Tutorial de Instalação – Solar

Tutorial de Instalação – Solar

1. Dockerfile personalizado

Para garantir o funcionamento correto da aplicação, utilize o `Dockerfile` abaixo. Ele realiza a instalação de bibliotecas necessárias e corrige erros comuns relacionados a dependências de sistema e pacotes Python.

Substitua o conteúdo do `Dockerfile` localizado na raiz do projeto por este:

```
FROM python:3.9-slim

# Exibe as mensagens de saída sem buffer
ENV PYTHONUNBUFFERED=1

# Argumentos de build
ARG POETRY_ARGS
ARG SOLAR_INSTALL_MSODBC_DRIVERS

# Instala dependências do sistema
RUN apt-get update && apt-get install -y \
    build-essential \
    cron \
    curl \
    ffmpeg \
    git-core \
    libfontconfig1 \
    libfreetype6-dev \
    libldap2-dev \
    libmagic1 \
    libpq-dev \
    libsasl2-dev \
    libssl-dev \
    libxrender1 \
    unixodbc \
```

```
    python3-dev && \
    rm -rf /var/lib/apt/lists/*

# Instala drivers do Microsoft SQL Server, se solicitado
RUN if [ ! -z "$SOLAR_INSTALL_MSODBC_DRIVERS" ]; then \
    curl https://packages.microsoft.com/keys/microsoft.asc | tee
/etc/apt/trusted.gpg.d/microsoft.asc && \
    echo "deb [arch=amd64,arm64,armhf]
https://packages.microsoft.com/debian/12/prod bookworm main" | tee
/etc/apt/sources.list.d/mssql-release.list && \
    apt-get update && \
    ACCEPT_EULA=Y apt-get install -y msodbcsql18 mssql-tools18 unixodbc-dev
&& \
    echo 'export PATH="$PATH:/opt/mssql-tools18/bin"' >> ~/.bashrc; \
    fi

# Configuração do ambiente Python
COPY --chown=1000:1000 pyproject.toml poetry.lock /app/src/
WORKDIR /app/src/

RUN pip install --upgrade pip && \
    pip install --force-reinstall virtualenv && \
    pip install poetry && \
    poetry config virtualenvs.create false && \
    poetry install ${POETRY_ARGS:-} && \
    rm -rf /app/src/pyproject.toml /app/src/poetry.lock

# Copia o código-fonte da aplicação
COPY . /app/src/

# Prepara diretórios e arquivos estáticos
RUN mkdir -p /app/src/staticfiles_producao/ /app/src/log/ && \
    python manage.py collectstatic_js_reverse && \
    python manage.py collectstatic --clear --noinput && \
    python manage.py compress --force

# Configura o cron para limpeza periódica da pasta /tmp
COPY crontab_scripts/cleanup-tmp /etc/cron.d/cleanup-tmp
RUN chmod 0644 /etc/cron.d/cleanup-tmp && \
    chmod u+x "/app/src/start-cron.sh"

# Define o entrypoint e porta exposta
ENTRYPOINT ["sh", "/app/src/start-cron.sh"]
EXPOSE 1024
```

```
Comando padrão do container
CMD ["uwsgi", "--ini", "/app/src/uwsgi/solar.ini"]
```

2. Arquivo `.env`

Crie um arquivo chamado `.env` na raiz do projeto e adicione o conteúdo abaixo. Ele define todas as variáveis de ambiente necessárias para executar o projeto localmente com a base de testes do Solar:

```
# Configurações de Debug e Logs
DEBUG=True
DEBUG_TOOLBAR=False
DEBUG_TEMPLATE=False
DEBUG_VSCODE=False

SENTRY_DSN=
SENTRY_TRACES_SAMPLE_RATE=0.1

# Analytics
GOOGLE_ANALYTICS_ID=
GOOGLE_ANALYTICS_4_ID=

# Acesso
SECRET_KEY=WD8saoCWCsjy4IQS3m7DGGNGn2nLKVsv
ALLOWED_HOSTS=*
TIME_ZONE=America/Sao_Paulo

# Sessão
SESSION_COOKIE_AGE=86400
SESSION_SAVE_EVERY_REQUEST=False

# Banco de Dados
DATABASE_URL=postgres://postgres:postgres@172.88.0.95:5432/db_solar
DATABASE_CONN_MAX_AGE=60
DATABASE_APPLICATION_ID=
DATABASE_APPLICATION_NAME=

MEMCACHED_DATABASE_URL=memcached:11211
REDIS_DATABASE_URL=redis://redis:6379
CACHEOPS_ENABLED=True
```

```
# APIs
CHRONUS_PODE_GERAR_XLSX_SEM_PAGINACAO=False
CHRONUS_URL=

PROCAPI_URL=http://172.88.0.95:8001/
PROCAPI_TOKEN=e406d9344a85bf229fc01e1ce4d6603c5432b8f8

LIVRE_API_URL=
LIVRE_API_TOKEN=
ATHENAS_API_URL=
PLANTAO_API_URL=

# LDAP
LDAP_AUTH_BIND_DN=CN=ldap.sitic@defensoria,OU=defensoria,DC=defensoria
,DC=gdfnet,DC=d
LDAP_AUTH_BIND_PASSWORD=f&7A*mS9
LDAP_AUTH_BIND_SUFFIX=DC=defensoria,DC=gdfnet,DC=df
LDAP_AUTH_SERVER_URI=10.233.68.2

# Egide
EGIDE_CLIENT_ID=
EGIDE_CLIENT_SECRET=
EGIDE_URL=
EGIDE_REDIRECT_URI=

# E-mail
EMAIL_HOST=relay.gdfnet.df.gov.br
DEFAULT_FROM_EMAIL=no-replay-solar@defensoria.df.gov.br
EMAIL_TO_REPORT_ERRORS=no-replay-solar@defensoria.df.gov.br
EMAIL_HOST_PASSWORD=
EMAIL_HOST_USER=
EMAIL_PORT=25
EMAIL_USE_TLS=False
EMAIL_USE_SSL=False
EMAIL_TIMEOUT=None

# Notificações
SIGNO_REST_API_URL=
SIGNO_WEBSOCKET_URL=

# Chatbot
CHATBOT_LUNA_API_TOKEN=
CHATBOT_LUNA_WEBHOOK_URL=
CHATBOT_LUNA_VERIFY_CERTFILE=True
```

```
# Integração com E-Defensor (DPE-RR)
USAR_EDEFENSOR=False
EDEFENSOR_ADONIS_HOSTNAME=
EDEFENSOR_ADONIS_PORT=
EDEFENSOR_CHAT_WEBSERVICE_TOKEN_URL=
EDEFENSOR_CHAT_WEBSERVICE_TOKEN_USERNAME=
EDEFENSOR_CHAT_WEBSERVICE_TOKEN_PASSWORD=
EDEFENSOR_CHAT_WEBSERVICE_APP_SYSTEM=
EDEFENSOR_CATEGORIA_AGENDA_ID=

VERIFY_CERTFILE=
CORS_ORIGIN_ALLOW_ALL=True

# SMS
MOVILE_API_URL=http://api-messaging.movile.com/v1/send-sms
MOVILE_AUTH_TOKEN=
MOVILE_AUTH_USER=

# Outras configurações
SIGLA_UF=DF
SIGLA_INSTITUICAO=DPDF
NOME_INSTITUICAO=Defensoria Pública do Distrito Federal
CNPJ_INSTITUICAO=12219624000183

EXIBIR_ALERTA_AVALIACAO_ASSISTIDO=True
MENSAGEM_ALERTA_AVALIACAO_ASSISTIDO=<p></p>
EXIBIR_ALERTA_AVALIACAO_ASSISTIDO_129=True
MENSAGEM_ALERTA_AVALIACAO_ASSISTIDO_129=<p></p>

DATA_UPLOAD_MAX_MEMORY_SIZE=2621440
```

3. docker-compose

1. Acesse a pasta `/docker`, copie o arquivo `docker-compose.override.dev.yml` para a raiz do projeto e renomeie para `docker-compose.override.yml`.
2. No arquivo `docker-compose.yml` localizado na raiz, comente o serviço `consumers`:

```
#  consumers:
#    build:
#      context: .
#      dockerfile: docker/web/Dockerfile
#    depends_on:
#      - redis
#      - memcached
```

```
#     - python
#   image: gitlab.defensoria.to.def.br:5000/defensoria/sisat/web:latest
#   command: python manage.py start_consumers
#   volumes:
#     - ./media/:/app/src/media/
#     - ./data/wkhtmltox/:/app/src/binarios_executaveis/
#     - ./log:/app/src/log
#   env_file:
#     - .env
```

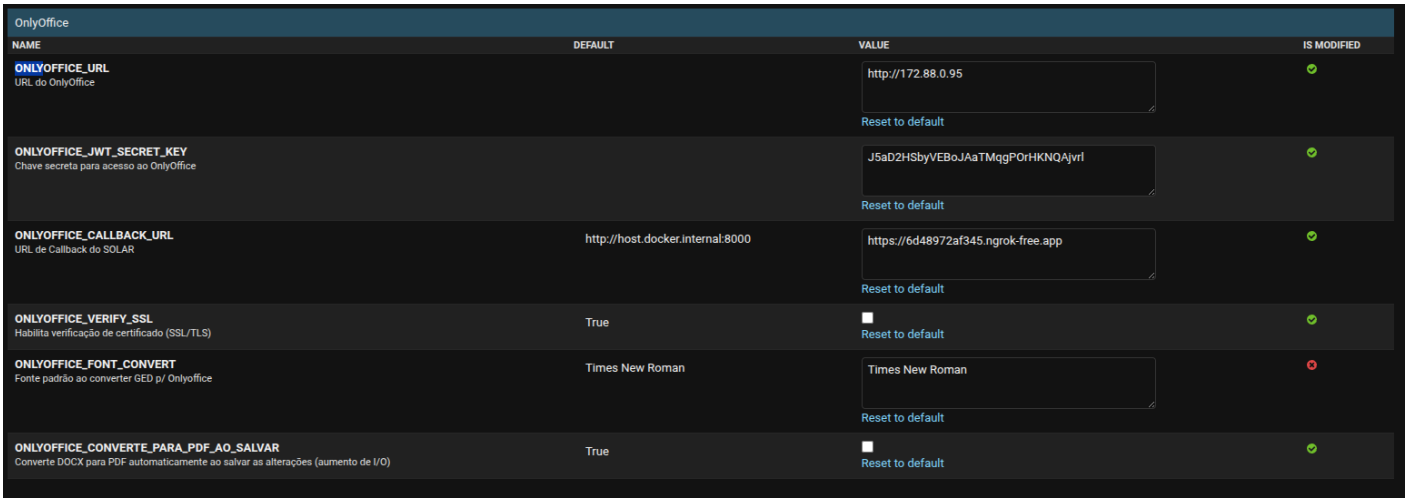
No terminal, execute o comando para subir a aplicação:

> docker compose up -d --build

Configuração do OnlyOffice

1. Acesse a aplicação em `http://localhost:8000`.
2. Faça login com suas credenciais.
3. No painel do Django Admin, localize a seção `constance` na barra lateral.
4. Configure os seguintes campos:

- **ONLYOFFICE_URL**: `http://172.88.0.95`
- **ONLYOFFICE_JWT_SECRET_KEY**: `J5aD2HSbyVEBoJAaTMqgPOrHKNQAjvrl`
- **ONLYOFFICE_CALLBACK_URL**: gere um link público da sua aplicação local (porta 8000) com o ngrok e insira a URL aqui.
  Exemplo: `https://6d48972af345.ngrok-free.app`



Considerações Importantes

Evite versionar arquivos Docker

Não adicione os arquivos de configuração Docker ao controle de versão do Git. Esses arquivos são compartilhados entre todos os ambientes e **não devem ser alterados ou enviados com mudanças locais**.

Revision #1
Created 18 July 2025 16:36:36 by Admin
Updated 18 July 2025 16:48:34 by Admin