|  |  |  |  |
| --- | --- | --- | --- |
| **Студент:** | Еселидзе Д.В. | **Допуск:** | Муравьёв К.А. |
| **Отчет по лабораторной работе №2**  «Установка python и запуск веб-сервера flask» | | | |
| Дата выполнения | 2.12.2022 | Дата защиты | 19.12.2022 |
| Оценка |  | Подпись |  |

Цель работы:

|  |
| --- |
| Получить навыки по установке и сборке из исходного кода пакетов в Linux, запуску веб-приложений flask |

Задачи работы:

|  |
| --- |
| Собрать пакет python, подключить flask к нашему проекту, настроить работу Nginx, wsgi-сервера gunicorn, подключить базы данных для хранения информации. |

Краткий конспект теоретической части:

|  |
| --- |
| Python:высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости кода и его качества, а также на обеспечение переносимости написанных на нём программ  Виртуальное окружение (venv): это каталог, в который устанавливаются некоторые исполняемые файлы и скрипты. Оно удобно тем, что легко контролируется, не вредит операционной системе и уберегает от конфликтов версий.  Установка и сборка пакетов в Linux: установка и сборка пакетов Linux осуществляется с помощью файлового менеджера yum, а командой устанавливающей пакеты является yum install ‘наименования пакета’.  Установка модулей Python: происходит аналогично установке пакетов Linux с помощью pip.  Настройка фаерволла в Linux: фаервол защищает сеть от нежелательных проникновений. Упарвление фаерволом осуществляется по средствам команд firewall -cmd. После применения данной команды стоит перезагрузить фаервол командой sudo firewall -cmd –reload.  Запуск, остановка и диагностика сервисов в Linux: управлять работой служб в Linux можно с помощью команды systemctl. |

**1 Установка и настройка Python**

Для начала работы стоит получить python самой свежей версии. Чтобы это сделать для начала мы установим дополнительное ПО, а затем клонируем репозиторий. (рисунок 1.1).

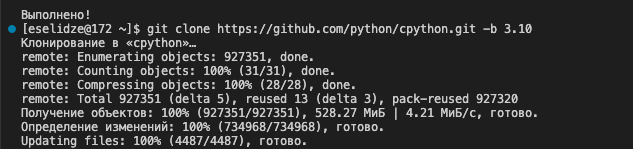


Рисунок 1.1 – Клонирование репозитория.

После того как клонировали репозиторий мы должны настроить конфигурацию (рисунок 1.2) и собрать python с помощью команд “./configure --enable-optimizations” и “make -s -j2” соответственно (рисунок 1.3).

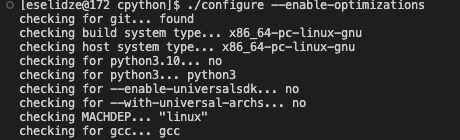


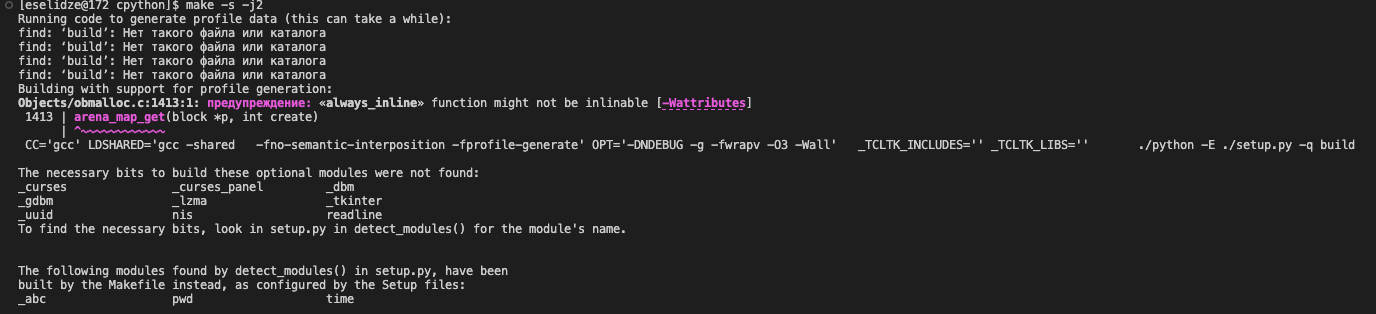
Рисунок 1.2 – Настройка конфигурации python. 

Рисунок 1.3 – Сборка python.

Следующее что мы сделаем, будет проверка версии python (рисунок 1.4).

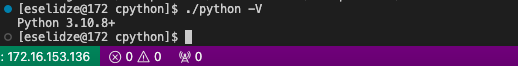


Рисунок 1.4 – Проверка версии python.

Для нашего удобства создадим виртуальное окружение, в которое будут загружаться нужные файлы и пакеты, но перед этим проверим версию pip – пакетный менеджер, использованный нами для установки пакетов python (рисунок 1.5).

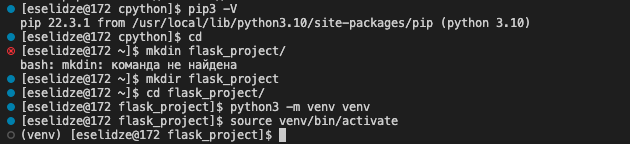


Рисунок 1.5 – Проверка версии pip и создание виртуального окружения.

Следующим действием будет установка flask – фреймворк для создания веб-приложений (рисунок 1.6).

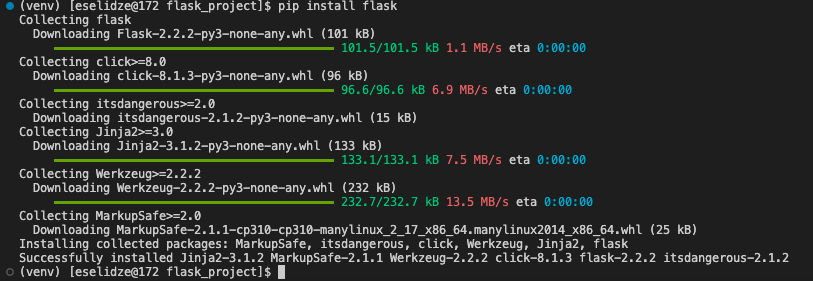


Рисунок 1.6 – Установка flask.

Для проверки работоспособности flask создадим и запустим самое простое приложение (рисунок 1.7).

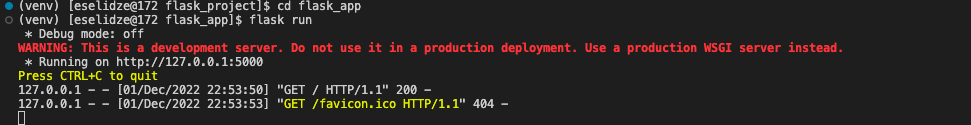


Рисунок 1.7 – Запуск приложения.

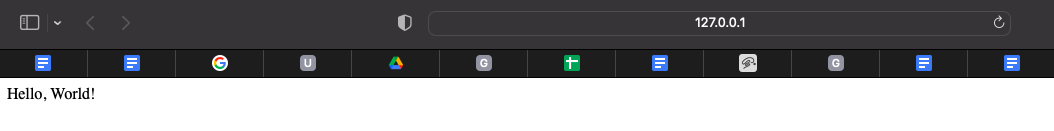
Вот так работа приложения выглядит в браузере (рисунок 1.8). 

Рисунок 1.8 – Работа приложения в браузере.

Для тог чтобы приложение работало по IP нашей виртуальной машины нужно подключить 5000 порт в настройках фаервола и запустить наше приложение (рисунок1.9).

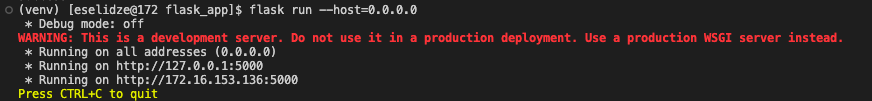


Рисунок 1.9 – Подключение 5000 порта и запуск приложения.

Убедимся в правильной работе приложения, перейдя в браузере по IP нашей виртуальной машины (рисунок 1.10).

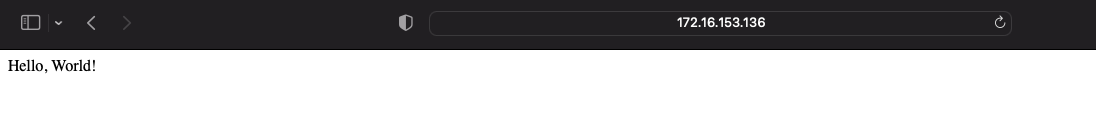


Рисунок 1.10 – Приложение, запущенное по IP нашей виртуальной машины.

На данном этапе мы создали полностью рабочее приложение, работающее с помощью flask и открывающееся по IP нашей виртуальной машины.

**2 Установка веб-сервера Nginx**

Для начала установим дополнительные пакеты (рисунок 2.1).

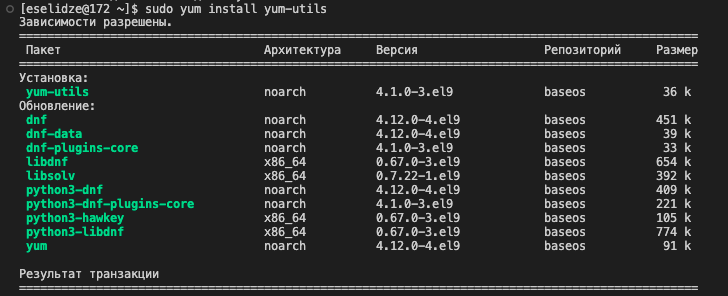


Рисунок 2.1 – Установка дополнительных пакетов.

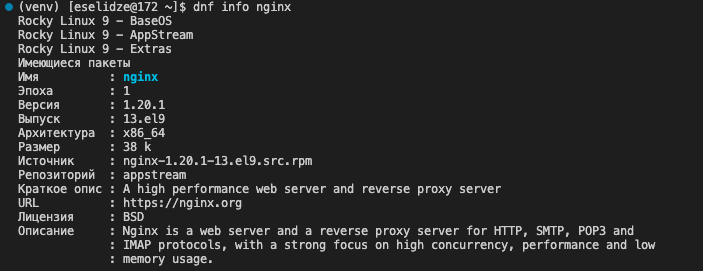
После создания файла nginx.repo и копирования в него содержимого со страницы сайта разработчика мы проверяем какую версию и из какого репозитория будет устанавливать пакетный менеджер (рисунок 2.2). 

Рисунок 2.2 – Проверка версии и репозитория.

Потом мы установили Nginx, используя dnf install nginx -y (рисунок 2.3).

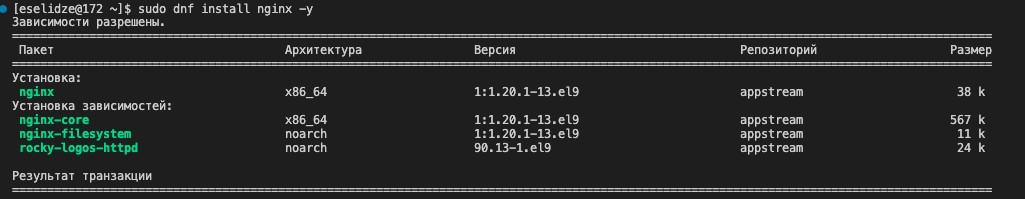


Рисунок 2.3 – Установка Nginx.

После мы удостоверились, что веб-сервер работает и готов принимать http запросы (рисунок 2.4).

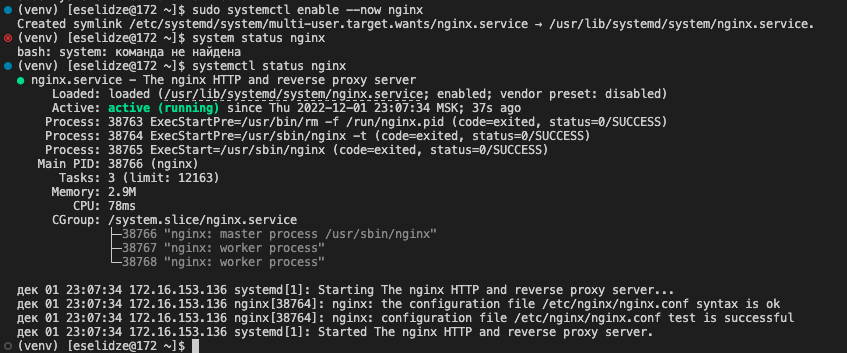


Рисунок 2.4 – Проверка работоспособности Nginx.

После провальной попытки запуска сайта мы понимаем, что в настройках фаерволла по умолчанию нет разрешающих правил для протоколов http и https, которые используются браузером при обращении к веб-серверу. С помощью соответствующих команд устраняем проблему и снова проверяем наличие нужных нам правил

(рисунок 2.5).



Рисунок 2.5 – Изменение и проверка настроек фаервола.

Далее осуществляем проверку работоспособности веб-сервера Nginx

(рисунок 2.6).



Рисунок 2.6 – Проверка работоспособности веб-сервера.

На данном этапе мы разработали веб-сервер принимающий http и запросы и разрешающий правила для протоколов http и https, которые используются браузером при обращении к веб-серверу.

**3 Управление конфигурацией веб-сервера Nginx**

Первым делом выдаем права нужные для изменения названия файла default.conf, а далее меняем название (рисунок 3.1).

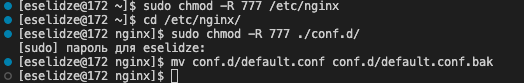


Рисунок 3.1 – Получение прав и смена названия файла.

Далее создаем файл flask\_app.conf и сохраняем его. Чтобы убедиться в отсутствии ошибок пропишем команду nginx -t (рисунок 3.2).



Рисунок 3.2 – Проверка правильности синтаксиса.

Перейдя в браузере по IP нашей виртуальной машины, мы увидим ошибку. Для ее устранения были прочитаны логи nginx. Это помогло обнаружить ошибку, а именно работа selinux. Отключим его (рисунок 3.3).



Рисунок 3.3 – Отключение selinux.

Так же отключим selinux отредактировав конфигурационный файл (рисунок 3.4).

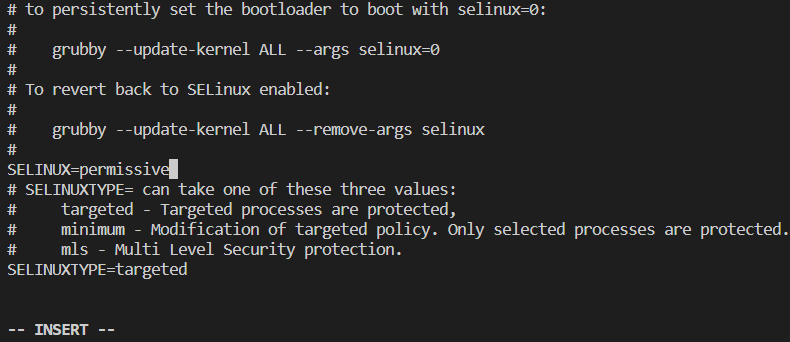


Рисунок 3.4 – Отключение selinux в конфигурационном файле.

Проделав все вышеперечисленное, запустим наше приложение и перейдем в браузере по IP нашей виртуальной машины (рисунок 3.5).

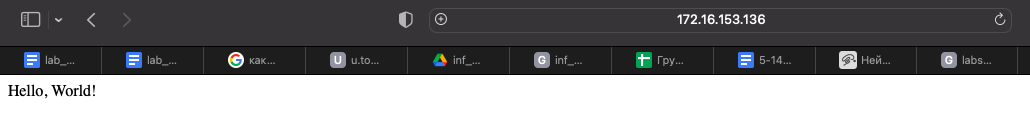


Рисунок 3.5 – Проверка работоспособности приложения.

По итогам проделанной на данный момент работы мы имеем python приложение доступное в браузере по http.

**4 УСТАНОВКА И ЗАПУСК WSGI-СЕРВЕРА**

Для начала создадим файл wsgi.py, позволяющий запускать наше приложение, и изменим содержимое файла app.py. Затем установим gunicorn (рисунок 4.1).



Рисунок 4.1 – Установка gunicorn.

Далее укажем интерфейс и порт для привязки, чтобы наше приложение запускалось через общедоступный интерфейс. Запустим сервер. (рисунок 4.2).

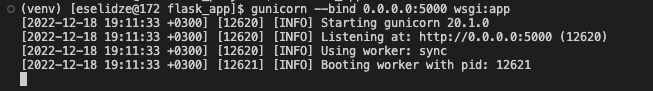


Рисунок 4.2 – Запуск сервера.

Создали новый файл flask\_app.service для системного сервиса (рисунок 4.3).

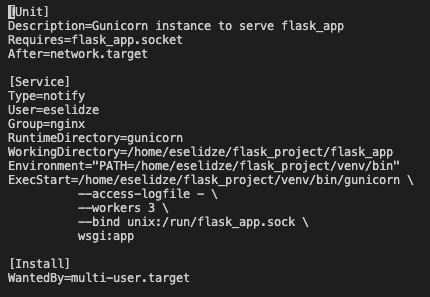


Рисунок 4.3 – содержимое файла flask\_app.service.

Также создадим еще один файл flask\_app.socket, отвечающий за связь nginx и gunicorn (рисунок 4.4).

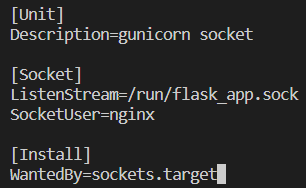


Рисунок 4.4 – Содержимое файла flask\_app.socket.

Затем изменим файл flask\_app.conf, перезагрузим нашу виртуальную машину и наше приложение станет доступно по 80 порту (рисунок 4.5).

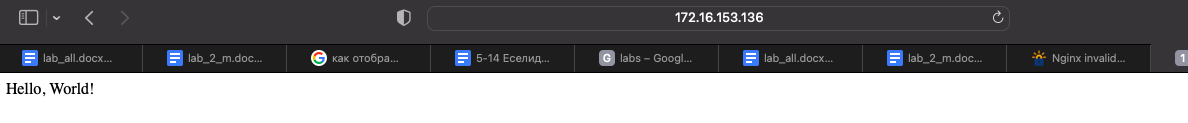


Рисунок 4.5 – Доступность приложения по 80 порту.

В результате мы имеем приложение, работающее через 80 порт и wsgi-сервер.

**5 Использование СУБД для хранения информации**

Для начала удалим папку flask\_app и склонируем готовый проект с возможностью авторизации и сохранением данных в нем (рисунок 5.1).

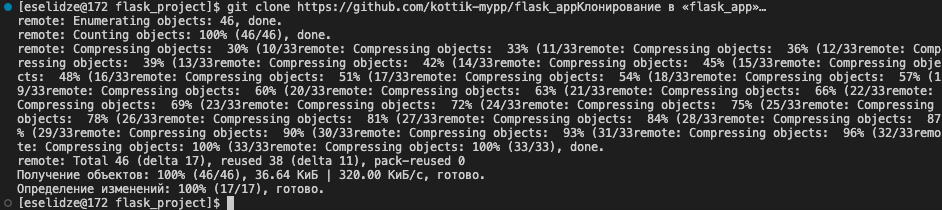


Рисунок 5.1 – Клонирование проекта.

Следующим действием установим необходимые модули и пакеты для запуска нашего приложения (рисунок 5.2).



Рисунок 5.2 – Установка необходимых модулей и пакетов.

Потом перезагрузим службу и сокет приложения (рисунок 5.3).

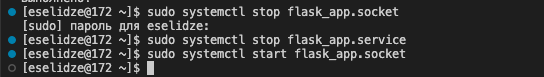


Рисунок 5.3 – Перезагрузка сокет приложений и службы.

Перейдем на страницу в браузере и увидим работу приложения (рисунок 5.4).

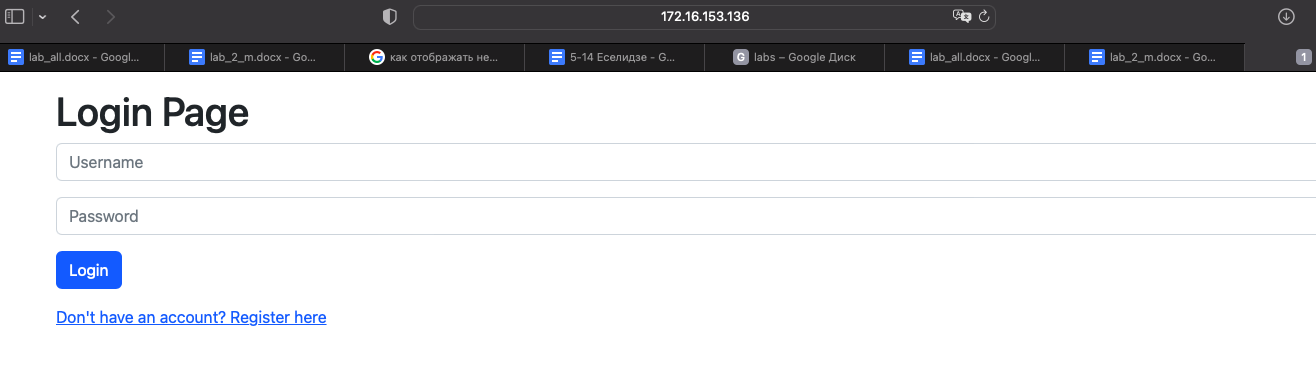


Рисунок 5.4 – работа приложения в браузере.

За данную лабораторную работу мы установили python, настроили работу wsgi-сервера, имеем автоматически работающее приложение с авторизацией и хранением информации в СУБД.