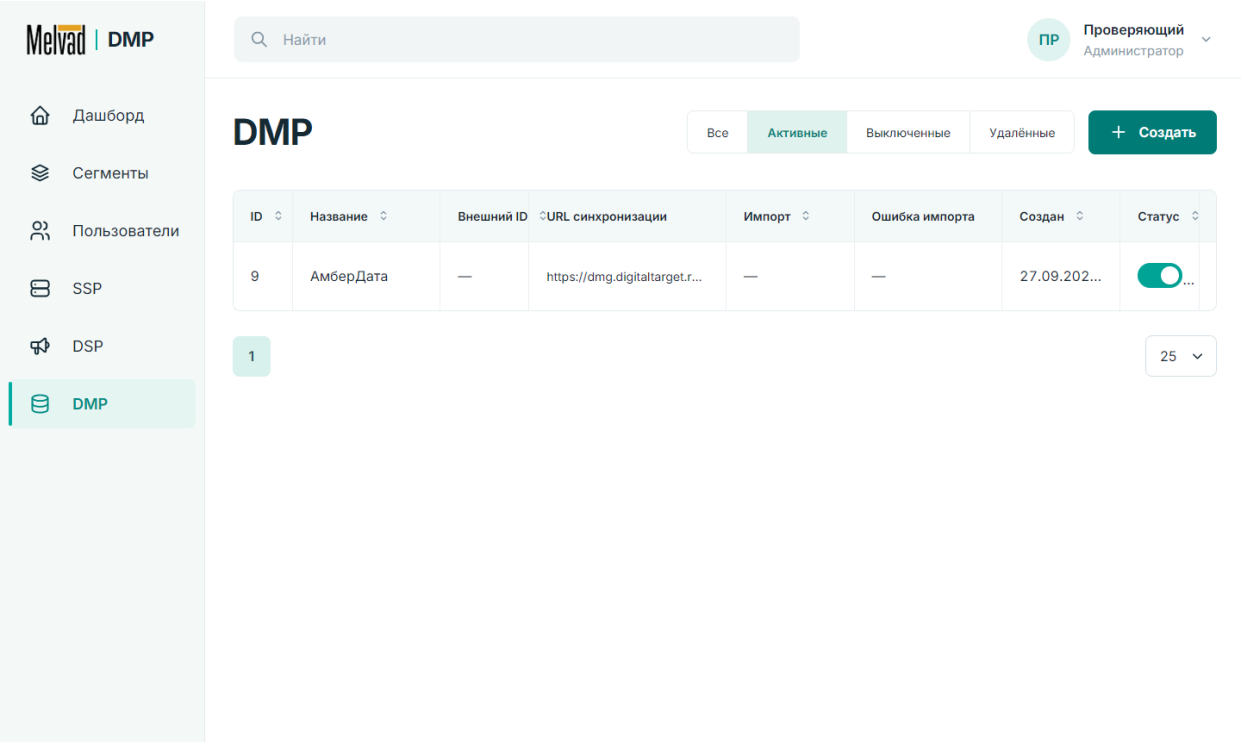


Рисунок 6 — DMP

2.7. Сегменты

Сегмент — список пользователей, объединённых общим признаком. Типы сегментов:

Источник	Как пользователи попадают в сегмент	Время жизни
Пиксель	пользователь загружает пиксель сегмента на сайте рекламодателя	30 дней с последней загрузки пикселя
SSP	файл CSV с идентификаторами пользователей у выбранной SSP; в сегмент попадают пользователи, уже синхронизированные с этой SSP	30 дней
Мобильные идентификаторы	файл CSV с IDFA/GAID или их MD5, от 1000 строк	задаётся, 1–365 дней
Сегменты DMP	импорт из S3 стороннего DMP, только чтение, с ценой CPM	7 дней с последнего импорта

Администратор может создать **общий сегмент** (владелец «Общий сегмент») с необязательной ценой CPM — его видят все рекламодатели. Статус сегмента меняется переключателем в колонке «Статус», код пикселя открывается кнопкой `</>`, колонка «Кол-

во записей» показывает число пользователей в сегменте (пересчитывается раз в час). Список сегментов — на вкладке «Сегменты».

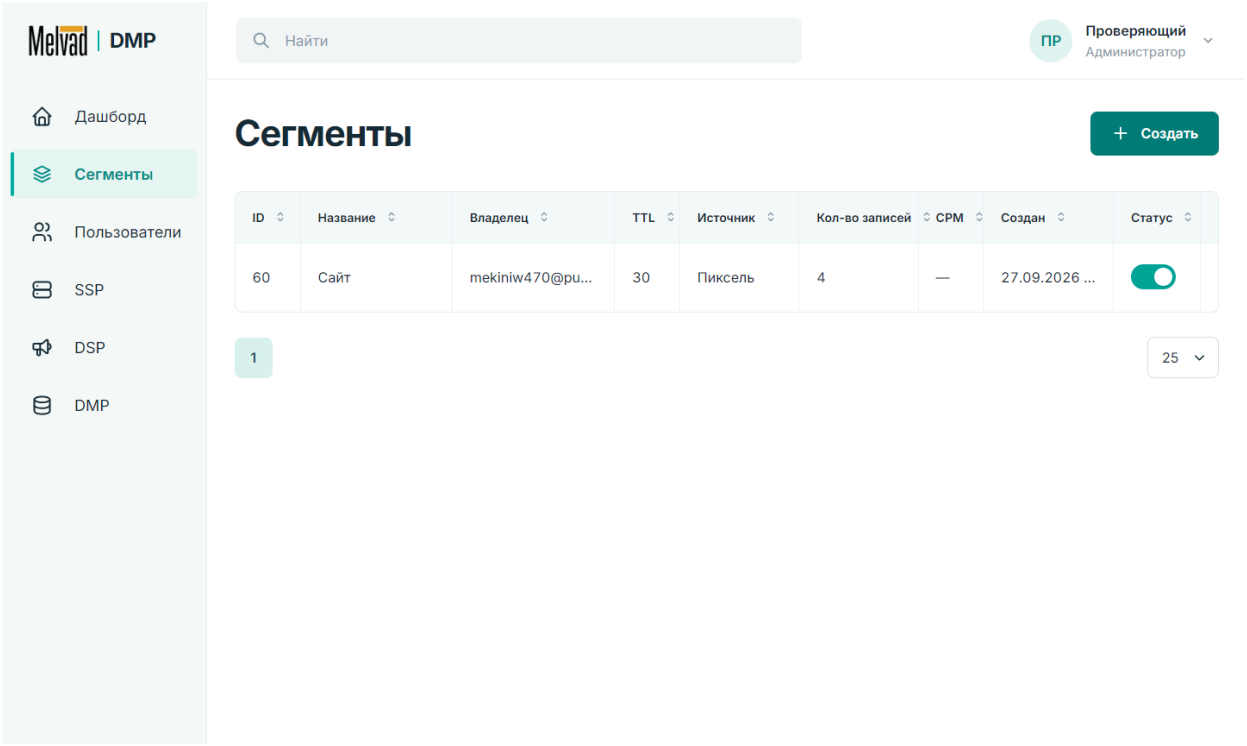


Рисунок 7 — Сегменты

2.8. Синхронизация кук

Куки (Cookie) — небольшой текстовый файл, который сайт сохраняет в браузере пользователя, чтобы запоминать его данные. DMP ставит пользователю куку `oui` со своим идентификатором на домене `match.dmp.melvad.com` на 30 дней.

Синхронизация кук позволяет связать идентификаторы партнёров (SSP, DSP, DMP) и DMP, чтобы партнёры могли показывать релевантную рекламу на основе сегментов пользователя. Пары «идентификатор партнёра — идентификатор DMP» хранятся 30 дней с последней синхронизации.

Метод	Кто вызывает	Назначение
`GET /match/<SSP>?id=&rt=`	браузер по синк-ссылке SSP	входящий матчинг: записывает пару, ставит куку `oui`, перенаправляет на `rt` или в DMP
`GET /usermatch`	скрипт на сайте	исходящий матчинг: загружает синк-ссылки активных DSP
`GET /match/<DSP>?id=`	браузер, возврат от DSP	записывает пару DSP
`GET /match/pixel?sadd=`	пиксель на сайте	добавляет пользователя в пиксельный сегмент
`GET /match/ubs?user_id=&user_type=`	сервер партнёра, только http	сегменты пользователя

Метод	Кто вызывает	Назначение
`POST /match/ubu`	сервер партнёра, только http	идентификатор пользователя у DSP

2.9. Мониторинг

Состояние сервисов показывает Grafana: **https://grafana.dmp.melvad.com** (пользователь **review**, пароль **VUx5hkYv9UiQdivL**, только просмотр). Домашний дашборд «DMP matching» — запросы по методам, коды ответов, задержки, итоги синхронизаций по партнёрам, запросы к базе Scylla. В папке DMP также есть дашборды сервера, процессов Go и PostgreSQL.

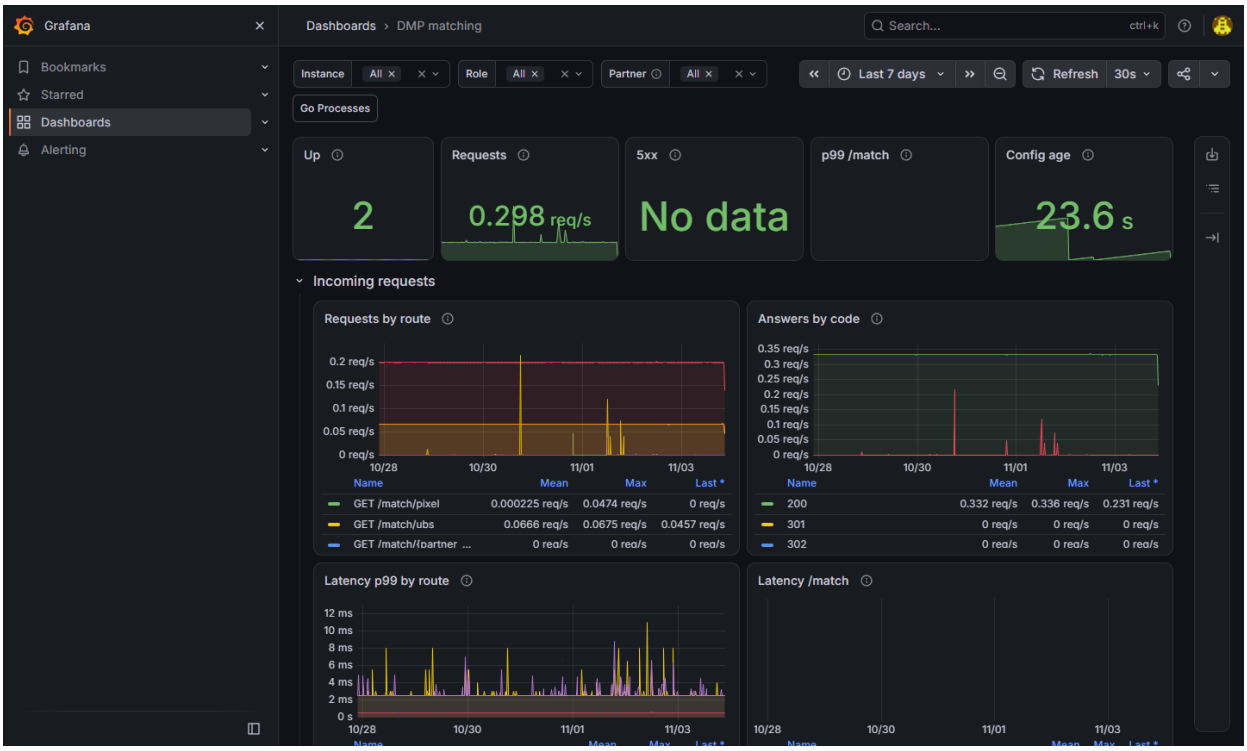


Рисунок 8 — Дашборд «DMP matching» в Grafana

3. Руководство для проведения проверки эксплуатации ПО

Ниже — последовательность действий, которая проверяет весь путь данных: создание партнёров и сегментов в UI, синхронизацию кук, попадание пользователя в сегменты и выдачу сегментов партнёру. Запросы из п. 3.3 выполняются в браузере и в приложении Postman (скачать можно на официальном сайте — https://www.postman.com) или командой `curl`.

Названия SSP и DSP уникальны. Если название из примера уже занято (ошибка «уже существует»), добавьте к нему цифры, например `check-ssp2`, и используйте новое название во всех ссылках ниже.

3.1. Авторизация

Перейдите по ссылке https://ui.dmp.melvad.com

Для авторизации вам понадобится:

- Пользователь: **review@dmp.melvad.com**
- Пароль: **YckmffWrsDK4ssgV**

3.2. Создание основных сущностей

3.2.1. Создание рекламодателя

Перейдите в меню «Пользователи» и нажмите кнопку «Создать». Сначала выберите роль, затем заполните поля:

- Роль – Рекламодатель
- Имя – Иван Иванов
- Email – ivanov@example.com (любой адрес, письмо не отправляется)
- Пароль – Check-2026 (не короче 8 символов)
- Внешний ID – можно не заполнять

Нажмите «Сохранить». Рекламодатель появится в списке пользователей, под его email и паролем можно войти в UI и увидеть интерфейс рекламодателя.

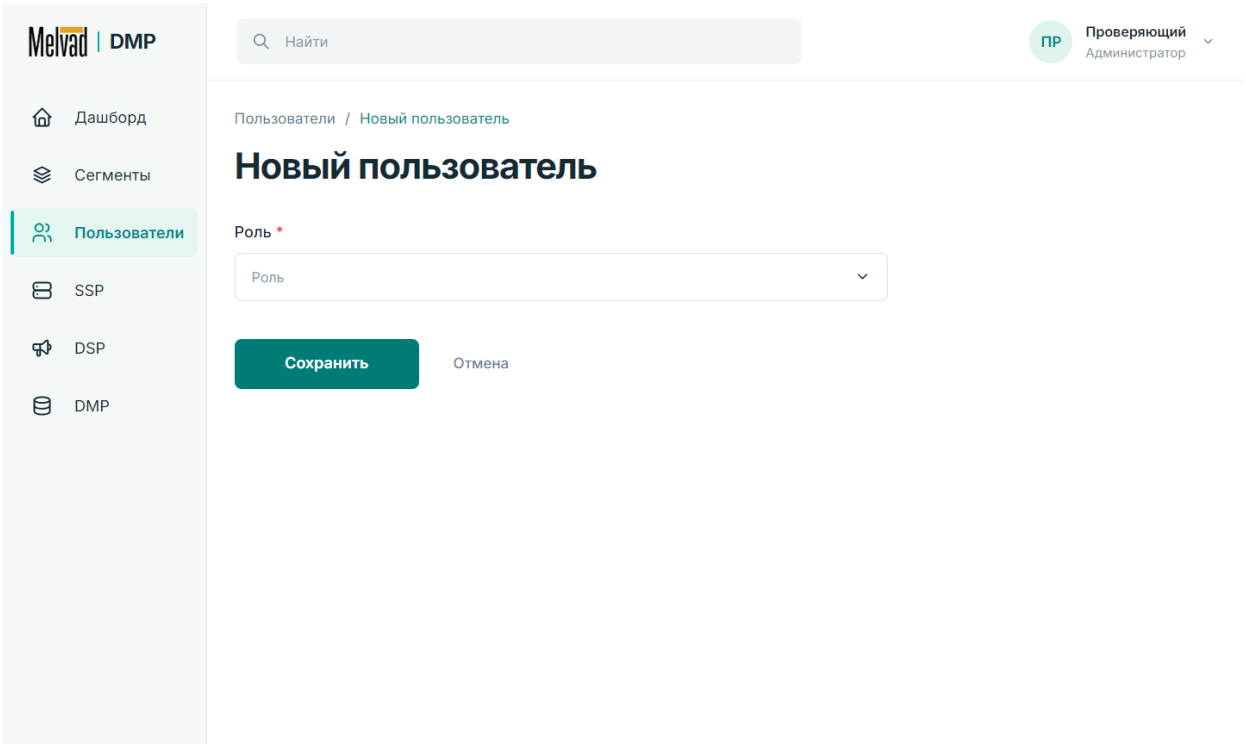


Рисунок 9 — Новый пользователь

3.2.2. Создание SSP

Перейдите в меню «SSP» и нажмите кнопку «Создать»:

- Название – check-ssp (латиницей: буквы, цифры, «.», «-», «_»)
- Внешний ID – check-ssp-1

Нажмите «Сохранить». В списке SSP нажмите кнопку </> у созданной SSP — откроется синк-ссылка:

```
https://match.dmp.melvad.com/match/check-ssp?id={user_id}&rt={REDIRECT_TO_ENCODE_URL}
```

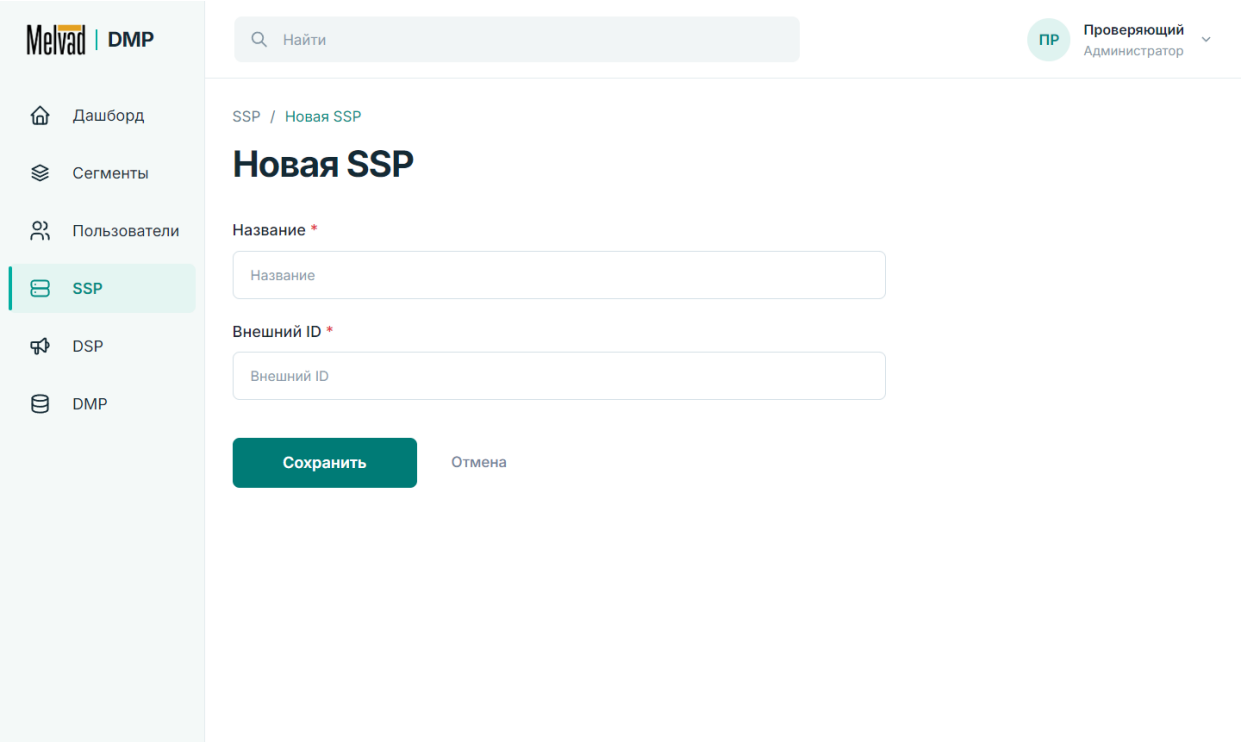


Рисунок 10 — Новая SSP

3.2.3. Создание DSP

Перейдите в меню «DSP» и нажмите кнопку «Создать»:

- Название – check-dsp
- Внешние ID – check-dsp-1
- URL синхронизации – https://example.com/sync?uid={{oui}}
- Тип синхронизации – Пиксель

Нажмите «Сохранить». Кнопка </> у DSP показывает ссылку возврата:

`https://match.dmp.melvad.com/match/check-dsp?id={user_id}`.

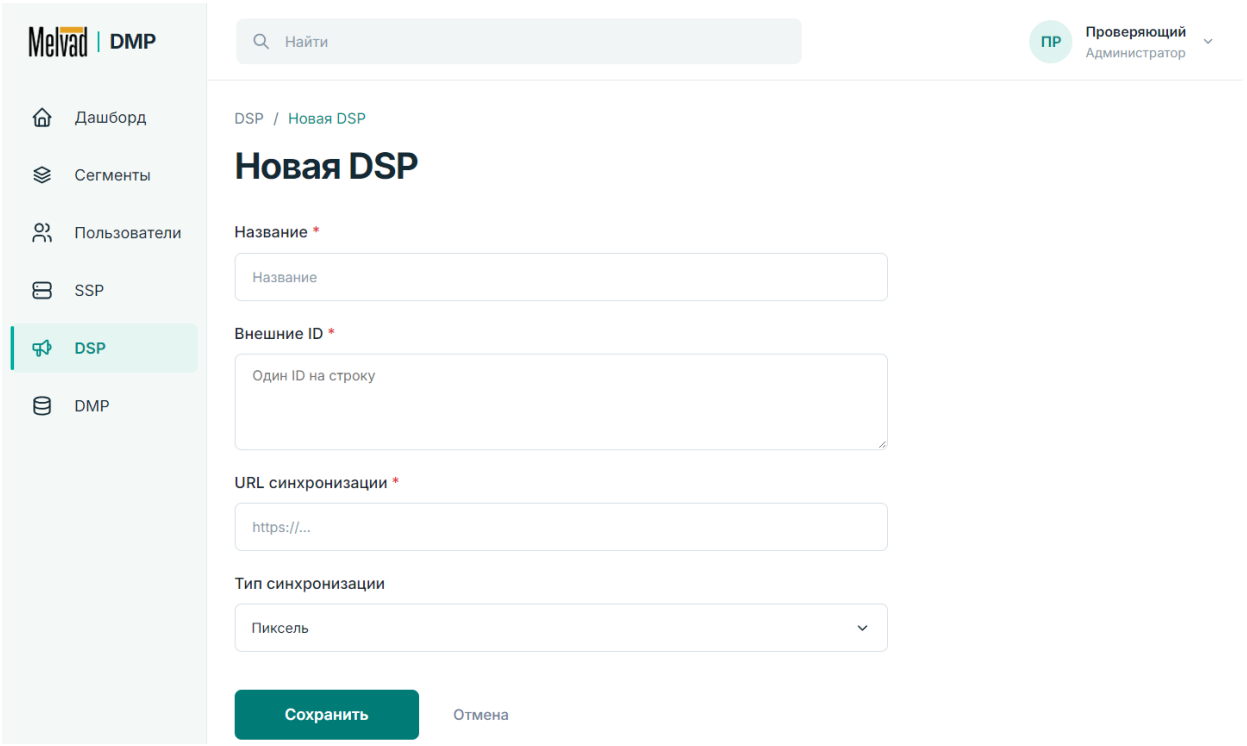


Рисунок 11 — Новая DSP

3.2.4. Создание пиксельного сегмента

Перейдите в меню «Сегменты» и нажмите кнопку «Создать»:

- Название – Посетители сайта
- Владелец – выбрать из списка рекламодателя Иван Иванов
- Тип – Пиксель

Нажмите «Сохранить». Запомните ID сегмента из первой колонки списка (далее — `<ID пикселя>`). Кнопка `</>` показывает код пикселя для сайта:

```
<img src="https://match.dmp.melvad.com/match/pixel?sadd=<ID пикселя>" width="1" height="1"
style="display:none" alt="">
```

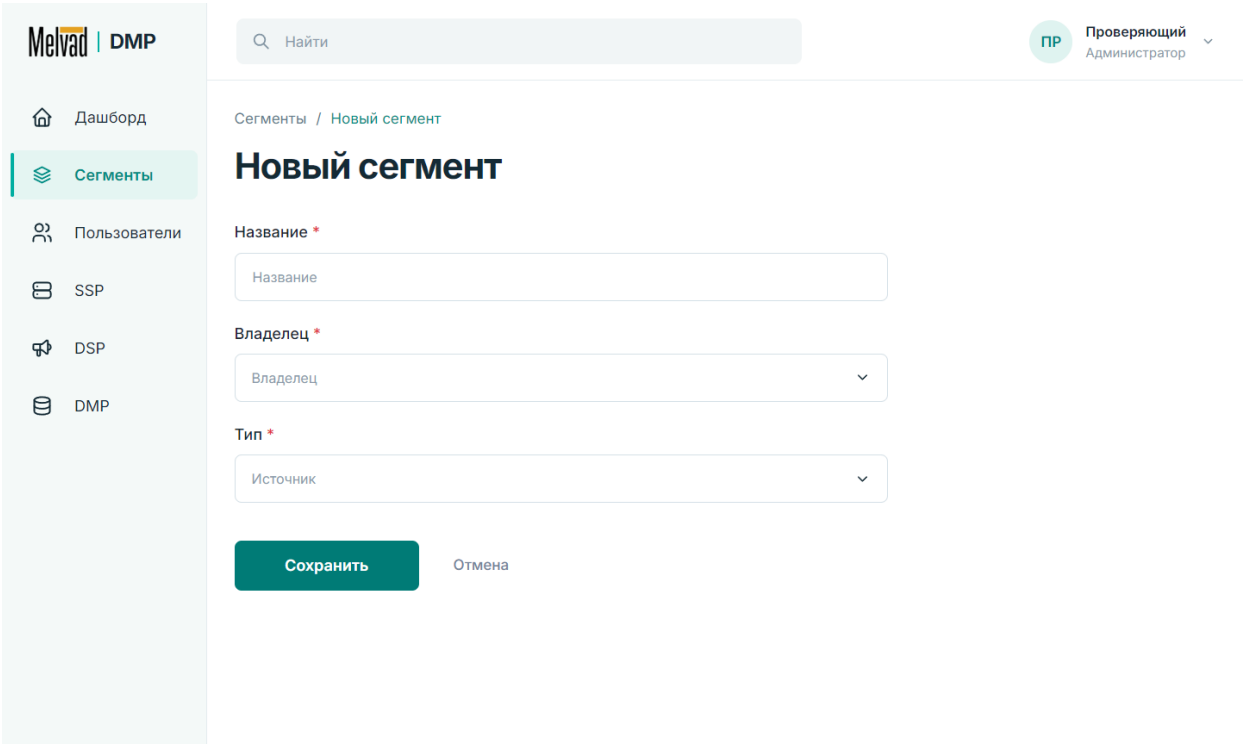


Рисунок 12 — Новый сегмент

3.3. Проверка синхронизации кук и сегментов

Шаги 3.3.1–3.3.3 выполняются в одном браузере (Chrome, Edge или Firefox, сторонние cookie не заблокированы): DMP узнаёт пользователя по куке `oui`.

3.3.1. Входящий матчинг SSP

Откройте в браузере синк-ссылку SSP, в которой вместо `{user_id}` подставлен идентификатор пользователя у SSP (`check-user-1`), а вместо `{REDIRECT_TO_ENCODE_URL}` — закодированная ссылка возврата `https://example.com/sync?dmp_uid={{oui}}`:

```
https://match.dmp.melvad.com/match/check-ssp?id=check-user-
1&rt=https%3A%2F%2Fexample.com%2Fsync%3Fdmp_uid%3D%7B%7Boui%7D%7D
```

Ожидаемый результат: браузер перенаправлен на страницу example.com, в адресной строке — идентификатор пользователя DMP:

```
https://example.com/sync?dmp_uid=ZjQyNzBhYmI0OWI0ODM3MA%3D%3D
```

Скопируйте значение после `dmp_uid=` как есть (с `%3D%3D` на конце) — далее `<oui>`. DMP записала пару «check-user-1 у SSP check-ssp — <oui>» и поставила браузеру куку `oui`.

Повторное открытие той же ссылки в течение срока ресинхронизации (настройка окружения) перенаправляет браузер не на `rt`, а на синк-ссылку одной из активных DMP: пользователь уже синхронизирован.

3.3.2. Пиксель сегмента

В том же браузере откройте ссылку пикселя (подставьте ID сегмента):

```
https://match.dmp.melvad.com/match/pixel?sadd=<ID пикселя>
```

Ожидаемый результат: пустая страница (ответ 200). Пользователь с кукой `oui` добавлен в пиксельный сегмент на 30 дней.

3.3.3. Возврат от DSP

В том же браузере откройте ссылку возврата DSP с идентификатором пользователя у DSP `dsp-user-1`:

```
https://match.dmp.melvad.com/match/check-dsp?id=dsp-user-1
```

Ожидаемый результат: пустая страница (ответ 200). DMP записала пару «dsp-user-1 у DSP check-dsp — <oui>».

3.3.4. Сегмент по идентификаторам SSP

Создайте на компьютере текстовый файл `ssp-ids.csv` (кодировка UTF-8) из двух строк:

```
check-user-1
check-user-unknown
```

В меню «Сегменты» нажмите «Создать»: Название – Пользователи SSP, Владелец – Иван Иванов, Тип – SSP, SSP – check-ssp, файл – `ssp-ids.csv`. Нажмите «Сохранить».

Ожидаемый результат: «загружено 1 из 2» — пользователь `check-user-1` синхронизирован с SSP на шаге 3.3.1 и попал в сегмент, `check-user-unknown` неизвестен DMP и пропущен. Запомните ID сегмента (далее — `<ID SSP-сегмента>`). В колонке «Кол-во записей» — 1.

3.3.5. Получение сегментов пользователя (запрос партнёра)

Методы `/match/ubs` и `/match/ubu` вызывают серверы партнёров, они работают только по http (по https — ответ 403). Их можно вызывать по домену (`http://match.dmp.melvad.com`) или по IP-адресу сервера; в примерах — IP-адрес, на нём браузер не переключается на https.

В браузере, Postman (GET) или командой curl:

```
curl "http://81.163.21.79/match/ubs?user_id=<oui>&user_type=1"
```

Ожидаемый ответ — оба сегмента пользователя:

```
{"segments":[<ID пикселя>,<ID SSP-сегмента>]}
```

3.3.6. Идентификатор пользователя у DSP (запрос партнёра)

В Postman выполните POST-запрос на адрес `http://81.163.21.79/match/ubu` с телом (Body → raw → JSON). Значение `oui` здесь передаётся в раскодированном виде: `%3D` заменяется на `=`.

```
{
    "oui": "ZjQyNzBhYmI0OWI0ODM3MA==",
    "dsp_ids": ["check-dsp-1"]
}
```

То же командой curl:

```
curl -X POST "http://81.163.21.79/match/ubu" -H "Content-Type: application/json" \
     -d "{\"oui\": \"<oui с = вместо %3D>\", \"dsp_ids\": [\"check-dsp-1\"]}"
```

Ожидаемый ответ — идентификатор пользователя у DSP из шага 3.3.3:

```
{"check-dsp-1":"dsp-user-1"}
```

3.3.7. Исходящий матчинг

Откройте в браузере https://match.dmp.melvad.com/usermatch. Ответ — JavaScript-код для сайтов. Если браузер ещё не синхронизировался с DSP, в коде есть синк-ссылка DSP check-dsp с идентификатором пользователя вместо `{{oui}}`:

```
p("https://example.com/sync?uid=ZjQyNzBhYmI0OWI0ODM3MA%3D%3D");
```

Пока не истёк срок ресинхронизации, повторная загрузка скрипта вместо DSP загружает синк-ссылку одной из DMP.

3.3.8. Проверка на дашборде

Через несколько минут откройте «Дашборд» в UI с периодом «Сегодня»: на графике есть запросы SSP и DSP из шагов 3.3.1 и 3.3.3, на диаграмме синков — партнёры check-ssp и check-dsp. Те же запросы видны в Grafana (п. 2.9) на панелях «Requests by route» и «Synced cookies by partner over the range».

4. Описание развернутого экземпляра инфраструктуры ПО

4.1. Общее описание

Проверочный стенд (окружение dev) представляет из себя несколько docker-контейнеров на сервере 81.163.21.79 (Ubuntu 24.04). Ниже представлен полный список контейнеров DMP с их кратким описанием. Схема движения данных представлена в Приложении 1.

Сервисы:

Frontend

- ui-dev (Nginx) — веб-интерфейс на Vue 3, проксирует запросы `/api/*` к API.

Backend:

- api-dev — API: пользователи, справочники SSP, DSP и DMP, сегменты, импорт сегментов DMP, статистика дашборда; владеет схемой баз данных.
- matching-dev-dmp и matching-dev-replica-dmp — сервис матчинга (два экземпляра): синхронизация кук, пиксели сегментов, выдача сегментов партнёрам.

Базы данных:

- dmp-postgres-dev (PostgreSQL 17.5) — пользователи, справочники партнёров, сегменты, сессии UI.
- dmp-scylla-dev (ScyllaDB 2026.1) — пары синхронизации кук, участники сегментов.

Мониторинг:

- dmp-prometheus-dev — сбор и хранение метрик (90 дней).
- dmp-grafana-dev — дашборды.
- dmp-postgres-exporter-dev — метрики PostgreSQL.

Публикация:

- nginx — общий контейнер сервера: принимает HTTP/HTTPS на портах 80 и 443 и проксирует домены DMP на контейнеры. Сертификат Let's Encrypt доменов DMP продлевается ежедневно по расписанию CI/CD.

Адреса:

Адрес	Назначение
https://ui.dmp.melvad.com	веб-интерфейс
https://api.dmp.melvad.com	API, документация — /swagger/index.html
https://match.dmp.melvad.com	сервис матчинга, документация — /swagger/index.html
http://81.163.21.79 (только /match/ubs и /match/ubu)	сервис матчинга для серверов партнёров
https://grafana.dmp.melvad.com	мониторинг

Порты, на которых запущены модули (снаружи сервера доступны только 80 и 443 через nginx):

- UI (ui-dev): 18083
- API (api-dev): 18082, 15061 (debug), 15071 (метрики)
- matching-dev-dmp: 18081, 15060 (debug), 15070 (метрики)
- matching-dev-replica-dmp: 18084, 25160 (debug), 25170 (метрики)
- PostgreSQL (dmp-postgres-dev): 25432

- ScyllaDB (dmp-scylla-dev): 29042 (CQL), 29180 (метрики)
- Prometheus: 19090
- Grafana: 13000
- postgres_exporter: 19187
- nginx: 80, 443

4.2. Подробное описание контейнеров сервисов, их процессов и компонентов

4.2.1. UI

Контейнер: ui-dev

Описание

Веб-интерфейс на базе Vue 3, раздаваемый через Nginx (сеть хоста, 127.0.0.1:18083). Проксирует API запросы к api-dev, поэтому UI и API открываются с одного домена.

Процессы

- nginx (master) — главный процесс Nginx, управляет worker процессами.
- nginx (worker) — рабочие процессы, обрабатывают HTTP запросы.

Компоненты

- Nginx — веб-сервер.
- Vue 3 приложение (SPA), графики Chart.js.

Взаимодействие

- Принимает HTTP запросы от общего nginx сервера (домен ui.dmp.melvad.com).
- Статические файлы (JS, CSS, изображения) отдаёт напрямую.
- API запросы (/api/*) проксирует на api-dev (127.0.0.1:18082).

4.2.2. API

Контейнер: api-dev

Описание

REST API на Go для управления пользователями, справочниками SSP, DSP и DMP, сегментами; вход в UI по email и паролю; статистика дашборда. Владеет схемой PostgreSQL и Scylla и применяет миграции при старте.

Процессы

- api — обработка HTTP запросов и бизнес-логика.
- Фоновые задачи: загрузка и удаление участников сегментов в Scylla, пересчёт числа пользователей в сегментах (раз в час), импорт сегментов DMP из S3 (по расписанию, каждый час).

Компоненты

- HTTP сервер на порту 18082.
- Debug сервер на порту 15061 (только 127.0.0.1).
- Метрики Prometheus на порту 15071.
- Swagger/OpenAPI — документация API.
- Утилиты в контейнере: migrate (миграции), useradd (создание пользователя).

Взаимодействие

- PostgreSQL — пользователи, справочники, сегменты, сессии UI; триггеры сообщают сервису матчинга об изменениях (NOTIFY).

- ScyllaDB — запись и удаление участников сегментов, поиск пар SSP для сегментов по идентификаторам SSP.
- Prometheus — число синхронизаций для дашборда.
- S3 — файлы сегментов DMP (необязательно).

4.2.3. Matching

Контейнер: matching-dev-dmp, matching-dev-replica-dmp

Описание

Сервис матчинга на Go: публичные методы, к которым приходят браузеры пользователей, SSP, DSP и DMP. Запущен в двух экземплярах, общий nginx распределяет запросы между ними. Конфиг (активные SSP, DSP, DMP, пиксельные сегменты) держит в памяти и перечитывает по уведомлению PostgreSQL.

Процессы

- matching — обработка запросов /match, /usermatch, /match/pixel, /match/ubs, /match/ubu.
- Подписка на уведомления PostgreSQL (LISTEN matching_config) и перечитывание конфига.

Компоненты

- HTTP сервер на порту 18081 (реплика — 18084).
- Debug сервер на порту 15060 (реплика — 25160), только 127.0.0.1: конфиг экземпляра, профили.
- Метрики Prometheus на порту 15070 (реплика — 25170).
- Клиент ScyllaDB (shard-aware), таймаут горячего пути 10 мс.

Взаимодействие

- PostgreSQL — только чтение конфига.
- ScyllaDB — запись и чтение пар синхронизации, запись участников пиксельных сегментов, чтение сегментов пользователя.

4.2.4. PostgreSQL

Контейнер: dmp-postgres-dev

Описание

Реляционная БД PostgreSQL 17.5, база matching. Роли: matching_migrator (изменение схемы), matching_app (работа сервисов), dmp_monitoring (мониторинг).

Процессы

- postgres — главный процесс и служебные процессы (checkpointer, background writer, walwriter, autovacuum).

Компоненты

- Порт 25432 на 127.0.0.1.

Взаимодействие

- api-dev — чтение и запись.
- matching — чтение конфига, уведомления NOTIFY.
- dmp-postgres-exporter-dev — метрики.

4.2.5. ScyllaDB

Контейнер: dmp-scylla-dev

Описание

NoSQL БД ScyllaDB 2026.1, keyspace matching (одна нода). Хранит пары синхронизации кук (30 дней) и участников сегментов с TTL.

Процессы

- scylla — сервер БД.
- supervisord, scylla-housekeeping — служебные процессы образа.

Компоненты

- CQL на порту 29042 (127.0.0.1).
- Метрики на порту 29180 (127.0.0.1).

Взаимодействие

- matching — пары синхронизации, пиксельные сегменты, чтение сегментов пользователя.
- api-dev — участники сегментов, подсчёт и удаление.

4.2.6. Мониторинг

Контейнер: dmp-prometheus-dev, dmp-grafana-dev, dmp-postgres-exporter-dev

Описание

Prometheus собирает метрики сервисов, сервера и баз и хранит их 90 дней; Grafana показывает дашборды (https://grafana.dmp.melvad.com); postgres_exporter отдаёт метрики PostgreSQL.

Процессы

- prometheus — сбор и хранение метрик.
- grafana — веб-интерфейс дашбордов.
- postgres_exporter — метрики PostgreSQL.

Компоненты

- Prometheus на порту 19090, Grafana на порту 13000, postgres_exporter на порту 19187 (все только 127.0.0.1).
- Конфиги и дашборды встроены в образы.

Взаимодействие

- Метрики api-dev, matching, ScyllaDB, postgres_exporter, а также node-exporter и cAdvisor сервера.
- api-dev читает из Prometheus число синхронизаций для дашборда UI.

4.3. Настройка удаленного доступа к развернутому экземпляру ПО

Для проверки создан отдельный пользователь сервера **dmp-review**. Он не имеет прав администратора и прямого доступа к Docker: состояние, логи и данные DMP он смотрит командой `sudo dmp`, которая работает только с контейнерами DMP и только на чтение. Подключение можно осуществить двумя способами.

- Через пароль.

IP-адрес – 81.163.21.79
Логин – dmp-review
Пароль – gHCWjUkDXetEQVYjaA9K
Пример подключения: `ssh dmp-review@81.163.21.79`

- Через ssh-ключ.

Для подключения по ключу необходимо скачать файл с именем `id_ed25519_dmp_review`, представленный по ссылке https://cdn1.melvad.com/ssh/id_ed25519_dmp_review. Файл может быть перемещен в рабочий каталог (или в текущий рабочий каталог вашего терминала).

Затем необходимо выставить ему минимальные права:

```
chmod 0400 id_ed25519_dmp_review
```

После этого можно подключаться (ключ -i позволяет задать путь и имя файла с ключом):

```
ssh -i id_ed25519_dmp_review dmp-review@81.163.21.79
```

Команды на сервере (пароль для sudo не нужен):

Команда	**Что показывает**
`sudo dmp ps`	контейнеры DMP: состояние, число перезапусков, образ
`sudo dmp stats`	CPU, память, сеть и диск контейнеров DMP
`sudo dmp inspect <контейнер>`	образ, состояние, сеть, порты, тома и ресурсы контейнера
`sudo dmp top <контейнер>`	процессы контейнера
`sudo dmp logs <контейнер> [N]`	последние N строк лога (по умолчанию 100)
`sudo dmp volumes`	тома DMP и их размер
`sudo dmp health`	проверки доступности сервисов
`sudo dmp config`	конфиг сервиса матчинга: активные SSP, DSP, DMP и пиксельные сегменты
`sudo dmp sql "<запрос>"`	запрос к PostgreSQL DMP (база matching), только чтение
`sudo dmp cql "<запрос>"`	запрос SELECT или DESCRIBE к ScyllaDB DMP (keyspace matching)

Примеры:

```
sudo dmp ps
sudo dmp logs api-dev 50
sudo dmp sql "select id, name, status, external_id from ssps order by id"
sudo dmp sql "select id, name, type, ttl, status from segments order by id desc limit 10"
sudo dmp cql "describe tables"
sudo dmp cql "select pui, partner_role, partner_id, oui, ttl(oui) from matching limit 10"
```

4.4. Расположение файлов приложения

Приложение работает в Docker-контейнерах (команда `sudo dmp ps` на сервере покажет список, описанный в пункте 4.1). Образы собираются в CI/CD и загружаются из реестра gitlab.melvad.com:5050, данные хранятся в Docker volumes и смонтированных каталогах хоста.

4.4.1. Список volumes

Volume	Контейнер	Назначение	Путь внутри контейнера
dmp-infra-dev_postgres	dmp-postgres-dev	Данные PostgreSQL	/var/lib/postgresql/data
dmp-infra-dev_scylla	dmp-scylla-dev	Данные ScyllaDB	/var/lib/scylla
dmp-monitoring-dev_prometheus	dmp-prometheus-dev	Метрики Prometheus (90 дней)	/prometheus
dmp-monitoring-dev_grafana	dmp-grafana-dev	Данные Grafana (пользователи, плагины)	/var/lib/grafana
dmp-monitoring-secrets-dev	dmp-postgres-exporter-dev	Пароль роли мониторинга PostgreSQL	/secrets

4.4.2. Смонтированные каталоги хоста

Путь на хосте	Контейнер	Назначение
/home/nginx/config/nginx.conf	nginx	общий конфиг nginx, подключает конфиг DMP
/home/nginx/config/conf.d/dmp.conf	nginx	конфигурация доменов DMP (UI, API, matching, Grafana)
/home/nginx/certs/dmp/	nginx	сертификат Let's Encrypt доменов DMP
/usr/local/bin/dmp	—	команда просмотра DMP для пользователя dmp-review (п. 4.3)

Контейнеры ui-dev, api-dev и matching не используют каталоги хоста: настройки передаются им переменными окружения при развёртывании из CI/CD, логи уходят через драйвер fluentd в Loki сервера.

Приложение 1. Общая схема контейнеров и движения данных между ними

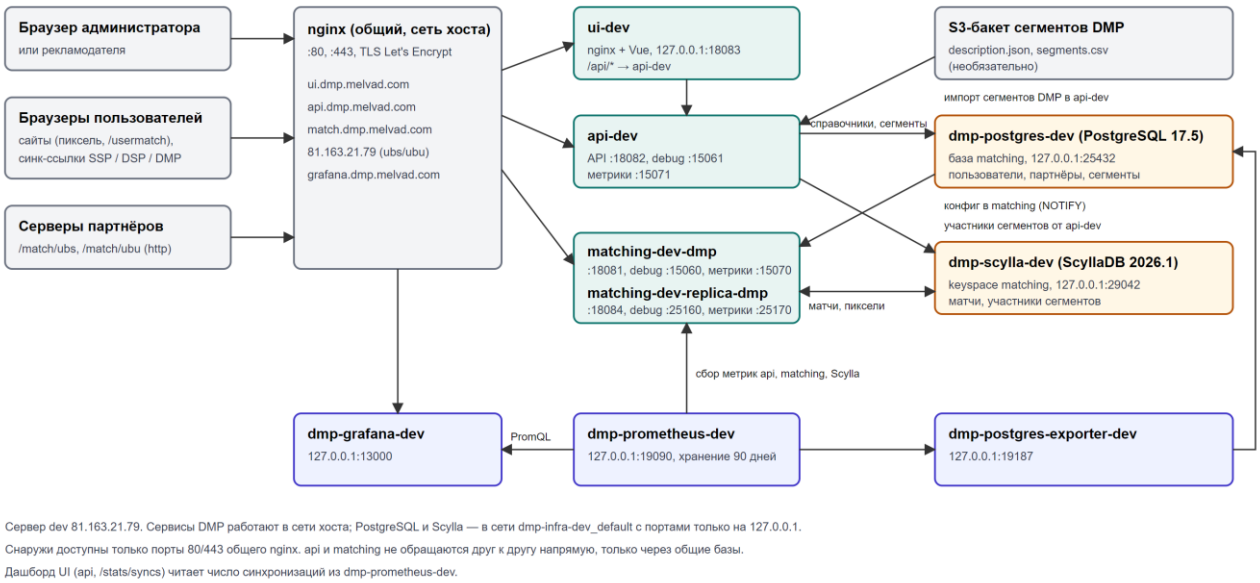


Рисунок 13 — Схема контейнеров DMP на сервере