Big Data Dashboard - Kendi Sunucuya Deployment Rehberi

Gereksinimler

Sunucu Gereksinimleri

- **OS**: Ubuntu 20.04+ / CentOS 8+ / Debian 11+
- **RAM**: Minimum 2GB (Önerilen 4GB+)
- **CPU**: 2 core+
- **Storage**: Minimum 20GB SSD
- **Network**: 1Gbps bağlantı

Yazılım Gereksinimleri

- Node.js 18.x veya üzeri
- PostgreSQL 13+ veya MySQL 8+
- Nginx (reverse proxy için)
- PM2 (process manager)
- Git

Adım 1: Sunucu Kurulumu

Ubuntu/Debian için:

```
# Sistem güncelleme
sudo apt update && sudo apt upgrade -y

# Node.js kurulumu
curl -fsSL https://deb.nodesource.com/setup_18.x | sudo -E bash -
sudo apt-get install -y nodejs

# Yarn kurulumu
npm install -g yarn

# PostgreSQL kurulumu
sudo apt install postgresql postgresql-contrib -y

# Nginx kurulumu
sudo apt install nginx -y

# PM2 kurulumu
npm install -g pm2

# Git kurulumu
sudo apt install git -y
```

CentOS/RHEL için:

```
# Node.js kurulumu
curl -fsSL https://rpm.nodesource.com/setup_18.x | sudo bash -
sudo dnf install nodejs -y

# Yarn kurulumu
npm install -g yarn

# PostgreSQL kurulumu
sudo dnf install postgresql postgresql-server -y
sudo postgresql-setup initdb
sudo systemctl enable postgresql --now

# Nginx kurulumu
sudo dnf install nginx -y

# PM2 kurulumu
npm install -g pm2
```

Adım 2: Proje Dosyalarını Sunucuya Transfer Etme

Yöntem 1: Git ile

```
# Sunucuda proje dizini oluştur
cd /var/www/
sudo mkdir big-data-dashboard
sudo chown $USER:$USER big-data-dashboard

# Projeyi klonla (eğer GitHub'da ise)
git clone https://github.com/kullaniciadi/big-data-dashboard.git
cd big-data-dashboard/app
```

Yöntem 2: SCP ile (Mevcut projeden)

```
# Local makinenizden sunucunuza kopyala
scp -r /home/ubuntu/big_data_dashboard/ user@your-server-ip:/var/www/
```

Yöntem 3: SFTP/FTP ile

FileZilla, WinSCP gibi araçlarla proje klasörünü sunucunuza yükleyin.

Adım 3: PostgreSQL Veritabanı Kurulumu

```
# PostgreSQL'e bağlan
sudo -u postgres psql

# Veritabanı ve kullanıcı oluştur
CREATE DATABASE big_data_dashboard;
CREATE USER dashboard_user WITH ENCRYPTED PASSWORD 'güçlü_şifre_123';
GRANT ALL PRIVILEGES ON DATABASE big_data_dashboard TO dashboard_user;
ALTER USER dashboard_user CREATEDB;
\q
```

Adım 4: Environment Variables (.env) Konfigürasyonu

```
cd /var/www/big-data-dashboard/app

# .env dosyasını oluştur
cp .env.example .env  # Eğer varsa
# veya
nano .env
```

.env dosyası içeriği:

```
# Database
DATABASE_URL="postgresql://dashboard_user:güçlü_şifre_123@localhost:5432/
big_data_dashboard"

# NextAuth
NEXTAUTH_URL="https://yourdomain.com"
NEXTAUTH_SECRET="generate-random-32-char-string-here"

# AWS S3 (Kendi S3 bucket'ınız için)
AWS_PROFILE=default
AWS_REGION=eu-west-1  # Size yakın region seçin
AWS_BUCKET_NAME=your-bucket-name
AWS_FOLDER_PREFIX=uploads/
AWS_ACCESS_KEY_ID=your-access-key
AWS_SECRET_ACCESS_KEY=your-secret-key

# LLM API
ABACUSAI_API_KEY="your-api-key-here"

# Production
NODE_ENV=production
```

Adım 5: AWS S3 Konfigürasyonu

Yöntem 1: AWS S3 Bucket oluşturma

1. AWS Console’a giriş yapın
2. S3 servisine gidin
3. Yeni bucket oluşturun
4. Public access’i kapatın
5. IAM user oluşturun ve S3 yetkileri verin

Yöntem 2: Alternatif Object Storage (MinIO, DigitalOcean Spaces)

```
# MinIO self-hosted kurulumu
wget https://dl.min.io/server/minio/release/linux-amd64/minio
chmod +x minio
sudo mv minio /usr/local/bin/

# MinIO başlatma
mkdir ~/minio-data
minio server ~/minio-data --address ":9000" --console-address ":9001"
```

Adım 6: Bağımlılıkları Yükleme ve Build

```
cd /var/www/big-data-dashboard/app

# Bağımlılıkları yükle
yarn install

# Prisma migration
npx prisma generate
npx prisma db push  # veya npx prisma migrate deploy

# Next.js build
yarn build

# Build sonucu kontrol et
ls -la .next/
```

Adım 7: PM2 ile Process Management

```
# PM2 ecosystem dosyası oluştur
nano ecosystem.config.js
```

ecosystem.config.js içeriği:

```
module.exports = {
  apps: [{
    name: 'big-data-dashboard',
    script: './node_modules/next/dist/bin/next',
    args: 'start',
    cwd: '/var/www/big-data-dashboard/app',
    instances: 'max',
    exec_mode: 'cluster',
    env: {
      NODE_ENV: 'production',
      PORT: 3000
    },
    log_file: '/var/log/pm2/big-data-dashboard.log',
    error_file: '/var/log/pm2/big-data-dashboard-error.log',
    out_file: '/var/log/pm2/big-data-dashboard-out.log',
    time: true
  }]
};
```

```
# PM2 ile başlat
pm2 start ecosystem.config.js

# Otomatik başlatma için
pm2 startup
pm2 save

# Status kontrol
pm2 status
pm2 logs big-data-dashboard
```

Adım 8: Nginx Reverse Proxy Konfigürasyonu

```
# Nginx site konfigürasyonu
sudo nano /etc/nginx/sites-available/big-data-dashboard
```

Nginx konfigürasyon dosyası:

```
server {
    listen 80;
    server_name yourdomain.com www.yourdomain.com;

    # HTTP'den HTTPS'e yönlendirme
    return 301 https://$server_name$request_uri;
}

server {
    listen 443 ssl http2;
    server_name yourdomain.com www.yourdomain.com;

    # SSL Sertifikaları (Let's Encrypt ile)
    ssl_certificate /etc/letsencrypt/live/yourdomain.com/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/yourdomain.com/privkey.pem;

    # SSL Ayarları
    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_ciphers HIGH:!aNULL:!MD5;
    ssl_prefer_server_ciphers on;

    # Güvenlik Headers
    add_header X-Frame-Options SAMEORIGIN;
    add_header X-Content-Type-Options nosniff;
    add_header X-XSS-Protection "1; mode=block";

    # Gzip Compression
    gzip on;
    gzip_types text/plain application/xml text/css text/js application/javascript;

    # Client Max Body Size (file upload için)
    client_max_body_size 100M;

    location / {
        proxy_pass http://127.0.0.1:3000;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        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_cache_bypass $http_upgrade;
        proxy_read_timeout 300;
        proxy_connect_timeout 300;
        proxy_send_timeout 300;
    }

    # Static files caching
    location /_next/static/ {
        proxy_pass http://127.0.0.1:3000;
        add_header Cache-Control "public, max-age=31536000, immutable";
    }
}
```

```
# Site'ı aktif et
sudo ln -s /etc/nginx/sites-available/big-data-dashboard /etc/nginx/sites-enabled/

# Nginx konfigürasyon test
sudo nginx -t

# Nginx restart
sudo systemctl restart nginx
sudo systemctl enable nginx
```

Adım 9: SSL Sertifikası (Let’s Encrypt)

```
# Certbot kurulum
sudo apt install certbot python3-certbot-nginx -y

# SSL sertifikası al
sudo certbot --nginx -d yourdomain.com -d www.yourdomain.com

# Otomatik yenileme test
sudo certbot renew --dry-run
```

Adım 10: Firewall Konfigürasyonu

```
# UFW firewall
sudo ufw enable
sudo ufw allow ssh
sudo ufw allow 'Nginx Full'
sudo ufw allow 5432  # PostgreSQL (sadece local erişim için)
sudo ufw status
```

Adım 11: Monitoring ve Logging

```
# Log rotasyonu için logrotate
sudo nano /etc/logrotate.d/big-data-dashboard

# İçerik:
/var/log/pm2/*.log {
    daily
    missingok
    rotate 14
    compress
    notifempty
    create 0644 ubuntu ubuntu
    postrotate
        pm2 reload big-data-dashboard
    endscript
}

# Sistem monitoring
sudo apt install htop netstat-nat -y
```

Adım 12: Backup Stratejisi

```
# Veritabanı backup script
nano ~/backup-db.sh
```

Backup script:

```
#!/bin/bash
DATE=$(date +%Y%m%d_%H%M%S)
BACKUP_DIR="/home/ubuntu/backups"
DB_NAME="big_data_dashboard"

mkdir -p $BACKUP_DIR

# PostgreSQL backup
pg_dump -U dashboard_user -h localhost $DB_NAME > $BACKUP_DIR/db_backup_$DATE.sql

# Proje dosyaları backup
tar -czf $BACKUP_DIR/app_backup_$DATE.tar.gz /var/www/big-data-dashboard

# Eski backup'ları sil (30 günden eski)
find $BACKUP_DIR -name "*.sql" -mtime +30 -delete
find $BACKUP_DIR -name "*.tar.gz" -mtime +30 -delete

echo "Backup completed: $DATE"
```

```
# Script'i çalıştırılabilir yap
chmod +x ~/backup-db.sh

# Cron job ekle (günlük backup)
crontab -e

# Cron içeriği:
0 2 * * * /home/ubuntu/backup-db.sh >> /var/log/backup.log 2>&1
```

Adım 13: Domain ve DNS Ayarları

1. **Domain registrar'da**:
   - A record: `yourdomain.com` → `sunucu-ip-adresi`
   - A record: `www.yourdomain.com` → `sunucu-ip-adresi`

2. **DNS propagation kontrolü**:
   `bash`
   `nslookup yourdomain.com`

Troubleshooting

Yaygın Problemler ve Çözümleri

1. Build Error

```
# Node modules temizle
rm -rf node_modules package-lock.json yarn.lock
yarn install
yarn build
```

2. Database Connection Error

```
# PostgreSQL servis kontrol
sudo systemctl status postgresql
sudo systemctl restart postgresql

# Bağlantı test
psql -U dashboard_user -d big_data_dashboard -h localhost
```

3. PM2 Process Error

```
# PM2 restart
pm2 restart big-data-dashboard
pm2 logs big-data-dashboard --lines 100
```

4. Nginx Error

```
# Nginx logs
sudo tail -f /var/log/nginx/error.log
sudo nginx -t
sudo systemctl restart nginx
```

5. SSL Certificate Issues

```
# Sertifika yenile
sudo certbot renew
sudo systemctl restart nginx
```

Performance Optimizasyon

1. Database Optimizasyon

```
-- Index'leri kontrol et
SELECT * FROM pg_indexes WHERE tablename = 'your_table';

-- Slow query'leri logla
ALTER SYSTEM SET log_min_duration_statement = 1000;
SELECT pg_reload_conf();
```

2. Caching

```
# Redis kurulumu (opsiyonel)
sudo apt install redis-server -y
sudo systemctl enable redis-server
```

3. CDN Integration

- Cloudflare, AWS CloudFront gibi CDN'leri entegre edin
- Static asset'ları CDN'den servis edin

Güvenlik Önlemleri

1. **Fail2Ban kurulumu**:

```
sudo apt install fail2ban -y
sudo systemctl enable fail2ban
```

1. **SSH Key Authentication**:

```
# Password authentication'ı kapat
sudo nano /etc/ssh/sshd_config
# PasswordAuthentication no
sudo systemctl restart ssh
```

1. **Regular Updates**:

```
# Otomatik güvenlik güncellemeleri
sudo apt install unattended-upgrades -y
sudo dpkg-reconfigure unattended-upgrades
```

Maintenance

Günlük Kontroller

```
# System status
pm2 status
sudo systemctl status nginx postgresql
df -h  # disk usage
free -m  # memory usage
```

Weekly Tasks

```
# System update
sudo apt update && sudo apt upgrade -y

# Log cleanup
sudo journalctl --vacuum-time=30d

# SSL certificate check
sudo certbot certificates
```

Bu rehberi takip ederek projenizi kendi sunucunuza başarıyla deploy edebilirsiniz. Herhangi bir sorun yaşarsanız, hata loglarını kontrol ederek problemi teşhis edebilirsiniz.