

Дедупликация сделок: anti_double_deals

Для разработчиков. Состояние на 24.09.2026. Подробности, переменные и выкатка — [README Buharestskie/anti_double_deals](#).

Коротко

anti_double_deals не даёт заводить в amoCRM ([donula.amocrm.ru](#)) повторные сделки на одного клиента и раскладывает новые заявки по этапам. На каждую новую сделку он сливает контакты с тем же телефоном, решает, какая сделка лишняя, убирает её в этап для дублей и переносит оставшуюся в рабочий этап. Всё, что сделал, пишет в журнал.

Как обрабатывается сделка

1. **Вебхук о новой сделке** (штатный amoCRM или Sensei) → `api` ставит задачу в Celery на каждую сделку. `worker` берёт блокировку сделки в Redis.
2. **Все телефоны иностранные** (казахстанские `+7 7xx` тоже) → этап для дублей, тег «Иностранный номер». Конец.
3. **Слияние контактов.** Ищутся контакты с теми же последними 10 цифрами телефона. Выживает самый старый, остальные сливаются в него. В публичном API слияния нет — используется внутренний `/ajax` amoCRM, для которого `worker` входит в веб-интерфейс браузером Playwright под служебным пользователем; сессия хранится в Redis. После слияния все сделки группы заново привязываются к выжившему контакту.
4. **Кто дубль.** Если у контакта есть более старая *активная* сделка в одной из 11 «рабочих» воронок (не в статусе 143) — дублем считается новая: этап для дублей, тег «Дубль новая». Иначе в этап для дублей уходят старые неактивные сделки с тегом «Дубль старая», а новая остаётся.
5. **Маршрутизация.** Оставшаяся сделка переносится в этап группы А или В по полю-списку 664291 (пять значений — группа А, остальное и пустое — В).
6. **Журнал.** Исход пишется в `lead_routing_log`, одна строка на сделку: «Маршрутизация», «Дубль новая», «Иностранный номер», «В обработке». Страница «Журнал сделок» показывает последние 100 записей со ссылками на сделки.

Страховка: `beat` раз в 30 минут прогоняет сделки старше 5 минут, застрявшие на входном этапе, и раз в 5 минут лечит сделки, больше 10 минут висящие «В обработке».

Откуда данные: историю (кто ещё с этим телефоном, сделки контакта, справочники) сервис читает из реплики `amo_db`, с откатом на API при ошибке. Всё, от чего зависит решение, и всё, что читается сразу после записи, — из amoCRM напрямую: реплика отстаёт на секунды, в худшем случае на минуты.

Где живёт

Сервер `global`, compose из шести контейнеров: `api`, `worker`, `beat`, `frontend`, `postgres`, `redis`. Наружу — через шлюз `donu1a26.ru`. Каталог на сервере **без git**: выкатка копированием, порядок — в README. GitHub Actions `deploy.yml` продю не соответствует — коммиты в `main` пока только с `[skip ci]`. База бэкапится ежедневно, хранение 14 дней. В учёте обращений сервис называется «Журнал маршрутизации лидов».

На что наступают

- **Бизнес-правила защиты в код** (`backend/app/constants.py`): рабочие воронки, поле 664291, значения группы А, этапы групп, входной этап страховки. Поменяли воронку или этап в amoCRM — нужна выкатка. Id этапов печатает `backend/scripts/dump_pipelines.py`.
- **Слияние держится на входе через браузер.** Включат служебному пользователю 2FA, сменяют пароль или amoCRM поменяет форму входа — слияние встанет, сделки с дублями будут повторяться до 5 раз и зависать «В обработке». Скриншот и HTML неудачного входа — в `/tmp` внутри контейнера `worker`; сброс сессии — удалить ключ `amocrm:ajax_session_v2` в Redis.
- **Локальный запуск с боевыми ключами обрабатывает боевые сделки** параллельно с продом и не видит его блокировок. Локально — только с тестовыми ключами.
- **Одна сделка может идти минутами:** после слияния `worker` ждёт до 120 секунд, пока amoCRM удалит слитые контакты; блокировка живёт 600 секунд.
- **Лимит 7 запросов в секунду считается на процесс**, а у `worker` несколько процессов — вместе они могут упереться в лимит amoCRM; спасают повторы на 429.
- **Реплика пускает не всех:** IP сервера должен быть в списке доступа `amo_db`, иначе сервис молча работает только через API и тратит квоту. Видно по `amo_db read ... failed` в логах.
- **Имя каталога = имена контейнеров и тома.** Бэкап ищет контейнер `anti_double_deals-postgres-1`; выкатка в каталог с другим именем сломает бэкап и поднимет пустую базу.
- **Redis без тома** — очередь Celery пропадает при пересоздании. Сделки со входного этапа подберёт страховка, остальные — вручную.

- **Страница журнала — dev-сервер Vite**, без сборки; в интерфейсе только последние 100 записей.
-

Версия #1

Admin создал 2026-09-24 14:46:59 UTC

Admin обновил 2026-09-24 14:47:00 UTC