Инструкция по установке экземпляра

ПО «Контроль платежей»

Оглавление

1. Требования к системе

Для установки ПО «Контроль платежей» необходимо:

- Операционная система: Linux
- Размер оперативной памяти не менее: 2 Гб
- Размер места на диске не менее: 10 Гб
- Подключение к сети: internet
- Предустановленный PostgreSQL, Nginx
- Приобретено доменное имя

1.1. Настройка базы данных PostgreSQL

- Создаём базу данных

```
CREATE DATABASE kpln_db OWNER postgres;
```

- Изменение локали "lc_monetary" на "ru_RU.UTF-8"
  - Проверить, есть ли в ОС такая локаль # locale
  - Если нет, # sudo nano /etc/locale.gen, ищем нужную локаль, раскомментируем строку, сохраняем изменения `Ctrl + O`=>`Ctrl + X`.
  - Генерируем локали sudo locale-gen
  - Подключаемся к БД, находим место положение файла конфигурации psql -U postgres -c 'SHOW config_file'
  - В нём меняем lc_monetary = 'ru_RU.UTF-8', остальные локали должны быть en_US.utf8
- Восстанавливаем схему базы данных

```
psql -U postgres kpln_db < '/*путь к файлу*/dump-payments 2023-11-29 (1).sql'
```

1.2. Установка компонентов из репозиториев Ubuntu

Первым шагом будет установка всех необходимых частей из репозиториев Ubuntu. Сюда входит `pip`, менеджер пакетов Python, который будет управлять компонентами Python. Вы также получите файлы разработки Python, необходимые для создания некоторых компонентов Gunicorn.

Сначала обновите локальный индекс пакетов и установите пакеты, которые

позволят вам создать среду Python. Они будут включать `python3-pip`, а также еще несколько пакетов и инструментов разработки, необходимых для надежной среды программирования:

```
sudo apt update
sudo apt install python3-pip python3-dev build-essential libssl-dev
libffi-dev python3-setuptools
```

1.3. Создание виртуальной среды Python

Далее вы настроите виртуальную среду, чтобы изолировать приложение Flask от других файлов Python в вашей системе.

Начните с установки пакета `python3-venv`, который установит модуль `venv`:

```
sudo apt install python3-venv
```

Далее создайте родительский каталог для вашего проекта Flask. Перейдите в каталог с помощью команды `cd` после его создания:

```
mkdir ~/myproject
cd ~/myproject
```

Создайте виртуальную среду для хранения требований Python вашего проекта Flask, набрав:

```
python3 -m venv venv
```

При этом будет установлена локальная копия Python и `pip` в каталог `myprojectenv` внутри каталога вашего проекта.

Перед установкой приложений в виртуальной среде необходимо ее активировать. Сделайте это, набрав:

```
source venv/bin/activate
```

Ваше приглашение изменится, показывая, что теперь вы работаете в виртуальной среде. Это будет выглядеть примерно так: `(myprojectenv)user@host:~/myproject$`

1.4. Настройка приложения Flask

Распаковываем содержащийся в загруженном выше по предоставленной ссылке архиве с экземпляром ПО архив «**kpln - 2023.11.29_1.rar**» в директорию `myproject`. Устанавливаем необходимые пакеты из файла **requirements.txt** командой

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

Перед установкой приложений в виртуальной среде необходимо ее активировать. Сделайте это, набрав:

Далее установите Gunicorn:

```
pip install gunicorn
```

1.5. Настройка Gunicorn

Теперь ваше приложение настроено с установленной точкой входа. Теперь вы можете перейти к настройке Gunicorn.

Прежде чем двигаться дальше, убедитесь, что Gunicorn может корректно обслуживать приложение.

Вы можете сделать это, передав ему имя точки входа приложения. Он состоит из имени модуля (без расширения `.py`) плюс имени вызываемого объекта в приложении. В данном случае это `wsgi:app`.

Также укажите интерфейс и порт для привязки, используя аргумент `0.0.0.0:5000`, чтобы приложение запускалось на общедоступном интерфейсе:

```
cd ~/myproject
gunicorn --bind 0.0.0.0:5000 wsgi:app
```

Вы должны увидеть вывод, подобный следующему:

```
Output
[2020-05-20 14:13:00 +0000] [46419] [INFO] Starting gunicorn 20.0.4
```

```
[2020-05-20 14:13:00 +0000] [46419] [INFO] Listening at:
http://0.0.0.0:5000 (46419)
[2020-05-20 14:13:00 +0000] [46419] [INFO] Using worker: sync
[2020-05-20 14:13:00 +0000] [46421] [INFO] Booting worker with pid: 46421
```

Снова посетите IP-адрес вашего сервера с добавленным в конец `:5000` в веб-браузере:

```
http://your_server_ip:5000
```

Должна отобразиться главная страница ПО «Контроль платежей»

Когда вы закончите использовать виртуальную среду, вы можете деактивировать ее:

```
deactivate
```

Затем создайте файл сервисного модуля systemd. Создание юнит-файла systemd позволит системе инициализации Ubuntu автоматически запускать Gunicorn и обслуживать приложение Flask при каждой загрузке сервера.

Для начала создайте модульный файл, заканчивающийся на `.service` в каталоге `/etc/systemd/system`:

```
sudo nano /etc/systemd/system/myproject.service
```

Внутри вы начнете с раздела `[Unit]`, который используется для указания метаданных и зависимостей. Добавьте сюда описание вашей службы и укажите системе инициализации, чтобы она запускалась только после достижения сетевой цели.

Далее добавьте раздел `[Service]`. Это укажет пользователя и группу, под которыми вы хотите запустить процесс. Предоставьте право собственности на процесс вашей обычной учетной записи пользователя, поскольку ей принадлежат все соответствующие файлы. Также передайте право владения группой www-data, чтобы Nginx мог легко взаимодействовать с процессами Gunicorn. Не забудьте заменить здесь имя пользователя своим именем пользователя

Далее наметьте рабочий каталог и установите переменную среды `PATH`, чтобы система инициализации знала, что исполняемые файлы процесса расположены в нашей виртуальной среде. Также укажите команду для

запуска службы. Эта команда выполнит следующее:

- Запустите 3 рабочих процесса (хотя вам следует настроить это по мере необходимости)
- Создайте файл сокета Unix `myproject.sock` и привяжите его к нему в каталоге нашего проекта. Мы установим значение umask `007`, чтобы файл сокета создавался с предоставлением доступа владельцу и группе и ограничением другого доступа
- Укажите имя файла точки входа WSGI, а также вызываемый Python в этом файле (`wsgi:app`)

Systemd требует, чтобы вы указали полный путь к исполняемому файлу Gunicorn, который установлен в вашей виртуальной среде.

Не забудьте заменить имя пользователя и пути к проекту своей собственной информацией

Наконец, добавьте раздел `[Install]`. Это сообщит systemd, с чем связать эту службу, если вы включите ее запуск при загрузке. Вам нужно, чтобы эта служба запускалась при запуске обычной многопользовательской системы:

/etc/systemd/system/myproject.service

```
[Unit]
Description=Gunicorn instance to serve myproject
After=network.target

[Service]
User=root
Group=www-data
WorkingDirectory=/var/www/myproject
Environment="PATH=/var/www/myproject/venv/bin"
ExecStart=/var/www/myproject/venv/bin/gunicorn --workers 3 --bind
unix:kpln.sock -m 007 wsgi:app

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

На этом ваш служебный файл systemd готов. Сохраните и закройте его сейчас.

Теперь вы можете запустить созданную вами службу Gunicorn и включить ее, чтобы она запускалась при загрузке:

```
sudo systemctl start myproject
sudo systemctl enable myproject
```

Проверим статус:

```
sudo systemctl status myproject
```

Вы должны увидеть такой вывод:

```
Output

● kpln.service - Gunicorn instance to serve kpln
     Loaded: loaded (/etc/systemd/system/kpln.service; enabled; vendor
preset: enabled)
     Active: active (running) since Fri 2023-12-15 06:47:24 UTC; 4 days
ago
   Main PID: 775 (gunicorn)
      Tasks: 7 (limit: 2308)
     Memory: 233.7M
        CPU: 1min 41.817s
     CGroup: /system.slice/kpln.service
             ├─775 /var/www/kpln/venv/bin/python3
/var/www/kpln/venv/bin/gunicorn --workers 3 --bind unix:kpln.sock -m 007
wsgi:app
             ├─855 /var/www/kpln/venv/bin/python3
/var/www/kpln/venv/bin/gunicorn --workers 3 --bind unix:kpln.sock -m 007
wsgi:app
             ├─866 /var/www/kpln/venv/bin/python3
/var/www/kpln/venv/bin/gunicorn --workers 3 --bind unix:kpln.sock -m 007
wsgi:app
             └─867 /var/www/kpln/venv/bin/python3
/var/www/kpln/venv/bin/gunicorn --workers 3 --bind unix:kpln.sock -m 007
wsgi:app

Dec 15 06:47:24 lqmsfftrso systemd[1]: Started Gunicorn instance to serve
kpln.
```

Если вы видите какие-либо ошибки, обязательно устраните их, прежде чем продолжить работу с руководством.

1.6. Настройка Nginx для прокси-запросов

Теперь ваш сервер приложений Gunicorn должен быть запущен и ожидать запросов к файлу сокета в каталоге проекта. Теперь вы можете настроить Nginx для передачи веб-запросов в этот сокет, внеся небольшие дополнения в его файл конфигурации.

Начните с создания нового файла конфигурации блока сервера в каталоге `sites-available` Nginx. Назовите его `myproject`, чтобы соответствовать остальной части руководства:

```
Sudo nano /etc/nginx/sites-available/myproject
```

Откройте блок сервера и укажите Nginx, чтобы он прослушивал порт по умолчанию 80. Также укажите ему использовать этот блок для запросов доменного имени нашего сервера

/etc/nginx/sites-available/myproject

```
server {
    server_name www.your_domain your_domain;

    location / {
        include proxy_params;
        proxy_pass http://unix:/var/www/myproject/myproject.sock;
    }

    listen 443 ssl; # managed by Certbot
    ssl_certificate /etc/letsencrypt/live/kpln-
employees.ru/fullchain.pem; # managed by Certbot
    ssl_certificate_key /etc/letsencrypt/live/kpln-
employees.ru/privkey.pem; # managed by Certbot
    include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot
    ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot
}
server {
    if ($host = kpln-employees.ru) {
        return 301 https://$host$request_uri;
    } # managed by Certbot


    if ($host = www.kpln-employees.ru) {
        return 301 https://$host$request_uri;
    } # managed by Certbot
```

```
    listen 80;
    server_name kpln.ru www.kpln-employees.ru kpln-employees.ru;
    return 404; # managed by Certbot
}
```

Сохраните и закройте файл, когда закончите.

Чтобы включить только что созданную конфигурацию блока сервера Nginx, свяжите файл с каталогом `sites-enabled`:

```
sudo ln -s /etc/nginx/sites-available/myproject /etc/nginx/sites-enabled
```

Имея файл в этом каталоге, вы можете проверить наличие синтаксических ошибок:

```
sudo nginx -t
```

Если это возвращается без каких-либо проблем, перезапустите процесс Nginx, чтобы прочитать новую конфигурацию:

```
sudo systemctl restart nginx
```

Наконец, снова настройте брандмауэр. Вам больше не нужен доступ через порт `5000`, поэтому вы можете удалить это правило. Затем вы можете разрешить полный доступ к серверу Nginx:

```
sudo ufw delete allow 5000
sudo ufw allow 'Nginx Full'
```

Теперь вы сможете перейти к доменному имени вашего сервера в веб-браузере

Примечание. Вы получите сообщение об ошибке шлюза HTTP 502, если Nginx не сможет получить доступ к файлу сокета Gunicorn. Обычно это происходит потому, что домашний каталог пользователя не позволяет другим пользователям получать доступ к файлам внутри него.
Если ваш файл сокета называется `/var/www/myproject/myproject.sock`, убедитесь, что `/var/www` имеет минимум `0755` разрешений. Вы можете использовать такой инструмент, как `chmod`, чтобы изменить разрешения следующим образом:

```
sudo chmod 755 /var/www/myproject
```

Затем перезагрузите страницу и проверьте, исчезла ли ошибка HTTP 502.

Если вы столкнулись с какими-либо ошибками, попробуйте проверить следующее:

- `sudo less /var/log/nginx/error.log`: проверяет журналы ошибок Nginx.
- `sudo less /var/log/nginx/access.log`: проверяет журналы доступа Nginx.
- `sudo journalctl -u nginx`: проверяет журналы процессов Nginx.
- `sudo journalctl -u myproject`: проверяет журналы Gunicorn вашего приложения Flask.

1.7. Защита приложения

Получение SSL-сертификата для вашего домена. Ссылка на инструкцию

2. Результат установки и доступ

После выполнения действий из пункта 1 данной инструкции приложение будет доступно через браузер по адресу вашего сайта.

Далее необходимо добавить пользователей с ролями «Руководитель», «Бухгалтер» и «Пользователи»

2.1. Веб-клиент

Требования к системе
Для проверки функционала визуальных элементов подойдет любой браузер с последней версией.

Для тестирования доступен весь функционал программы.