🖥️ Guia de Configuração para Servidor

Este guia detalha como configurar a Automação de Prospecção em um servidor VPS para uso em produção.

🎯 Visão Geral

Arquitetura Recomendada

```
Internet → Nginx (Proxy Reverso) → Gunicorn → Flask App
                ↓
            Supervisor (Gerenciamento de Processo)
                ↓
            Systemd (Serviço do Sistema)
```

Especificações Mínimas do Servidor

- **CPU**: 2 vCPUs
- **RAM**: 4GB
- **Storage**: 20GB SSD
- **OS**: Ubuntu 20.04+ LTS
- **Bandwidth**: 100Mbps

🚀 Configuração Inicial do Servidor

1. Preparação do Servidor

```
# Conectar via SSH
ssh root@SEU_IP_DO_SERVIDOR

# Atualizar sistema
apt update && apt upgrade -y

# Instalar dependências essenciais
apt install -y \
    python3.11 \
    python3.11-venv \
    python3-pip \
    git \
    nginx \
    supervisor \
    ufw \
    htop \
    curl \
    wget \
    unzip
```

2. Configurar Firewall

```
# Configurar UFW
ufw default deny incoming
ufw default allow outgoing
ufw allow ssh
ufw allow 'Nginx Full'
ufw --force enable

# Verificar status
ufw status
```

3. Criar Usuário para Aplicação

```
# Criar usuário dedicado
useradd -m -s /bin/bash prospeccao
usermod -aG sudo prospeccao

# Configurar SSH para o usuário (opcional)
mkdir -p /home/prospeccao/.ssh
cp ~/.ssh/authorized_keys /home/prospeccao/.ssh/
chown -R prospeccao:prospeccao /home/prospeccao/.ssh
chmod 700 /home/prospeccao/.ssh
chmod 600 /home/prospeccao/.ssh/authorized_keys
```

📦 Instalação da Aplicação

1. Instalar Google Chrome

```
# Adicionar repositório do Google Chrome
wget -q -O - https://dl.google.com/linux/linux_signing_key.pub | apt-key add -
echo "deb [arch=amd64] http://dl.google.com/linux/chrome/deb/ stable main" > /etc/apt/
sources.list.d/google-chrome.list
apt update
apt install -y google-chrome-stable

# Verificar instalação
google-chrome --version
```

2. Configurar Aplicação

```
# Mudar para usuário prospeccao
su - prospeccao

# Clonar repositório
git clone https://github.com/seu-usuario/automacao-prospeccao.git
cd automacao-prospeccao

# Criar ambiente virtual
python3.11 -m venv venv
source venv/bin/activate

# Instalar dependências
pip install --upgrade pip
pip install -r requirements.txt

# Configurar ambiente
cp .env.example .env
```

3. Configurar Variáveis de Ambiente para Produção

```
nano .env
```

```
# Configurações de Produção
FLASK_ENV=production
FLASK_DEBUG=false
FLASK_HOST=127.0.0.1
FLASK_PORT=5000

# Chave secreta (GERAR UMA NOVA!)
SECRET_KEY=sua_chave_super_secreta_de_32_caracteres_aqui

# Banco de dados
DATABASE_URL=sqlite:///data/prospeccao.db

# Selenium - Configurações para servidor
SELENIUM_HEADLESS=true
CHROME_DRIVER_PATH=/usr/bin/chromedriver

# Rate limiting para produção
MAX_MESSAGES_PER_HOUR=5
MAX_SCRAPING_RESULTS=50

# Logging
LOG_LEVEL=INFO
LOG_FILE=logs/app.log

# Configurações específicas do servidor
USER_AGENT=Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrom
e/91.0.4472.124 Safari/537.36
```

4. Testar Aplicação

```
# Criar diretórios necessários
mkdir -p data logs export

# Testar aplicação
python app.py

# Em outro terminal, testar
curl http://127.0.0.1:5000/api/status
```

⚙️ Configuração do Gunicorn

1. Criar Arquivo de Configuração do Gunicorn

```
nano gunicorn.conf.py
```

```
# gunicorn.conf.py
import multiprocessing

# Configurações do servidor
bind = "127.0.0.1:5000"
workers = multiprocessing.cpu_count() * 2 + 1
worker_class = "sync"
worker_connections = 1000
timeout = 120
keepalive = 2

# Configurações de logging
accesslog = "logs/gunicorn_access.log"
errorlog = "logs/gunicorn_error.log"
loglevel = "info"

# Configurações de processo
daemon = False
pidfile = "gunicorn.pid"
user = "prospeccao"
group = "prospeccao"

# Configurações de performance
max_requests = 1000
max_requests_jitter = 100
preload_app = True

# Configurações de segurança
limit_request_line = 4094
limit_request_fields = 100
limit_request_field_size = 8190
```

2. Testar Gunicorn

```
# Ativar ambiente virtual
source venv/bin/activate

# Testar Gunicorn
gunicorn -c gunicorn.conf.py app:app

# Testar em outro terminal
curl http://127.0.0.1:5000
```

🔧 Configuração do Systemd

1. Criar Arquivo de Serviço

```
# Como root
sudo nano /etc/systemd/system/automacao-prospeccao.service
```

```
[Unit]
Description=Automação de Prospecção - Curitiba
After=network.target

[Service]
Type=simple
User=prospeccao
Group=prospeccao
WorkingDirectory=/home/prospeccao/automacao-prospeccao
Environment=PATH=/home/prospeccao/automacao-prospeccao/venv/bin
Environment=FLASK_ENV=production
ExecStart=/home/prospeccao/automacao-prospeccao/venv/bin/gunicorn -c gunicorn.conf.py
app:app
ExecReload=/bin/kill -s HUP $MAINPID
Restart=always
RestartSec=10
StandardOutput=journal
StandardError=journal

[Install]
WantedBy=multi-user.target
```

2. Habilitar e Iniciar Serviço

```
# Recarregar systemd
sudo systemctl daemon-reload

# Habilitar serviço
sudo systemctl enable automacao-prospeccao

# Iniciar serviço
sudo systemctl start automacao-prospeccao

# Verificar status
sudo systemctl status automacao-prospeccao

# Ver logs
sudo journalctl -u automacao-prospeccao -f
```

🌐 Configuração do Nginx

1. Criar Configuração do Site

```
sudo nano /etc/nginx/sites-available/automacao-prospeccao
```

```
server {
    listen 80;
    server_name seu-dominio.com www.seu-dominio.com;

    # Logs
    access_log /var/log/nginx/automacao_access.log;
    error_log /var/log/nginx/automacao_error.log;

    # Configurações de segurança
    add_header X-Frame-Options "SAMEORIGIN" always;
    add_header X-XSS-Protection "1; mode=block" always;
    add_header X-Content-Type-Options "nosniff" always;
    add_header Referrer-Policy "no-referrer-when-downgrade" always;
    add_header Content-Security-Policy "default-src 'self' http: https: data: blob: 'un
safe-inline'" always;

    # Configurações de upload
    client_max_body_size 100M;

    # Proxy para aplicação Flask
    location / {
        proxy_pass http://127.0.0.1:5000;
        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_redirect off;

        # Timeouts
        proxy_connect_timeout 60s;
        proxy_send_timeout 60s;
        proxy_read_timeout 60s;
    }

    # Servir arquivos estáticos diretamente
    location /static {
        alias /home/prospeccao/automacao-prospeccao/static;
        expires 1y;
        add_header Cache-Control "public, immutable";
    }

    # Servir arquivos de export
    location /export {
        alias /home/prospeccao/automacao-prospeccao/export;
        internal;
    }

    # Bloquear acesso a arquivos sensíveis
    location ~ /\. {
        deny all;
    }

    location ~ \.(env|log|db)$ {
        deny all;
    }
}
```

2. Habilitar Site

```
# Criar link simbólico
sudo ln -s /etc/nginx/sites-available/automacao-prospeccao /etc/nginx/sites-enabled/

# Remover site padrão
sudo rm /etc/nginx/sites-enabled/default

# Testar configuração
sudo nginx -t

# Reiniciar Nginx
sudo systemctl restart nginx
```

🔒 Configuração SSL (HTTPS)

1. Instalar Certbot

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

2. Obter Certificado SSL

```
# Obter certificado (substitua seu-dominio.com)
sudo certbot --nginx -d seu-dominio.com -d www.seu-dominio.com

# Testar renovação automática
sudo certbot renew --dry-run
```

3. Configurar Renovação Automática

```
# Adicionar ao crontab
sudo crontab -e

# Adicionar linha para renovação automática
0 12 * * * /usr/bin/certbot renew --quiet
```

📊 Monitoramento e Logs

1. Configurar Logrotate

```
sudo nano /etc/logrotate.d/automacao-prospeccao
```

```
/home/prospeccao/automacao-prospeccao/logs/*.log {
    daily
    missingok
    rotate 52
    compress
    delaycompress
    notifempty
    create 644 prospeccao prospeccao
    postrotate
        systemctl reload automacao-prospeccao
    endscript
}
```

2. Script de Monitoramento

```
nano /home/prospeccao/monitor.sh
```

```
#!/bin/bash

# Script de monitoramento da aplicação
LOG_FILE="/home/prospeccao/automacao-prospeccao/logs/monitor.log"
DATE=$(date '+%Y-%m-%d %H:%M:%S')

# Verificar se aplicação está rodando
if ! curl -f http://127.0.0.1:5000/api/status > /dev/null 2>&1; then
    echo "$DATE - ERRO: Aplicação não está respondendo" >> $LOG_FILE

    # Tentar reiniciar serviço
    sudo systemctl restart automacao-prospeccao

    # Aguardar 30 segundos e testar novamente
    sleep 30

    if curl -f http://127.0.0.1:5000/api/status > /dev/null 2>&1; then
        echo "$DATE - INFO: Aplicação reiniciada com sucesso" >> $LOG_FILE
    else
        echo "$DATE - CRÍTICO: Falha ao reiniciar aplicação" >> $LOG_FILE
        # Aqui você pode adicionar notificação por email/Slack
    fi
else
    echo "$DATE - OK: Aplicação funcionando normalmente" >> $LOG_FILE
fi

# Verificar uso de disco
DISK_USAGE=$(df /home/prospeccao | awk 'NR==2 {print $5}' | sed 's/%//')
if [ $DISK_USAGE -gt 80 ]; then
    echo "$DATE - AVISO: Uso de disco alto: ${DISK_USAGE}%" >> $LOG_FILE
fi

# Verificar uso de memória
MEM_USAGE=$(free | awk 'NR==2{printf "%.0f", $3*100/$2}')
if [ $MEM_USAGE -gt 80 ]; then
    echo "$DATE - AVISO: Uso de memória alto: ${MEM_USAGE}%" >> $LOG_FILE
fi
```

```
# Tornar executável
chmod +x /home/prospeccao/monitor.sh

# Adicionar ao crontab (verificar a cada 5 minutos)
crontab -e
# Adicionar:
*/5 * * * * /home/prospeccao/monitor.sh
```

🔄 Backup e Recuperação

1. Script de Backup

```
nano /home/prospeccao/backup.sh
```

```
#!/bin/bash

# Configurações
BACKUP_DIR="/home/prospeccao/backups"
DATE=$(date +%Y%m%d_%H%M%S)
APP_DIR="/home/prospeccao/automacao-prospeccao"

# Criar diretório de backup
mkdir -p $BACKUP_DIR

# Backup do banco de dados
echo "Fazendo backup do banco de dados..."
cp $APP_DIR/data/prospeccao.db $BACKUP_DIR/prospeccao_$DATE.db

# Backup dos arquivos de configuração
echo "Fazendo backup das configurações..."
tar -czf $BACKUP_DIR/config_$DATE.tar.gz $APP_DIR/.env $APP_DIR/gunicorn.conf.py

# Backup dos logs recentes (últimos 7 dias)
echo "Fazendo backup dos logs..."
find $APP_DIR/logs/ -name "*.log" -mtime -7 -exec cp {} $BACKUP_DIR/ \;

# Backup dos exports
echo "Fazendo backup dos exports..."
if [ -d "$APP_DIR/export" ] && [ "$(ls -A $APP_DIR/export)" ]; then
    tar -czf $BACKUP_DIR/exports_$DATE.tar.gz $APP_DIR/export/
fi

# Limpar backups antigos (manter últimos 30 dias)
echo "Limpando backups antigos..."
find $BACKUP_DIR -name "*.db" -mtime +30 -delete
find $BACKUP_DIR -name "*.tar.gz" -mtime +30 -delete

# Sincronizar com armazenamento remoto (opcional)
# rsync -av $BACKUP_DIR/ user@backup-server:/backups/automacao-prospeccao/

echo "Backup concluído: $DATE"
```

2. Configurar Backup Automático

```
# Tornar executável
chmod +x /home/prospeccao/backup.sh

# Adicionar ao crontab (backup diário às 2h)
crontab -e
# Adicionar:
0 2 * * * /home/prospeccao/backup.sh
```

🔧 Manutenção e Atualizações

1. Script de Atualização

```
nano /home/prospeccao/update.sh
```

```
#!/bin/bash

# Script de atualização da aplicação
APP_DIR="/home/prospeccao/automacao-prospeccao"
cd $APP_DIR

echo "Iniciando atualização..."

# Fazer backup antes da atualização
./backup.sh

# Parar serviço
sudo systemctl stop automacao-prospeccao

# Atualizar código
git pull origin main

# Ativar ambiente virtual
source venv/bin/activate

# Atualizar dependências
pip install --upgrade -r requirements.txt

# Executar migrações (se houver)
# python migrate.py

# Reiniciar serviço
sudo systemctl start automacao-prospeccao

# Verificar se está funcionando
sleep 10
if curl -f http://127.0.0.1:5000/api/status > /dev/null 2>&1; then
    echo "Atualização concluída com sucesso!"
else
    echo "ERRO: Aplicação não está respondendo após atualização"
    exit 1
fi
```

2. Comandos Úteis de Manutenção

```
# Ver status dos serviços
sudo systemctl status automacao-prospeccao nginx

# Ver logs em tempo real
sudo journalctl -u automacao-prospeccao -f

# Reiniciar aplicação
sudo systemctl restart automacao-prospeccao

# Ver uso de recursos
htop

# Ver conexões de rede
sudo netstat -tulpn | grep :5000

# Verificar espaço em disco
df -h

# Ver logs do Nginx
sudo tail -f /var/log/nginx/automacao_access.log
```

🚨 Troubleshooting

Problemas Comuns

Aplicação não inicia

```
# Verificar logs
sudo journalctl -u automacao-prospeccao -n 50

# Verificar configuração
sudo systemctl status automacao-prospeccao

# Testar manualmente
su - prospeccao
cd automacao-prospeccao
source venv/bin/activate
python app.py
```

Nginx retorna 502 Bad Gateway

```
# Verificar se aplicação está rodando
curl http://127.0.0.1:5000

# Verificar logs do Nginx
sudo tail -f /var/log/nginx/error.log

# Verificar configuração do Nginx
sudo nginx -t
```

Chrome não funciona em modo headless

```
# Instalar dependências adicionais
sudo apt install -y xvfb

# Ou usar Xvfb
export DISPLAY=:99
Xvfb :99 -screen 0 1024x768x24 > /dev/null 2>&1 &
```

📈 Otimização de Performance

1. Configurações do Sistema

```
# Aumentar limites de arquivo
echo "prospeccao soft nofile 65536" >> /etc/security/limits.conf
echo "prospeccao hard nofile 65536" >> /etc/security/limits.conf

# Otimizar configurações de rede
echo "net.core.somaxconn = 65536" >> /etc/sysctl.conf
echo "net.ipv4.tcp_max_syn_backlog = 65536" >> /etc/sysctl.conf
sysctl -p
```

2. Configurações do Nginx

```
# Adicionar ao bloco http em /etc/nginx/nginx.conf
worker_processes auto;
worker_connections 1024;
keepalive_timeout 65;
gzip on;
gzip_vary on;
gzip_min_length 1024;
gzip_types text/plain text/css application/json application/javascript text/xml applic-
ation/xml application/xml+rss text/javascript;
```

Servidor configurado com sucesso! 🎉

Sua aplicação agora está rodando em produção com:
- ✅ Nginx como proxy reverso
- ✅ Gunicorn como servidor WSGI
- ✅ Systemd para gerenciamento de serviço
- ✅ SSL/HTTPS configurado
- ✅ Monitoramento e backup automático
- ✅ Logs estruturados

Próximos passos: Guia de Uso (USAGE.md) | Integração com Code LLM (CODE_LLM.md)