🗄 Руководство по работе с PostgreSQL

📋 Обзор

Проект использует PostgreSQL 16 в Docker контейнере. База данных автоматически запускается вместе с приложением через Docker Compose.

⚙ Конфигурация

Переменные окружения (.env)

```
# Настройки PostgreSQL
POSTGRES_USER=globalit24
POSTGRES_PASSWORD=changeme123  # ОБЯЗАТЕЛЬНО ИЗМЕНИТЕ!
POSTGRES_DB=globalit24_db

# URL для подключения
DATABASE_URL=postgresql://globalit24:changeme123@postgres:5432/global
```

⚠ **ВАЖНО:** Обязательно измените `POSTGRES_PASSWORD` на сильный пароль перед развертыванием!

🚀 Автоматический запуск

PostgreSQL запускается автоматически при выполнении:

```
./deploy.sh
# или
docker compose up -d
```

📊 Структура базы данных

Таблица: contact_submissions

Хранит все заявки с формы обратной связи.

```
CREATE TABLE contact_submissions (
```

```
    id VARCHAR PRIMARY KEY,           -- Уникальный идентификатор
        (cuid)
    name VARCHAR NOT NULL,            -- Имя клиента
    phone VARCHAR NOT NULL,           -- Телефон
    email VARCHAR,                    -- Email (опционально)
    service_type VARCHAR,             -- Тип услуги (опционально)
    message TEXT,                     -- Сообщение (опционально)
    created_at TIMESTAMP DEFAULT NOW() -- Дата создания
);
```

🔧 Управление базой данных

1. Подключение к PostgreSQL контейнеру

```
# Войти в контейнер PostgreSQL
docker exec -it global-it24-postgres psql -U globalit24 -d
        globalit24_db
```

2. Основные SQL команды

```
-- Посмотреть все таблицы
\dt

-- Описание таблицы
\d contact_submissions

-- Посмотреть все заявки
SELECT * FROM contact_submissions ORDER BY created_at DESC;

-- Посмотреть последние 10 заявок
SELECT * FROM contact_submissions ORDER BY created_at DESC LIMIT 10;

-- Посмотреть заявки за сегодня
SELECT * FROM contact_submissions
WHERE created_at::date = CURRENT_DATE
ORDER BY created_at DESC;

-- Статистика по типам услуг
SELECT service_type, COUNT(*) as count
FROM contact_submissions
GROUP BY service_type
ORDER BY count DESC;

-- Выход
\q
```

3. Резервное копирование базы данных

```bash
# Создать бэкап
docker exec -t global-it24-postgres pg_dump -U globalit24
        globalit24_db > backup_$(date +%Y%m%d_%H%M%S).sql

# Или автоматический скрипт
./backup-db.sh
```

4. Восстановление из бэкапа

```bash
# Восстановить из файла
docker exec -i global-it24-postgres psql -U globalit24 -d
        globalit24_db < backup_YYYYMMDD_HHMMSS.sql
```

5. Очистка старых данных (опционально)

```bash
# Удалить заявки старше 1 года
docker exec -it global-it24-postgres psql -U globalit24 -d
        globalit24_db -c "DELETE FROM contact_submissions WHERE
        created_at < NOW() - INTERVAL '1 year';"
```

🔍 Проверка состояния

Статус контейнера

```bash
# Проверка работы PostgreSQL
docker ps | grep postgres

# Логи PostgreSQL
docker logs global-it24-postgres

# Последние 50 строк логов
docker logs global-it24-postgres --tail 50 -f
```

Health Check

```bash
# Проверка готовности базы данных
docker exec global-it24-postgres pg_isready -U globalit24

# Должно вывести:
# /var/run/postgresql:5432 - accepting connections
```

Подключение к базе из приложения

```bash
# Проверка переменных окружения в контейнере приложения
docker exec global-it24-landing env | grep DATABASE_URL
```

🗃️ Prisma миграции

Применить миграции (после изменений в schema.prisma)

```
# Внутри контейнера приложения
docker exec -it global-it24-landing sh -c "cd /app && npx prisma
        migrate deploy"
```

Создать новую миграцию (в разработке)

```
cd nextjs_space
yarn prisma migrate dev --name название_миграции
```

Сгенерировать Prisma Client

```
cd nextjs_space
yarn prisma generate
```

Просмотр данных через Prisma Studio (только для разработки)

```
cd nextjs_space
yarn prisma studio
# Откроется на http://localhost:5555
```

🔐 Безопасность

Рекомендации:

1. **Измените пароль по умолчанию**

   ```
   # В .env файле установите сильный пароль
   POSTGRES_PASSWORD=your_strong_password_here_123!@#
   ```

2. **Ограничьте доступ к порту PostgreSQL**

   PostgreSQL работает внутри Docker сети и не доступен извне по умолчанию. Это правильно!

   Если нужен внешний доступ (не рекомендуется), добавьте:

   ```
   ports:
     - "127.0.0.1:5432:5432"  # Только localhost
   ```

3. **Регулярные бэкапы**

   Настройте автоматический бэкап через cron:

```
crontab -e
# Добавьте:
0 3 * * * cd /path/to/global_it24_landing && ./backup-db.sh
```

4. **Мониторинг места на диске**

```
# Проверка размера данных PostgreSQL
docker exec global-it24-postgres du -sh /var/lib/postgresql/data
```

🛠 Устранение неполадок

Проблема: Не могу подключиться к базе данных

```
# 1. Проверьте что контейнер запущен
docker ps | grep postgres

# 2. Проверьте логи
docker logs global-it24-postgres

# 3. Проверьте health check
docker inspect global-it24-postgres | grep -A 5 Health

# 4. Попробуйте перезапустить
docker compose restart postgres
```

Проблема: База данных не инициализируется

```
# 1. Остановите контейнеры
docker compose down

# 2. Удалите volume (ВНИМАНИЕ: потеря данных!)
docker volume rm global_it24_landing_postgres_data

# 3. Запустите снова
./deploy.sh
```

Проблема: Ошибка миграции Prisma

```
# 1. Проверьте DATABASE_URL
docker exec global-it24-landing env | grep DATABASE_URL

# 2. Сбросьте Prisma и примените миграции заново
docker exec -it global-it24-landing sh -c "cd /app && npx prisma
        migrate reset --skip-seed"
```

Проблема: Медленные запросы

```sql
-- Проверьте размер таблицы
SELECT
  schemaname,
  tablename,
  pg_size_pretty(pg_total_relation_size(schemaname||'.'||tablename)) AS
        size

FROM pg_tables
WHERE schemaname = 'public';

-- Создайте индексы для часто используемых полей
CREATE INDEX idx_created_at ON contact_submissions(created_at DESC);
CREATE INDEX idx_phone ON contact_submissions(phone);
```

📈 Мониторинг

Использование ресурсов

```bash
# Статистика контейнера PostgreSQL
docker stats global-it24-postgres --no-stream

# Размер базы данных
docker exec global-it24-postgres psql -U globalit24 -d globalit24_db
        -c "SELECT
        pg_size_pretty(pg_database_size('globalit24_db'));"
```

Количество записей

```bash
# Общее количество заявок
docker exec global-it24-postgres psql -U globalit24 -d globalit24_db
        -c "SELECT COUNT(*) FROM contact_submissions;"
```

🔄 Миграция данных

Экспорт данных в CSV

```bash
docker exec -t global-it24-postgres psql -U globalit24 -d
        globalit24_db -c "COPY contact_submissions TO STDOUT WITH
        CSV HEADER" > submissions_export.csv
```

Импорт данных из CSV

```bash
cat submissions_import.csv | docker exec -i global-it24-postgres
        psql -U globalit24 -d globalit24_db -c "COPY
        contact_submissions FROM STDIN WITH CSV HEADER"
```

📞 Полезные ссылки

- Документация PostgreSQL
- Документация Prisma
- Docker PostgreSQL Image

💡 Советы по производительности

1. **Регулярная очистка логов**

   ```
   docker compose logs --tail 100 > logs.txt
   ```

2. **VACUUM таблицы** (освобождение места после удаления данных)

   ```
   VACUUM ANALYZE contact_submissions;
   ```

3. **Мониторинг активных подключений**

   ```
   SELECT count(*) FROM pg_stat_activity WHERE datname =
           'globalit24_db';
   ```

Готово! База данных PostgreSQL полностью настроена и готова к использованию! 🚀