Инструкция по установке СУР
на ОС Ubuntu (Debian)

Установка СУР.

Сервисные утилиты:

git

```
sudo apt install git
```

htop

```
sudo apt install htop
```

mc:

```
sudo apt install mc
```

curl:

```
sudo apt install curl
```

Проверить, что есть:

Версия системы:

```
cat /etc/issue
lsb_release -a
uname -a
```

Список служб:

```
systemctl list-units --type service
```

Список прослушиваемых портов:

```
sudo netstat -ntlp | grep LISTEN
```

Адреса:

```
ifconfig
```

Системные утилиты:

python

Посмотреть, какие python есть в системе:

```
ls /usr/bin/python*
```

Доустановить:

```
sudo apt install python python-pip
#sudo apt install ipython
pip install ipython
```

Для установки pip на новый Ubuntu:

```
sudo add-apt-repository universe
sudo apt update
sudo apt install python2.7
```

Установка pip для Python2.7 на новых

```
    # Fetch get-pip.py for python 2.7
    curl https://bootstrap.pypa.io/pip/2.7/get-pip.py --output get-pip.py

    python2 get-pip.py
    pip --version
```

supervisor

```
    sudo apt install supervisor
```

nginx

```
    sudo apt install nginx
```

Перезапуск:

```
    nginx -t    # Проверка синтаксиса и конфигурации
    nginx -s reload
```

uwsgi

```
    sudo apt install uwsgi uwsgi-plugin-python
```

redis

```
    sudo apt install redis-server
```

Проверка:

```
    redis-cli
```

postgres

Есть варианты в зависимости от версии системы. Информация тут:
https://www.postgresql.org/download/linux/ubuntu/

Для Ubuntu Bionic (18.04):

```
    echo "deb http://apt.postgresql.org/pub/repos/apt/ bionic-pgdg main" >
/etc/apt/sources.list.d/pgdg.list

    wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo
apt-key add -

    sudo apt update

    sudo apt install postgresql-11 postgresql-server-dev-11
```

вариант для Ubuntu:

```
    apt-get install postgresql-12
```

Запуск терминала для проверки:

```
    sudo -u postgres psql
```

Выход:

```
\q
```

Развертывание СУР

Настройка БД

Создаем пользователя

```
sudo -u postgres psql
CREATE ROLE sur WITH LOGIN CREATEDB PASSWORD '********';
\q
```

Создаем базу:

```
createdb -U sur -h 127.0.0.1 -E 'UTF8' db_sur
createdb -U sur -h 127.0.0.1 -E 'UTF8' db_sur_arch
```

нужно создать именно 2 базы.

пароль для базы: *******

Если ошибка:

```
could not connect to database template1:
```

Поправить разрешения в /etc/postgresql/<xxx>/main/pg_hba.conf
Установить тип разрешения md5

Файл может находится в /var/lib/pgsql (CentOS?)
Потом перезапустить

При проблемах с кодировками:

```
createdb -U sur -h 127.0.0.1 --lc-collate='C.UTF-8' --lc-ctype='C.UTF-8' -E 'UTF8' -T template0 db_sur

createdb -U sur -h 127.0.0.1 --lc-collate='ru_RU.UTF-8' --lc-ctype='ru_RU.UTF-8' -E 'UTF8' -T template0 db_sur
```

Как правило команда `locale` позволяет увидеть наличие в системе

Если база была создана ранее от другого пользователя

```
ALTER DATABASE name OWNER TO new_owner;
GRANT USAGE ON SCHEMA public TO my_user;
GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA public TO my_user;
GRANT ALL PRIVILEGES ON ALL SEQUENCES IN SCHEMA public TO my_user;
```

Копируем архив пустой базы:

Скачиваем архив

```
curl -L http://officer24.ru/static/sur/db_sur_clear_********.gz >/var/backups/db_sur_clear.gz
```

Или

```
curl -L http://officer24.ru/static/sur/sur_clear_mig_*******.gz >/var/backups/sur_clear_mig_**********.gz
```

на сервере разворачиваем архив:

```
gunzip -c /var/backups/db_sur_clear.gz | psql -U sur -h 127.0.0.1 nameDb

gunzip -c /var/backups/sur_clear_mig_20230721.gz | psql -U sur -h 127.0.0.1
nameDb
```

проверяем:

```
psql -U sur -h 127.0.0.1 nameDb
\dt
\q
```

Переносим коды СУРа

Создаем каталоги

```
mkdir -pv /srv/django && cd /srv/django
```

Клонируем репо:

```
git clone http://alex@gitea.officer24.ru/*****/*******.git
```

Переходим на ветку:

```
cd sur
git checkout origin/**********
git checkout -b srv_<код клиента>
```

Досоздаем каталоги:

```
mkdir -pv /srv/django/sur/logs
mkdir -pv /srv/django/sur/media

mkdir -pv logs
mkdir -pv media
```

Ставим зависимости:
(sudo обязательно)

```
sudo pip install -r ./requirements.txt
```

Создаем и настраиваем settings_local:

```
cp /srv/django/sur/deploy/settings_local_example.py
/srv/django/sur/apps/settings_local.py

cp ./deploy/settings_local_example.py ./apps/settings_local.py
```

запускаем скрипт:

```
python gen_secret_key.py
```

копируем в буфер обмена выдачу скрипта (SECRET_KEY...)

Открываем в редакторе:

```
mcedit ./apps/settings_local.py
```

и корректируем необходимые настройки:

- Заменяем строку SECRET_KEY... сгенерированной

- ROOT_URL указываем свой
- ALLOWED_HOSTS = [...] Указываем свои.
- TIME_ZONE
- Настройки EMAIL_
- ADMINS
- MAPS
- пароли базы данных, если использованы свои
- ключи FCM
- список панелей

https://en.wikipedia.org/wiki/List_of_tz_database_time_zones

Выставляем права на файлы:

```
cd /srv/django

sudo chown -R www-data:www-data ./sur
sudo chmod -R o+r,g+rw ./sur
cd sur
```

Проверяем:

```
python manage.py showmigrations

python manage.py showmigrations | grep "\[ \]"
```

Если не все крестики выполнить:

```
python manage.py migrate
python manage.py migrate --database=archives
```

Проверка работы отсылки почты:

```
python manage.py sendtestemail <адрес>
```

Установить пароль для стартового пользователя:

```
python manage.py changepassword demo
```

Установить лимиты пользователей.

Настраиваем службы:

скрипт архивирования базы:

```
cp /srv/django/sur/deploy/backup_sur_plain.sh /srv/django/
sudo cp /srv/django/sur/deploy/pgpass /root/.pgpass
sudo chmod 0600 /root/.pgpass
```

Или

```
cp ./deploy/backup_sur_plain.sh ../
sudo cp ./deploy/pgpass /root/.pgpass
sudo chmod 0600 /root/.pgpass
```

Если нужно указать другой пароль к базе нужно указать его в файле /root/.pgpass

Проверяем:

```
sudo /srv/django/backup_sur_plain.sh
ls -lh /var/backups/db_sur_*
```

```
    sudo ../backup_sur_plain.sh
    ls -lh /var/backups/db_sur_*
```

Должен быть файл вида /var/backups/db_sur_plain_*****.gz

uWSGI:

```
    sudo cp /srv/django/sur/deploy/uwsgi/sur.ini /etc/uwsgi/apps-available/
    sudo ln -s /etc/uwsgi/apps-available/sur.ini /etc/uwsgi/apps-enabled/
    sudo systemctl restart uwsgi

    sudo cp ./deploy/uwsgi/sur.ini /etc/uwsgi/apps-available/
    sudo ln -s /etc/uwsgi/apps-available/sur.ini /etc/uwsgi/apps-enabled/
    sudo systemctl restart uwsgi
```

Если поправить:

```
    sudo mcedit /etc/uwsgi/apps-available/sur.ini
```

Возможно в файле поправить параметр `workers = 3` установить больше или меньше

Проверка:

```
    ps -A -o ppid,pid,command | grep [u]wsgi
```

Должно быть 4 процесса. (если `workers = 3` не меняли)

Проверить лог, что нет ошибок:

```
    sudo cat /var/log/uwsgi/app/sur.log
```

Фраза "unable to stat() /srv/django/sur/touchme,..." - нормально

Важно запустить uwsgi до nginx

Вариант Gunicorn

Вместо uWSGI

Устанавливаем:

```
    pip install gunicorn
```

Копируем *****.ini

Nginx:

Убедится, что дефолтовая конфигурация не конфликтует с СУР по портам в файле: `/etc/nginx/sites-enabled/default` или убрать его из `sites-enabled`.

```
    cp /srv/django/sur/deploy/nginx/sur.ini /etc/nginx/sites-available

    cp ./deploy/nginx/sur.ini /etc/nginx/sites-available/sur.ini
```

Открыть файл /etc/nginx/sites-available/******.ini на редактирование, поправить порт сервера и адрес. Параметры `listen` и `server_name`.

```
    mcedit /etc/nginx/sites-available/*****.ini
```

Активировать:

```
ln -s /etc/nginx/sites-available/sur.ini /etc/nginx/sites-enabled
```

Или

```
ln -s /etc/nginx/sites-available/sur.ini /etc/nginx/sites-enabled
```

Проверить и перезапустить nginx:

```
nginx -t    # Проверка синтаксиса и конфигурации
nginx -s reload
```

Проверить работу сервера из браузера.

Ротация логов nginx:

Убедится, что система logrotate установлена:

```
sudo cat /etc/logrotate.d/nginx
```

Далее вписать настройку СУР:

```
sudo cp /etc/logrotate.d/nginx /srv/
sudo cat /etc/logrotate.d/nginx ./deploy/logrotate > /etc/logrotate.d/nginx
```

Если поравить:

```
sudo mcedit /etc/logrotate.d/nginx
```

Проверить корректность настроек командой:

```
logrotate -d -f /etc/logrotate.d/
```

В выдаче найти информацию о ротации логов /srv/django/sur/logs/nginx...

Supervisor:

```
sudo cp /srv/django/sur/deploy/supervisor/sur.conf /etc/supervisor/conf.d/

sudo cp ./deploy/supervisor/sur.conf /etc/supervisor/conf.d/
```

Удалить или закомментировать лишние драйвера приборов, поправить порты нужных:

```
sudo mcedit /etc/supervisor/conf.d/****.conf
```

Перезапустить

```
sudo supervisorctl
reread
update
status
```

Нормально должны быть остановлены:

```
backup_db                        STOPPED   Not started
check-bind                       STOPPED   Not started
check-bind-all                   STOPPED   Not started
check_user_status                STOPPED   Not started
recharge                         STOPPED   Not started
update_panels                    STOPPED   Not started
```

Нормально должны работать:

```
calc-state                       RUNNING   pid 27986, uptime 0:05:50
celery                           RUNNING   pid 28087, uptime 0:03:43
```

Остальные процессы - драйвера приборов. Долдны работать если включены.
Если есть проблемы - см. логи в `/srv/django/sur/logs`

Выйти из управления supervisor

```
quit
```

Проверить работу архиватора через супервизор:

```
sudo supervisorctl start backup_db
```

Убедится, что отработало без ошибок:

```
sudo supervisorctl status backup_db
```

Проверить архив:

```
ls -lh /var/backups/db_sur_*
```

Cron:

```
sudo crontab -u root -e
```

Изменить данные, вписав из образца. Выйти с сохранением

Запуск

Зайти на web сервер.

Проверки

в shell:

```
python manage.py shell
from z_utils import dj_model as djm
djm.help()

djm.show_data("auth.User")  # Показать пользователей в системе
```