Руководство администратора

1. Учетные записи

Первого администратора создают из консоли:

```
cd /opt/local-llm
set -a; . ./.env; set +a
venv/bin/python -m app.cli create-user admin --admin
```

Список пользователей:

```
venv/bin/python -m app.cli list-users
```

Смена пароля:

```
venv/bin/python -m app.cli set-password username
```

После входа администратор может создавать, блокировать и разблокировать учетные записи, менять роли и пароли на странице «Администрирование».

2. Состояние приложения

Локальная проверка:

```
curl -fsS http://127.0.0.1:8130/api/config
curl -fsS http://127.0.0.1:8130/healthz
systemctl status own-chat.service
journalctl -u own-chat.service -n 100 --no-pager
```

Публичный /healthz рекомендуется закрыть обратным прокси.

3. Управление моделями

При включенном шлюзе административная страница показывает активную модель, состояние, очередь и доступную телеметрию. Кнопка переключения выгружает предыдущую модель после завершения текущих запросов и запускает выбранную.

Во время переключения запросы остаются в очереди. Не следует одновременно запускать серверы моделей вручную на тех же устройствах.

Проверка шлюза:

curl -fsS http://127.0.0.1:8120/health

systemctl status model-gateway.service

journalctl -u model-gateway.service -n 100 --no-pager

Если установлен Bearer-ключ, методы /v1/*, /metrics и /admin/* вызываются с заголовком Authorization: Bearer ....

4. Резервное копирование

Запуск копирования вручную в проверенной установке:

systemctl start own-chat-backup.service

journalctl -u own-chat-backup.service -n 20 --no-pager

Проверка расписания:

systemctl list-timers own-chat-backup.timer

Инструмент по умолчанию хранит 14 архивов. Копии на том же диске защищают от логической ошибки, но не от отказа диска. Необходимо регулярно переносить их во внешнее защищенное хранилище и проверять восстановление.

5. Восстановление

Сначала проверить архив без изменения рабочей базы:

tools/restore_db.sh --to /tmp/check.db data/backups/chat-YYYYmmdd-HHMMSS.db.gz

Для восстановления рабочей базы выполнить от root:

tools/restore_db.sh data/backups/chat-YYYYmmdd-HHMMSS.db.gz

Сценарий останавливает приложение, сохраняет текущую базу с суффиксом .broken-<время>, устанавливает проверенную копию и снова запускает службу.

6. Обновление

1. Проверить свободное место и состояние резервных копий.
2. Зафиксировать хеш текущего выпуска.
3. Создать резервную копию базы и конфигурации.
4. Проверить новый выпуск в отдельном каталоге.
5. Выполнить автоматические тесты.
6. Установить файлы и перезапустить затронутые службы.
7. Проверить вход, список моделей, создание чата и сохранение ответа.
8. При ошибке вернуть предыдущий выпуск.

7. Диагностика

Признак	Проверка
Сайт не открывается	Состояние обратного прокси и own-chat.service, локальный порт 8130
Вход не работает	Журнал приложения, наличие пользователя, время сервера, права на SQLite
Нет моделей	GET /v1/models, LLM_BASE_URL, ключ API
Ответ не начинается	Состояние шлюза, очередь, контейнер сервера, память ускорителя
История не сохраняется	Права на каталог данных, место на диске, PRAGMA integrity_check
Шлюз в FAILED	failed_reason, журнал службы, готовность Docker и сервера модели

8. Защита конфигурации

- Файлы .env и /etc/model-gateway.env должны быть недоступны обычным пользователям.

- Порты приложения, шлюза и сервера модели следует привязывать к loopback или закрытой сети.

- Публичный доступ предоставляется только через HTTPS.

- Root-доступ не требуется пользователям приложения.

- Доступ к Docker эквивалентен широким административным правам и предоставляется только оператору шлюза.

- Перед публикацией логов удалить токены, адреса и пользовательские тексты.

9. Контроль базы

Проверка целостности:

sqlite3 /opt/local-llm/data/chat.db 'PRAGMA integrity_check;'

Ожидаемый результат: ok. Проверку рабочей базы с активным WAL предпочтительно выполнять через резервную копию, созданную SQLite Backup API.