ДОКУМЕНТАЦИЯ ПО УСТАНОВКЕ

Программное обеспечение “Система предиктивной аналитики производственных параметров для заводов по производству газобетона”

ОГЛАВЛЕНИЕ

1. УСТАНОВКА И РАЗВЕРТЫВАНИЕ

1.1. Подготовка сервера

Шаг 1: Подключение к серверу

```
ssh root@ВАШ_IP_АДРЕС
```

Или с использованием ключа:

```
ssh -i /path/to/private_key root@ВАШ_IP_АДРЕС
```

Шаг 2: Обновление системы

```
apt-get update

apt-get upgrade -y
```

Шаг 3: Установка необходимых пакетов

```
apt-get install -y curl wget git
```

1.2. Автоматическое развертывание (рекомендуется)

Самый простой способ развертывания — использование автоматического скрипта:

```
curl -fsSL https://raw.githubusercontent.com/smartcontrolsoft-lab/gazobeton-docker3/main/deploy-server.sh | sudo bash
```

Что делает скрипт:

- Устанавливает Docker и Docker Compose

- Устанавливает Git LFS
- Клонирует репозиторий в /opt/gazobeton
- Загружает дамп базы данных (65 MB через Git LFS)
- Собирает Docker-образы
- Запускает все контейнеры
- Восстанавливает базу данных
- Устанавливает Laravel Passport

Время развертывания: 25-35 минут

1.3. Ручное развертывание

Если автоматический скрипт не подходит, выполните следующие шаги вручную:

Шаг 1: Установка Docker

```
curl -fsSL https://get.docker.com -o get-docker.sh

sh get-docker.sh

rm get-docker.sh
```

Проверка установки:

```
docker --version

docker compose version
```

Шаг 2: Установка Git LFS

Git LFS необходим для загрузки дампа базы данных (65 MB):

```
curl -s https://packagecloud.io/install/repositories/github/git-lfs/script.deb.sh | bash
apt-get install git-lfs -y

git lfs install
```

Проверка установки:

```
git lfs version
```

Шаг 3: Клонирование репозитория

```
# Создание директории для проекта

mkdir -p /opt/gazobeton

# Клонирование репозитория

git clone https://github.com/smartcontrolsoft-lab/gazobeton-docker3.git
/opt/gazobeton

# Переход в директорию проекта

cd /opt/gazobeton
```

Шаг 4: Загрузка дампа базы данных

ВАЖНО: Дамп базы данных хранится в Git LFS и должен быть загружен отдельно:

```
cd /opt/gazobeton

git lfs pull
```

Проверка наличия дампа:

```
ls -lh database/init/01-restore.sql
```

Размер файла должен быть примерно 65 MB.

Шаг 5: Настройка переменных окружения (опционально)

Если необходимо изменить настройки по умолчанию, скопируйте пример файла:

```
cp env.docker.example gazobeton-back-dev/.env
```

Отредактируйте файл .env в директории gazobeton-back-dev/:

- APP_KEY: Ключ приложения Laravel
- DB_PASSWORD: Пароль базы данных
- APP_DEBUG: Режим отладки (false для production)

Шаг 6: Сборка и запуск контейнеров

```
cd /opt/gazobeton

docker compose up -d --build
```

Что происходит:

- Собираются Docker-образы для frontend, backend и database
- Создаются Docker volumes для данных
- Запускаются контейнеры в фоновом режиме

Время сборки: 20-30 минут (зависит от скорости интернета и сервера)

Шаг 7: Ожидание запуска сервисов

Подождите 30-60 секунд, пока все сервисы запустятся:

```
# Проверка статуса контейнеров
docker compose ps
# Просмотр логов
docker compose logs -f
```

Дождитесь сообщений о готовности всех сервисов.

Шаг 8: Установка Laravel Passport

Laravel Passport необходим для аутентификации через OAuth2:

```
docker compose exec -T backend php artisan passport:install --force
```

Вы должны увидеть вывод с ключами Passport.

Шаг 9: Проверка работоспособности

Проверка контейнеров:

```
docker compose ps
```

Все контейнеры должны быть в статусе "Up".

Проверка логов:

```
# Логи backend

docker compose logs backend

# Логи frontend

docker compose logs frontend

# Логи database

docker compose logs database
```

Проверка доступности:

- Frontend: http://ВАШ_IP:3000
- Backend API: http://ВАШ_IP:8000/api/health (если есть такой endpoint)

1.4. Настройка домена и SSL (опционально)

Шаг 1: Настройка DNS

Настройте A-записи в вашем DNS-провайдере:

- gazobeton.tech → IP вашего сервера
- app.gazobeton.tech → IP вашего сервера
- www.gazobeton.tech → IP вашего сервера

Шаг 2: Установка Nginx на хост-сервере

```
apt-get install -y nginx certbot python3-certbot-nginx
```

Шаг 3: Настройка Nginx для landing page

Создайте файл /etc/nginx/sites-available/gazobeton.tech:

```
server {
    listen 80;
    server_name gazobeton.tech www.gazobeton.tech;
    root /var/www/landing;
    index index.html;
    location / {
    try_files $uri $uri/ =404;
    }
}
```

Активируйте конфигурацию:

```
ln -s /etc/nginx/sites-available/gazobeton.tech /etc/nginx/sites-enabled/
nginx -t
systemctl reload nginx
```

Шаг 4: Настройка Nginx для приложения

Создайте файл /etc/nginx/sites-available/app.gazobeton.tech:

```
server {
```

```
listen 80;

server_name app.gazobeton.tech;

 location / {

    proxy_pass http://localhost:3000;

    proxy_http_version 1.1;

    proxy_set_header Upgrade $http_upgrade;

    proxy_set_header Connection 'upgrade';

    proxy_set_header Host $host;

    proxy_cache_bypass $http_upgrade;

    proxy_read_timeout 3600;

    proxy_connect_timeout 3600;

   proxy_send_timeout 3600;

}

 location /api {

    proxy_pass http://localhost:8000;

    proxy_http_version 1.1;

    proxy_set_header Host $host;

    proxy_set_header X-Real-IP $remote_addr;

    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;

    proxy_set_header X-Forwarded-Proto $scheme;

    proxy_read_timeout 3600;

    proxy_connect_timeout 3600;
```

```
        proxy_send_timeout 3600;
    }
}
```

Активируйте конфигурацию:

```
ln -s /etc/nginx/sites-available/app.gazobeton.tech /etc/nginx/sites-enabled/
nginx -t
systemctl reload nginx
```

Шаг 5: Получение SSL-сертификата

```
# Для landing page
certbot --nginx -d gazobeton.tech -d www.gazobeton.tech
# Для приложения
certbot --nginx -d app.gazobeton.tech
```

Certbot автоматически настроит HTTPS и перенаправление с HTTP.

Шаг 6: Обновление конфигурации frontend

Обновите файл frontend/config.js в репозитории для использования правильного API хоста:

```
const URL_PREFIX = "";
 let HOST_API = undefined;
 window.getUrlPrefix = () => {
  return URL_PREFIX;
 }
  window.getHostApi = () => {
  if(!HOST_API) {
    const protocol = window.location.protocol;
    const hostname = window.location.hostname;
    HOST_API = protocol + "//" + hostname + "/";
  }
  return HOST_API;
 }
```

Пересоберите frontend:

```
cd /opt/gazobeton
docker compose up -d --build frontend
```

1.5. Первый вход в систему

После успешного развертывания:

- Откройте браузер и перейдите по адресу: - http://ВАШ_IP:3000 или - https://app.gazobeton.tech (если настроен домен)
- Зарегистрируйте первого пользователя: - Перейдите на страницу регистрации - Заполните форму регистрации - **Первый зарегистрированный пользователь автоматически получает права администратора**
- Войдите в систему используя созданные учетные данные
- Если база данных была восстановлена из дампа, используйте: - Email: test@test.ru - Пароль: 12345678

2. ЭКСПЛУАТАЦИЯ

2.1. Управление контейнерами

Просмотр статуса

```
cd /opt/gazobeton

docker compose ps
```

Просмотр логов

```
# Все сервисы

docker compose logs -f

# Конкретный сервис

docker compose logs -f backend

docker compose logs -f frontend

docker compose logs -f database
```

Перезапуск сервисов

```
# Все сервисы

docker compose restart

# Конкретный сервис

docker compose restart backend
```

Остановка сервисов

```
docker compose down
```

Остановка с удалением volumes (ОСТОРОЖНО!)

```
docker compose down -v
```

ВНИМАНИЕ: Это удалит все данные, включая базу данных!

2.2. Обновление системы

Шаг 1: Остановка контейнеров

```
cd /opt/gazobeton

docker compose down
```

Шаг 2: Обновление кода

```
git pull origin main

git lfs pull  # Если обновлялся дамп БД
```

Шаг 3: Пересборка и запуск

```
docker compose up -d --build
```

Шаг 4: Применение миграций (если есть)

```
docker compose exec backend php artisan migrate --force
```

Шаг 5: Очистка кеша

```
docker compose exec backend php artisan cache:clear

docker compose exec backend php artisan config:clear
```

```
docker compose exec backend php artisan route:clear
```

2.3. Работа с базой данных

Подключение к базе данных

```
docker compose exec database psql -U postgres -d gazobeton
```

Выполнение SQL-запросов

```
docker compose exec -T database psql -U postgres -d gazobeton -c "SELECT COUNT(*) FROM users;"
```

Создание резервной копии

```
docker compose exec -T database pg_dump -U postgres gazobeton > backup_$(date +%Y%m%d_%H%M%S).sql
```

Восстановление из резервной копии

```
docker compose exec -T database psql -U postgres -d gazobeton < backup_20240101_120000.sql
```

2.4. Импорт данных

Через веб-интерфейс

- Войдите в систему
- Перейдите в раздел "Данные"
- Нажмите "Импорт данных"
- Выберите CSV файл

- Загрузите файл

Через файловую систему

Поместите CSV файлы в директорию:

```
gazobeton-back-dev/storage/app/files/data/import/
```

Поддерживаемые файлы:

- ak.csv - данные автоклава
- apt.csv - данные автоклавной парообработки
- general.csv - общие данные
- mixreport.csv - отчеты смешивания
- otk.csv - данные ОТК
- otk_def.csv - дефекты ОТК

После размещения файлов выполните синхронизацию через веб-интерфейс.

2.5. Обучение ML-моделей

Через веб-интерфейс

- Перейдите в раздел "Аналитика"
- Выберите тип анализа
- Нажмите "Обучить модель"
- Выберите набор данных для обучения
- Дождитесь завершения обучения (10-30 минут)

Через API

```
curl -X POST http://localhost:8000/api/analysis/train \
  -H "Authorization: Bearer YOUR_TOKEN" \
  -H "Content-Type: application/json" \
  -d '{"data_id": 1, "model_type": "catb"}'
```

2.6. Мониторинг системы

Использование ресурсов

```
# Использование CPU и памяти
docker stats
# Использование диска
 df -h docker system df
```

Проверка здоровья сервисов

```
# Проверка backend
curl http://localhost:8000/api/health
# Проверка frontend
curl http://localhost:3000
# Проверка database
docker compose exec database pg_isready -U postgres
```

3. УСТРАНЕНИЕ НЕПОЛАДОК

3.1. Проблемы с запуском контейнеров

Контейнер не запускается

```
# Просмотр логов

docker compose logs [service_name]

# Проверка конфигурации

docker compose config

# Пересборка без кеша

docker compose build --no-cache

docker compose up -d
```

Ошибки с правами доступа

```
docker compose exec backend chmod -R 777 storage bootstrap/cache
```

3.2. Проблемы с базой данных

База данных не восстанавливается

```
# Проверка наличия дампа

ls -lh database/init/01-restore.sql

# Ручное восстановление

docker compose exec -T database psql -U postgres -d gazobeton < database/init/01-restore.sql
```

Ошибки миграций

```
# Сброс и повторное применение миграций

docker compose exec backend php artisan migrate:fresh --seed --force
```

ВНИМАНИЕ: Это удалит все данные!

3.3. Проблемы с производительностью

Медленная работа API

```
# Проверка логов backend

docker compose logs backend | grep -i error

# Очистка кеша

docker compose exec backend php artisan cache:clear

docker compose exec backend php artisan config:clear
```

Нехватка памяти

```
# Проверка использования памяти

docker stats

# Увеличение лимитов в docker-compose.yml

# Добавьте в секцию services:

 #   deploy:

 #     resources:

 #       limits:
```

```
#       memory: 4G
```

3.4. Проблемы с ML-серверами

ML-сервер не отвечает

```
# Проверка логов supervisor

docker compose exec backend supervisorctl status

# Перезапуск ML-сервера

docker compose exec backend supervisorctl restart [server_name]
```

Ошибки при обучении моделей

```
# Проверка логов Python

docker compose exec backend tail -f /var/www/storage/logs/laravel.log

# Проверка доступности ML-серверов

curl http://localhost:1082/prepare

curl http://localhost:1083/predict
```

3.5. Проблемы с сетью

Недоступен frontend/backend

```
# Проверка портов

netstat -tulpn | grep -E '3000|8000'

# Проверка firewall

ufw status
```

```
iptables -L

# Проверка контейнеров

docker compose ps

docker compose logs [service_name]
```

3.6. Очистка системы

Очистка неиспользуемых образов

```
docker system prune -a
```

Очистка volumes (ОСТОРОЖНО!)

```
docker volume prune
```

4. РЕЗЕРВНОЕ КОПИРОВАНИЕ И ВОССТАНОВЛЕНИЕ

4.1. Резервное копирование базы данных

Автоматическое резервное копирование

Создайте скрипт /opt/gazobeton/scripts/backup-db.sh:

```
#!/bin/bash

BACKUP_DIR="/opt/gazobeton/backups"

DATE=$(date +%Y%m%d_%H%M%S)

mkdir -p $BACKUP_DIR

docker compose exec -T database pg_dump -U postgres gazobeton | gzip >
$BACKUP_DIR/backup_$DATE.sql.gz

# Удаление старых бэкапов (старше 30 дней)

find $BACKUP_DIR -name "backup_*.sql.gz" -mtime +30 -delete
```

Сделайте скрипт исполняемым:

```
chmod +x /opt/gazobeton/scripts/backup-db.sh
```

Настройте cron для автоматического резервного копирования:

```
crontab -e
```

Добавьте строку (резервное копирование каждый день в 2:00):

```
0 2 * * * /opt/gazobeton/scripts/backup-db.sh
```

Ручное резервное копирование

```
cd /opt/gazobeton

docker compose exec -T database pg_dump -U postgres gazobeton > backup_$(date +%Y%m%d_%H%M%S).sql
```

4.2. Резервное копирование файлов

Резервное копирование storage

```
cd /opt/gazobeton

docker compose exec backend tar -czf /tmp/storage_backup.tar.gz /var/www/storage

docker cp gazobeton-backend:/tmp/storage_backup.tar.gz ./backups/storage_$(date +%Y%m%d_%H%M%S).tar.gz
```

Резервное копирование ML-моделей

```
cd /opt/gazobeton

docker compose exec backend tar -czf /tmp/models_backup.tar.gz /var/www/storage/app/files/analysis/model

docker cp gazobeton-backend:/tmp/models_backup.tar.gz ./backups/models_$(date +%Y%m%d_%H%M%S).tar.gz
```

4.3. Восстановление из резервной копии

Восстановление базы данных

```
# Остановка контейнеров (опционально)

docker compose stop backend

# Восстановление
```

```
docker compose exec -T database psql -U postgres -d gazobeton < backup_20240101_120000.sql

# Или для сжатого файла

gunzip < backup_20240101_120000.sql.gz | docker compose exec -T database psql -U postgres -d gazobeton

# Перезапуск контейнеров

docker compose start backend
```

Восстановление файлов

```
# Восстановление storage

docker cp ./backups/storage_20240101_120000.tar.gz gazobeton-backend:/tmp/

docker compose exec backend tar -xzf /tmp/storage_20240101_120000.tar.gz -C /

# Восстановление моделей

docker cp ./backups/models_20240101_120000.tar.gz gazobeton-backend:/tmp/

docker compose exec backend tar -xzf /tmp/models_20240101_120000.tar.gz -C /
```

4.4. Полное резервное копирование системы

Создайте скрипт для полного резервного копирования:

```
#!/bin/bash

BACKUP_DIR="/opt/gazobeton/backups/full"

DATE=$(date +%Y%m%d_%H%M%S)

mkdir -p $BACKUP_DIR
```

```
# Резервное копирование БД

docker compose exec -T database pg_dump -U postgres gazobeton | gzip >
$BACKUP_DIR/db_$DATE.sql.gz

# Резервное копирование storage

docker compose exec backend tar -czf /tmp/storage.tar.gz /var/www/storage

docker cp gazobeton-backend:/tmp/storage.tar.gz $BACKUP_DIR/storage_$DATE.tar.gz

# Резервное копирование моделей

docker compose exec backend tar -czf /tmp/models.tar.gz
/var/www/storage/app/files/analysis/model

docker cp gazobeton-backend:/tmp/models.tar.gz $BACKUP_DIR/models_$DATE.tar.gz

# Резервное копирование конфигурации

tar -czf $BACKUP_DIR/config_$DATE.tar.gz docker-compose.yml env.docker.example

echo "Резервное копирование завершено: $BACKUP_DIR"
```

ПРИЛОЖЕНИЯ

Приложение А: Структура проекта

```
/opt/gazobeton/
├── backend/                # Конфигурация backend
│   ├── Dockerfile
│   ├── nginx.conf
│   └── supervisord.conf
├── frontend/               # Конфигурация frontend
│   ├── Dockerfile
│   ├── nginx.conf
│   └── config.js
├── database/              # Конфигурация БД
│   └── init/
│       └── 01-restore.sql
├── gazobeton-back-dev/    # Исходный код backend
├── gazobeton-front-dev/   # Исходный код frontend
├── docker-compose.yml     # Конфигурация Docker Compose
├── deploy-server.sh      # Скрипт автоматического развертывания
└── scripts/               # Вспомогательные скрипты
    ├── backup-db.sh
    └── restore-db.sh
```

Приложение В: Полезные команды

```
# Просмотр использования ресурсов

docker stats

# Просмотр логов в реальном времени

docker compose logs -f [service]

# Выполнение команд в контейнере

docker compose exec [service] [command]

# Пересборка конкретного сервиса

docker compose up -d --build [service]

# Очистка неиспользуемых ресурсов

docker system prune -a
```

```
# Просмотр volumes

docker volume ls  # Просмотр сетей docker network ls
```

Приложение C: Контакты поддержки

- **Email**: rezanovdv@mail.ru
- **GitHub**: https://github.com/smartcontrolsoft-lab/gazobeton-docker3
- **Документация**: См. README.md в репозитории