**Задание 6.**

**Задача 1. Создание пользователя**

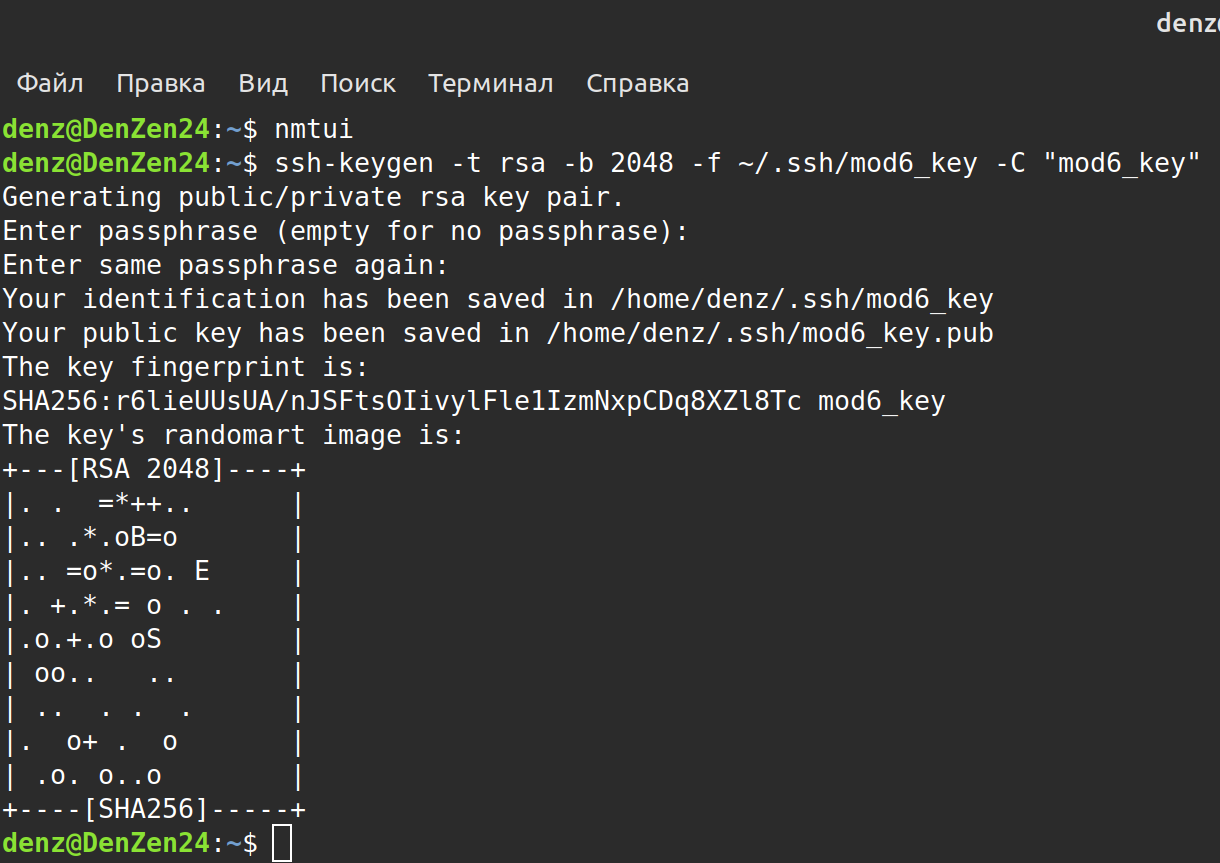
**Что нужно сделать**

Многопользовательская архитектура даёт возможность управлять правами доступа для выполнения различных действий. Если бы все пользователи обладали столь широкими правами, крах системы был бы неизбежен. Поэтому необходимо обезопасить систему. Кроме того, чтобы система стала по-настоящему удобной, каждому пользователю необходимо пространство, которое доступно только ему.

Потренируемся в организации такой архитектуры.

1. Создайте локально новый файл с rsa-ключом.

ssh-keygen -t rsa -b 2048 -f ~/.ssh/mod6\_key -C "mod6\_key"



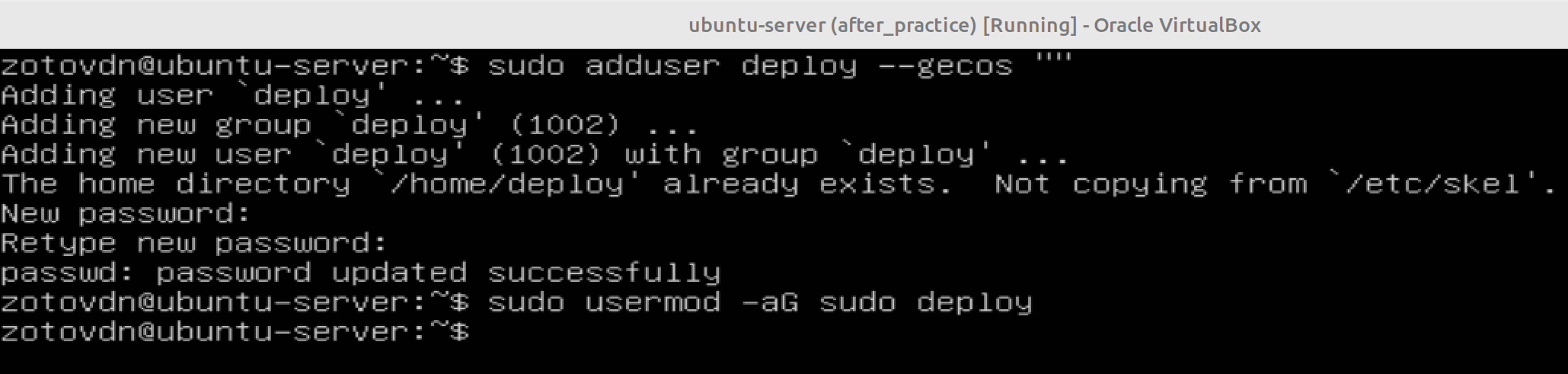
1. Создайте на удалённой машине нового пользователя командой adduser.

# ключ --gecos "" пропускает вопросы

sudo adduser deploy --gecos ""

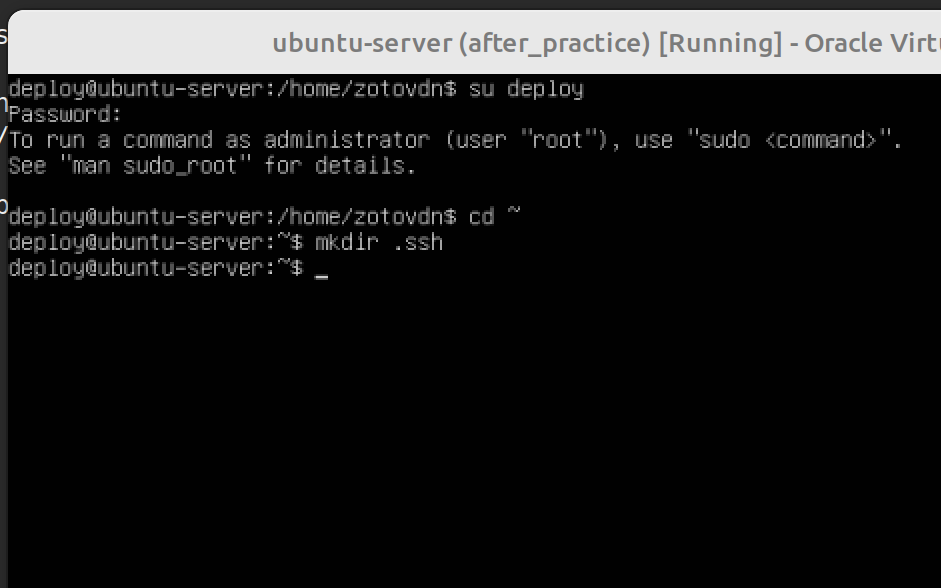
#Добовляем пользователя в группу sudo

sudo usermod -aG sudo deploy



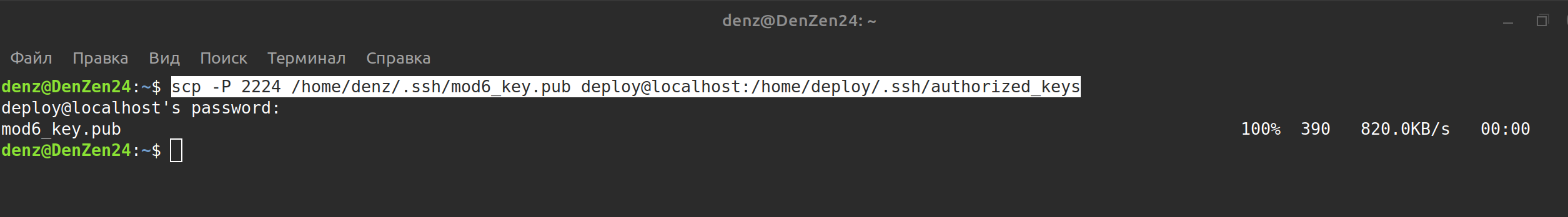
Зайдем под пользователем deploy и создадим папку .ssh в его домашнем каталоге

su deploy; mkdir ~/.ssh

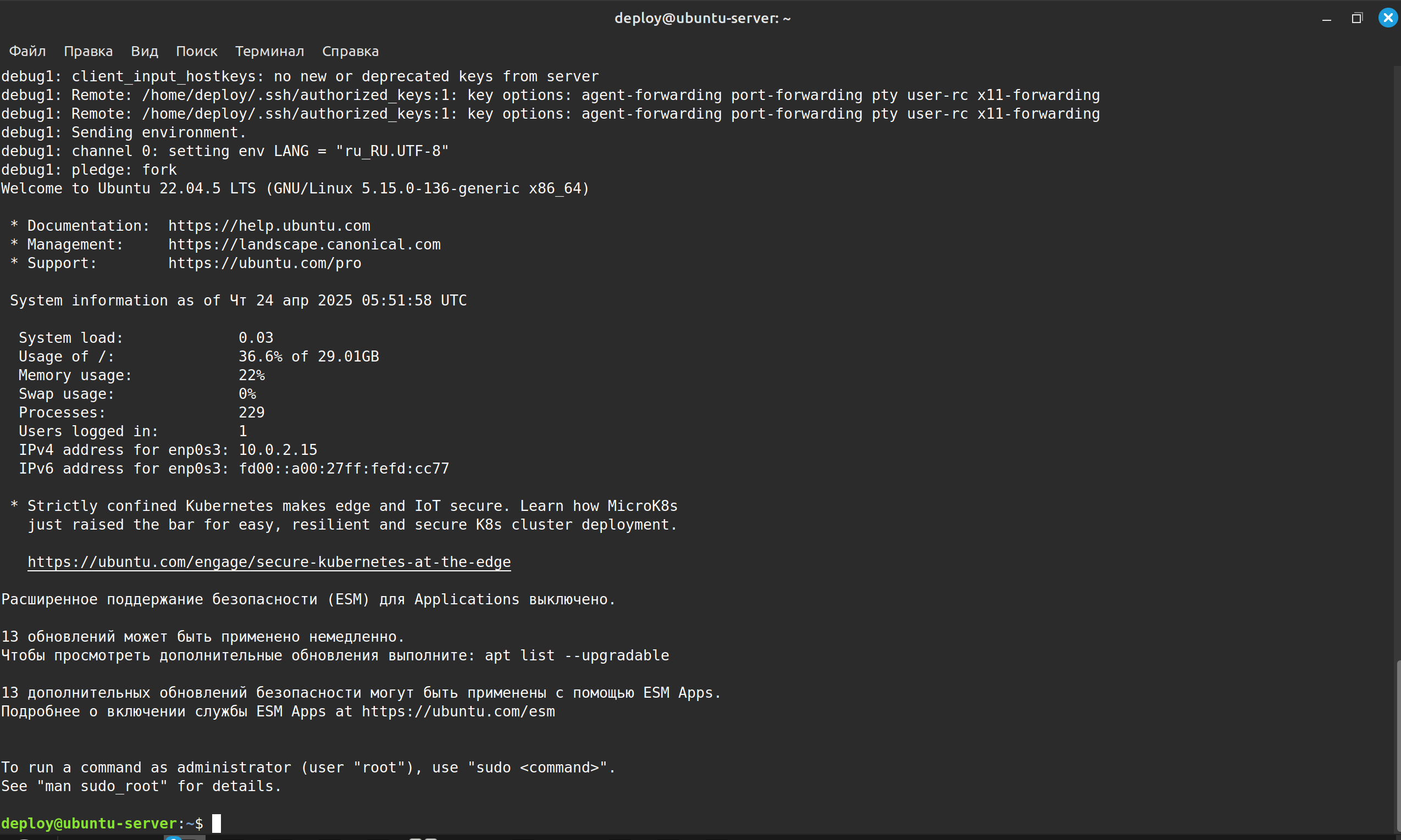


Скопируем публичный ключ на сервер

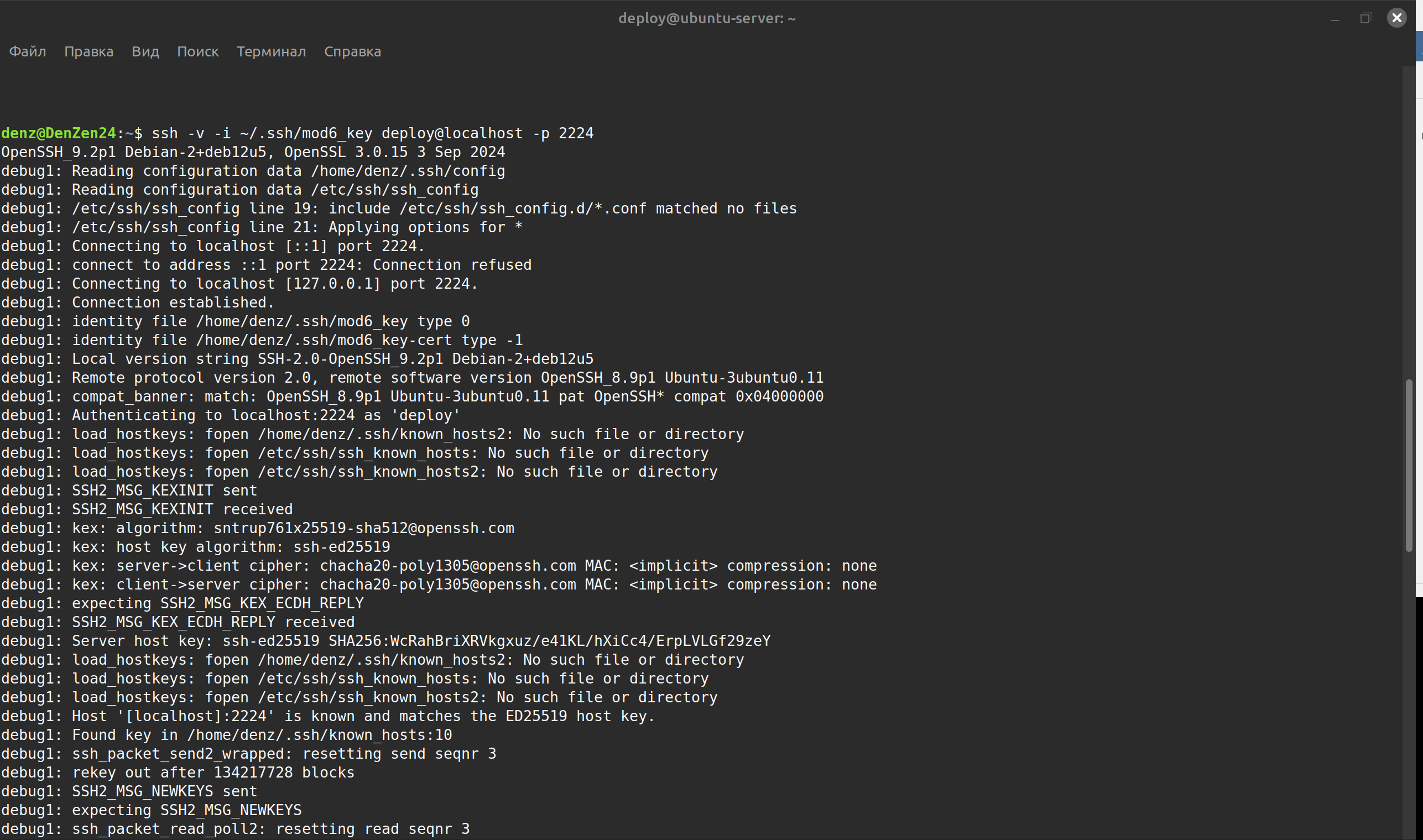
scp -P 2224 /home/denz/.ssh/mod6\_key.pub deploy@localhost:/home/deploy/.ssh/authorized\_keys



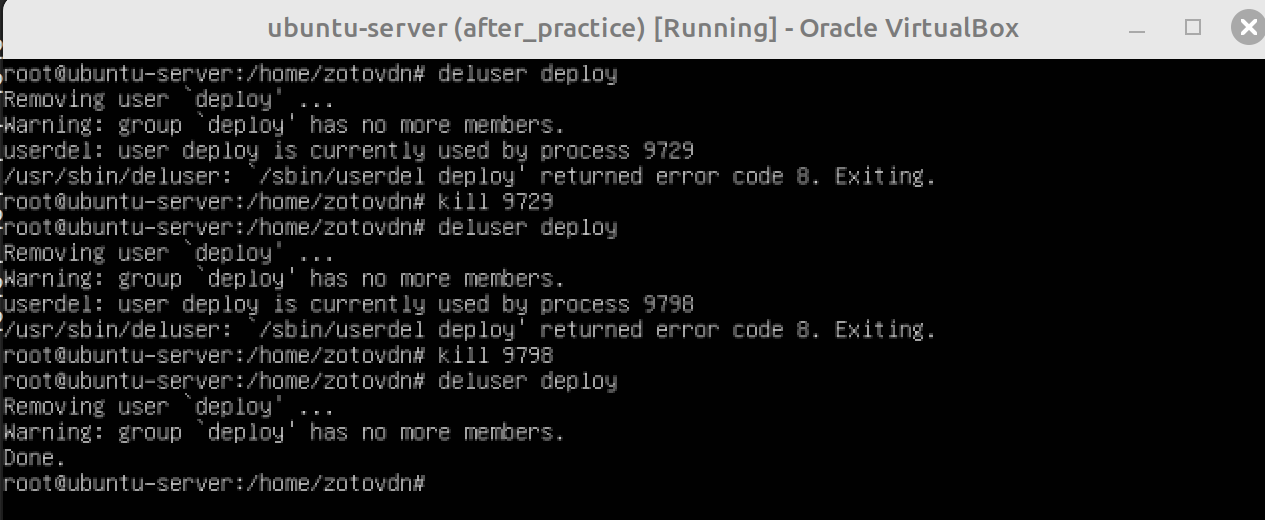
1. Сделайте так, чтобы войти под новым пользователем можно было командой SSH по ключу, созданному в пункте 1.



1. Подключитесь по ssh с опцией -v и с помощью созданного ключа без ввода пароля. Сделайте скриншот с логом подключения.



1. Удалите нового пользователя с помощью команды deluser.



**Задача 2. Конфигурация SSH-подключения**

**Что нужно сделать**

Довольно неудобно каждый раз добавлять путь к ключу при подключении к серверу. Да и указывать его в rsync тоже немного неприятно.

Давайте это исправим.

1. На локальной машине создайте папку ~/.ssh и задайте на неё права:

$ mkdir -p ~/.ssh  
$ chmod 700 ~/.ssh

1. Создайте в папке ~/.ssh файл с именем config:

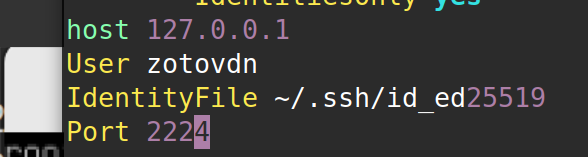
$ touch ~/.ssh/config && chmod 600 ~/.ssh/config

1. Скопируйте приватный и публичный ключи в папку ~/.ssh/:

$ cp -fv timeweb timeweb.pub ~/.ssh/  
$ chmod 600 ~/.ssh/timeweb\*

1. Отредактируйте файл ~/.ssh/config, написав следующие строки:

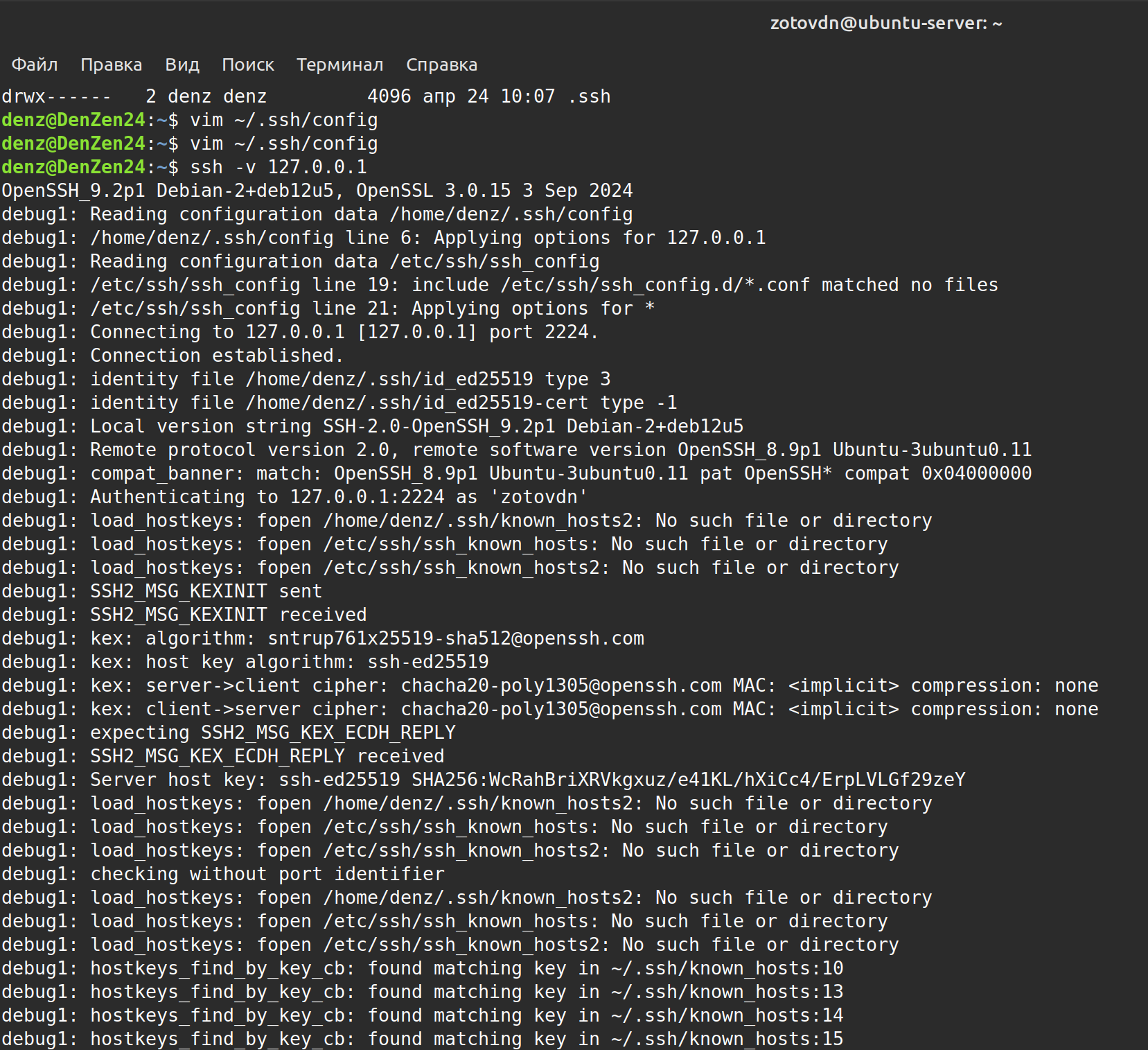
Host <ip\_address\_of\_your\_machine>  
User <your\_remote\_system\_user\_name>  
IdentityFile ~/.ssh/timeweb



1. Подключитесь по ssh с опцией -v, сделав скриншот с логом подключения:

$ ssh <your\_remote\_system\_user\_name>@<ip\_address\_of\_your\_machine>

В качестве решения приложите скриншот, сделанный в пункте 5.

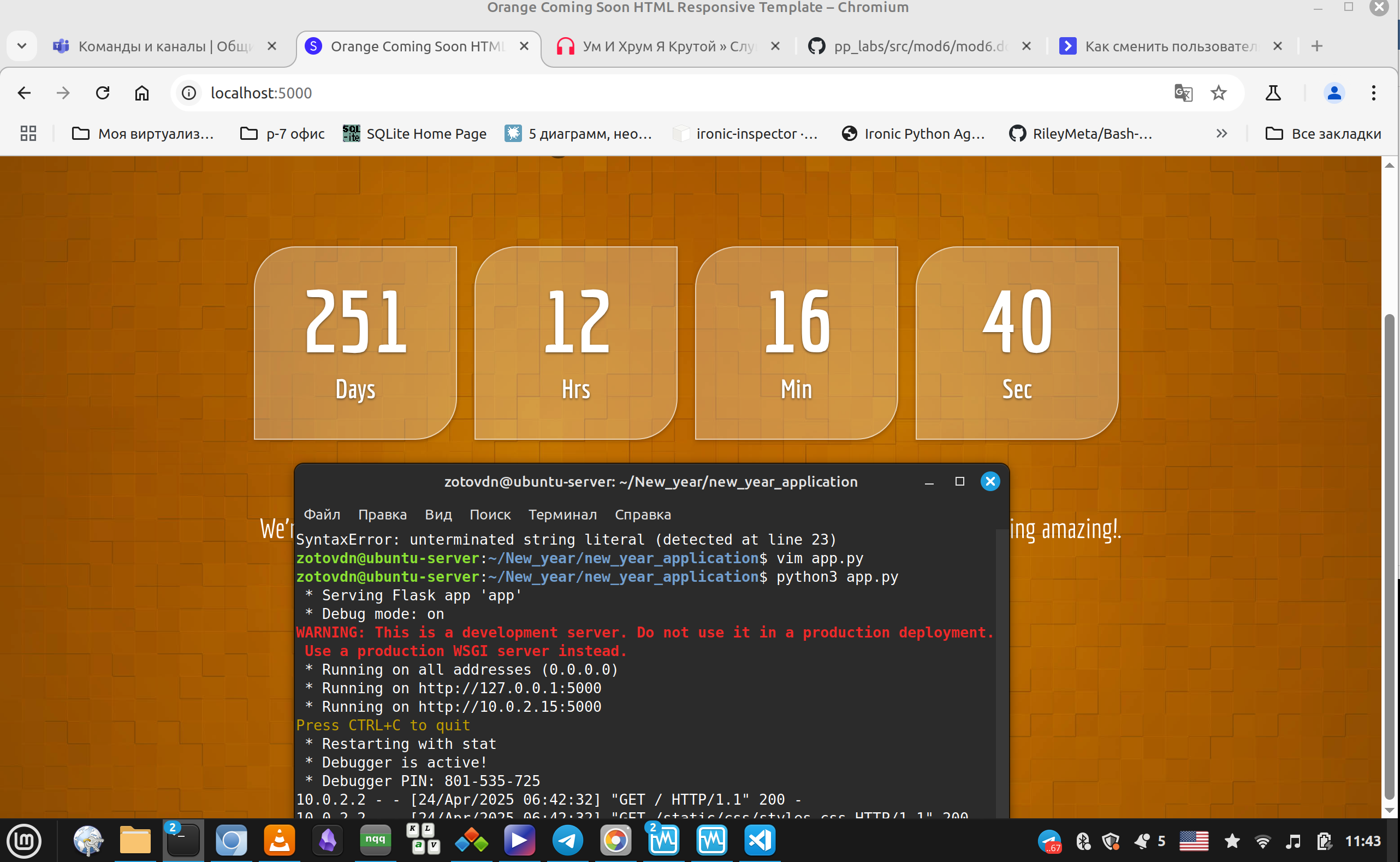


**Задача 3. Деплой сайта**

**Что нужно сделать**

Запускать сайт на localhost, конечно, интересно, но ещё лучше, когда этот сайт может увидеть каждый, подключившись к нему удалённо через браузер.

Сделайте деплой сайта, который находится в материалах к занятию в папке *new\_year\_application*.



**Задача 4. Let it snow!**

**Что нужно сделать**

Мы успешно задеплоили сайт, который показывает количество дней до Нового года. Давайте сделаем его красивее!

Добавьте на сайт снежинки, падающие сверху страницы, которые реализованы [в этой библиотеке](https://github.com/scottschiller/snowstorm/) .

Инструкция:

1. Скачайте в локальную папку /js/ скрипт со снежинками snowstorm.js.
2. Подключите этот файл в index.html, указав сразу после <!--Scripts–> новый JS-файл:

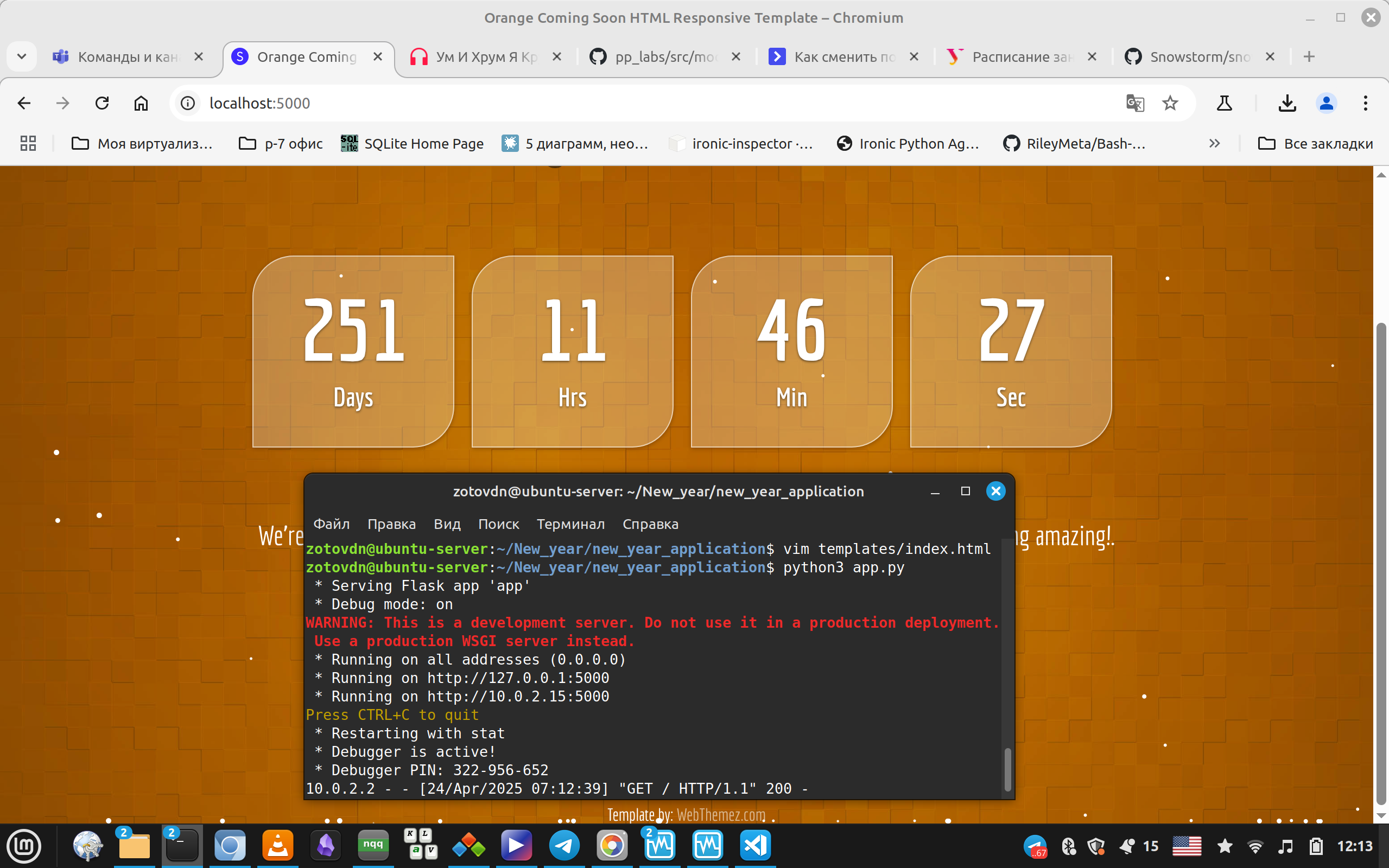
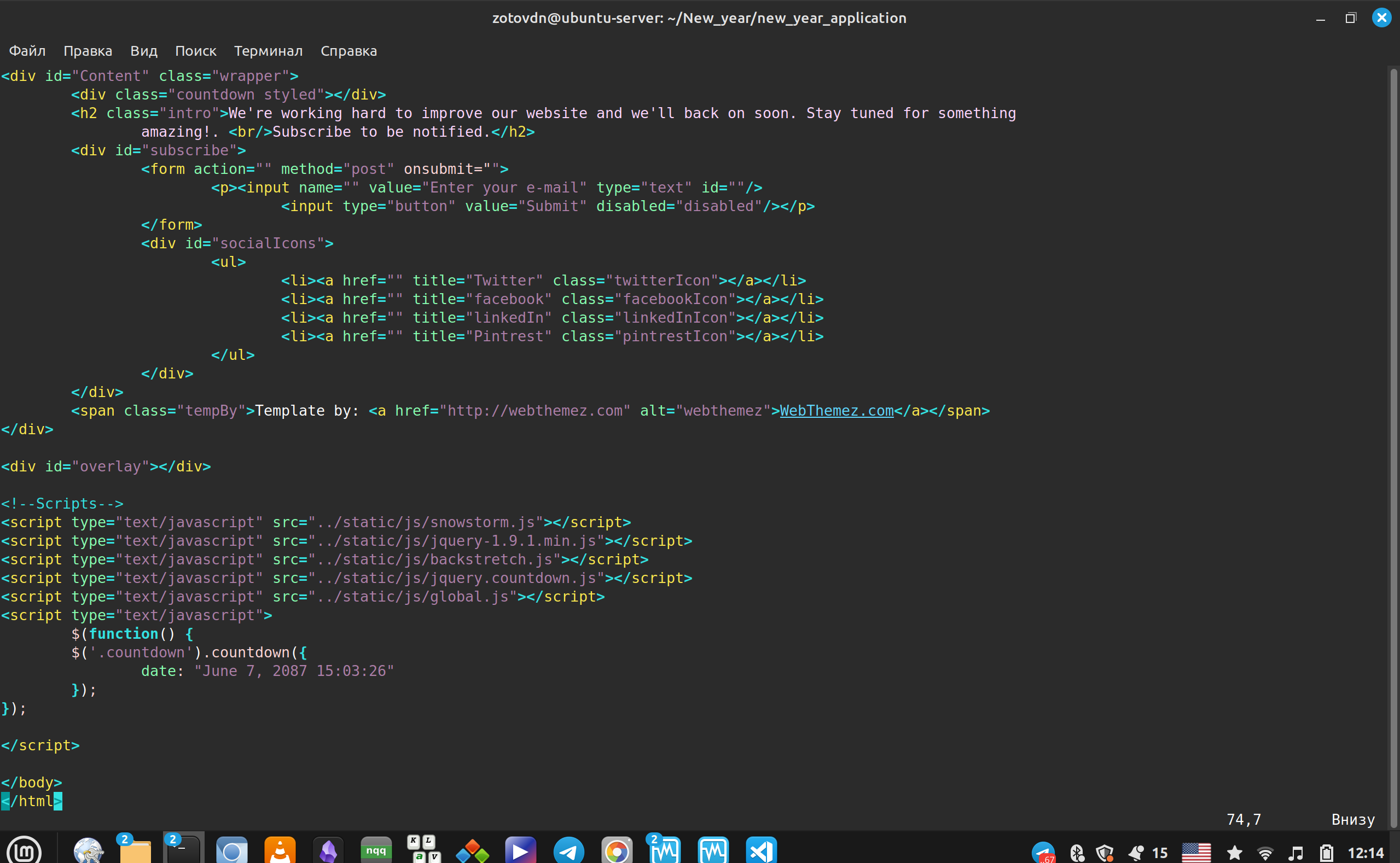
<script type="text/javascript" src="/js/snowstorm.js"></script>

1. Убедитесь, что на локальном сайте появились снежинки.
2. Задеплойте сайт на удалённый сервер. Убедитесь, что всё работает.

В качестве решения приложите скриншот с работающим удалённым сайтом.

**Что оценивается**

* На удалённом сайте появились снежинки.
* На скриншоте виден URL страницы.



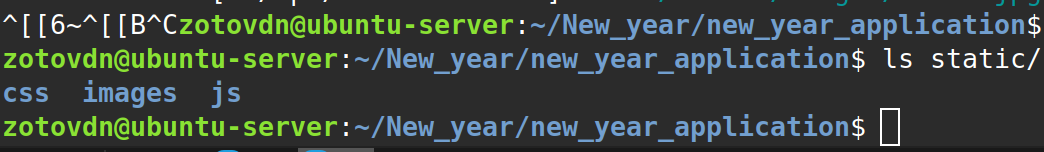
**Задача 5. Раздача статики**

**Что нужно сделать**

Создавать endpoint под каждую папку со статическими файлами непрактично. Обычно делают так, чтобы по URL /static можно было получить любой из них, будь то картинка, скрипт или таблица стилей.

Проведём эту операцию над нашим сайтом:

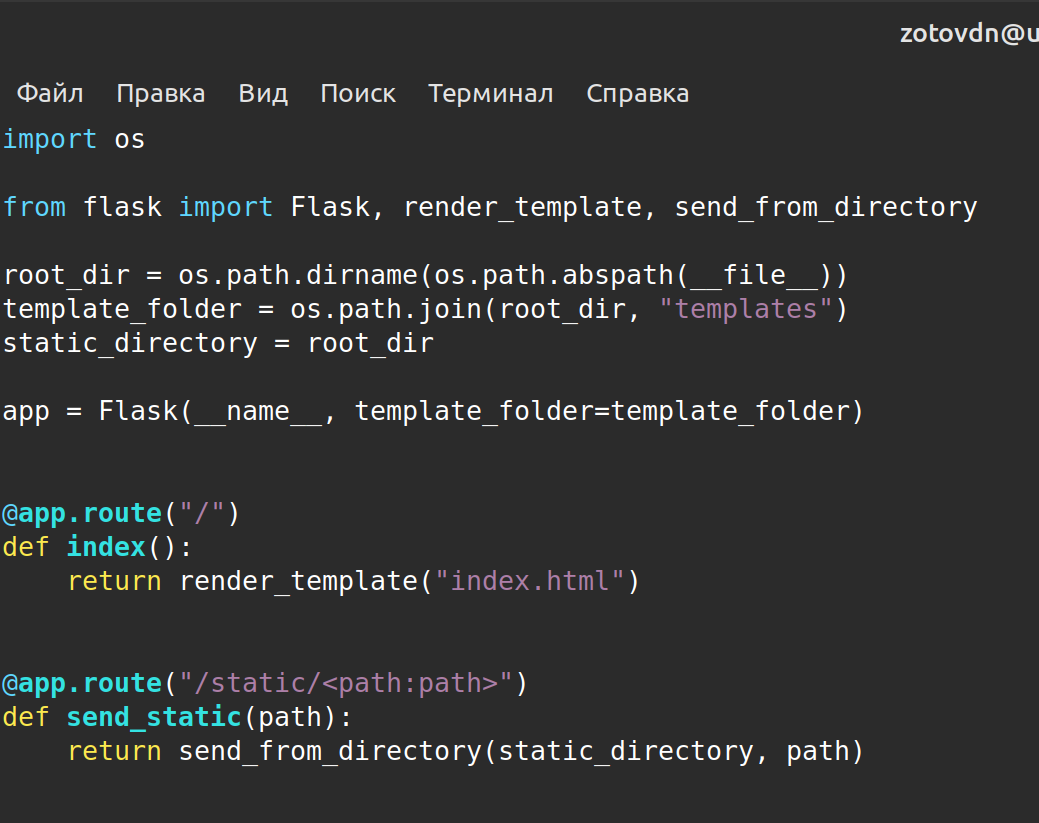
1. В папке с приложением создайте папку /static.
2. Перенесите в неё папки /css, /js и /images.



1. Измените url до статических файлов в index.html, добавив в начало каждого url /static.



1. Удалите endpoint'ы send\_css, send\_js и send\_images, заменив их на самореализованный endpoint send\_static (он должен уметь обрабатывать URL, начинающийся со /static).

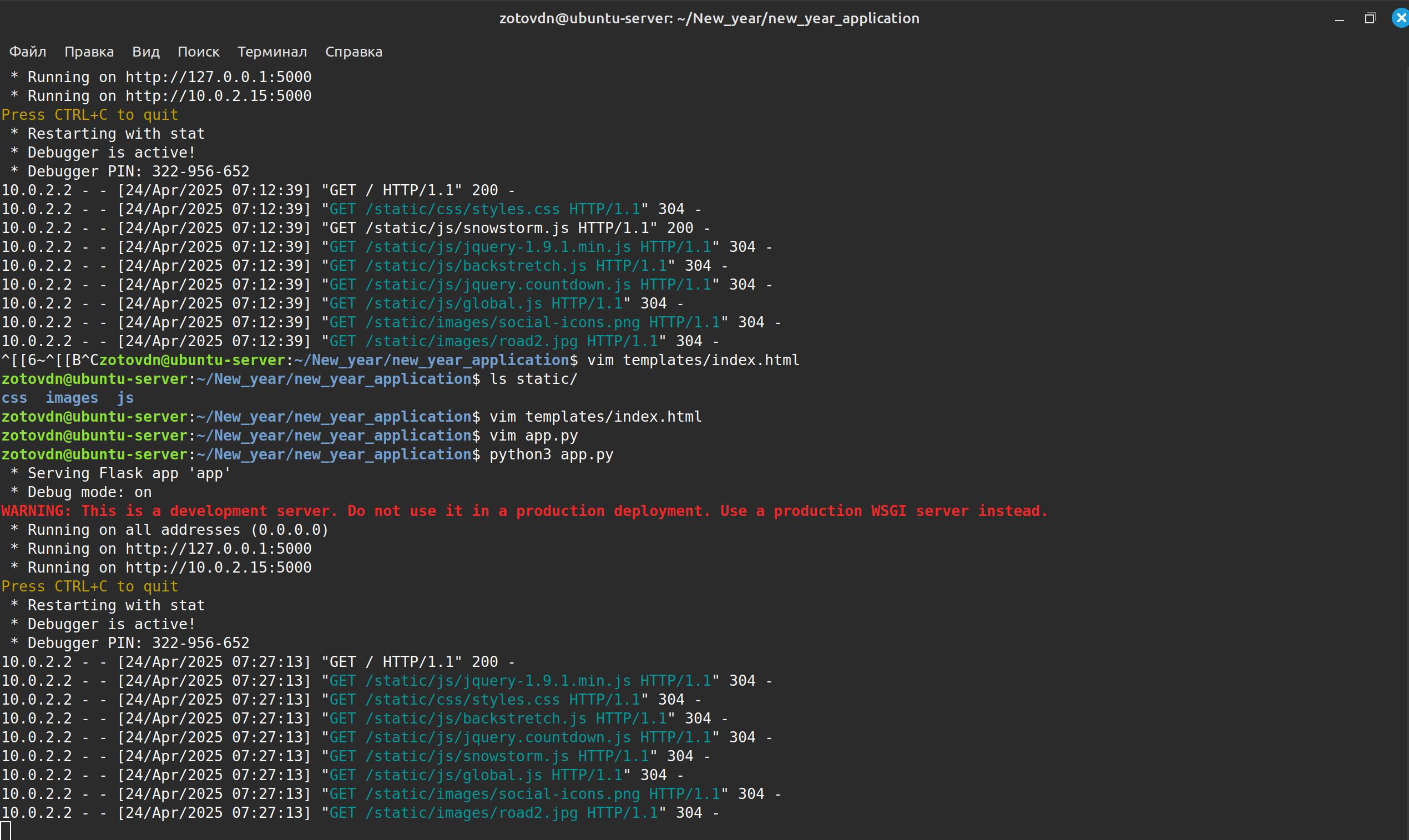


1. Убедитесь, что на локальном сайте всё работает.
2. Задеплойте сайт на удалённый сервер. Убедитесь, что всё работает.

В качестве решения приложите скриншот, показывающий, что запросы сайта на статику посылаются на URL /static. Сделать это можно двумя способами:

1. Показать логи сервера. При подключении к сайту должно выводиться что-то подобное:

"GET /static/css/styles.css HTTP/1.1" 200 –



1. Показать URL запросов в инструментах разработчика. Запросы к серверу находятся во вкладке «Сеть/Network». Далее нужно поставить фильтр «Все/All», нажать на какой-либо запрос и посмотреть Headers.

GET http://<YOUR\_SITE>/static/js/jquery-1.9.1.min.js

