Тестовый стенд – локальный сервер с kali linux. Подключение к серверу происходит через ssh по белому ip адресу.

Техническое задание:

Установить на локальную машину docker engine, docker-compose

Создать файл композа с двумя сервисами: nginx и couchdb. Версия nginx не имеет значения, couchdb должен быть версии 1.7.2.

Контейнер couchdb должен проксировать порт 5984->5984

Прописать в системный файл hosts доменное имя couchdb.local, после чего создать конфигурацию nginx для того, чтобы слушая 80 порт http, по url http://couchdb.local можно было попасть в сервис, развернутый на порту 5984.

Далее требуется создать bash-скрипт для автоматической смены пароля в couchdb на уже созданного пользователя.

Например есть пользователь org.couchdb.user:test с паролем 1234, необходимо изменить его пароль на 4321.

Учетную запись и пароль администратора скрипт должен брать из переменных окружения.

В качестве входных параметров скрипт должен принимать имя пользователя (из примера "test"), новый пароль для пользователя, мастер пароль для скрипта.

Результатом тестового задания должны быть скриншоты команды docker ps, файлы конфигурации docker-compose.yml, файл конфигурации сервера nginx с описанным сервером couchdb.local и файл скрипта из последней части задания.

Задачи:

1. Установка docker engine и docker compose
2. Написание yaml файла композа для запуска контейнеров с nginx любой версии и c couchdb версии 1.7.2

Оглавление

[1 Установка docker engine и docker compose 3](#_Toc204678123)

[2 Что такое couchdb 4](#_Toc204678124)

[3 Клонирование репозитория с github 4](#_Toc204678125)

[3 Написание нужного файла yaml compose и запуск контейнеров 4](#_Toc204678126)

[4 Написание Bash-скрипта для смены паролей. 10](#_Toc204678127)

## 1 Установка docker engine и docker compose

На моем локальном сервере уже установлены и функционируют docker engine и docker compose. Проверка установленных версий представлена на рисунке 1. Проверка запущенных docker контейнеров представлена на рисунке 2.

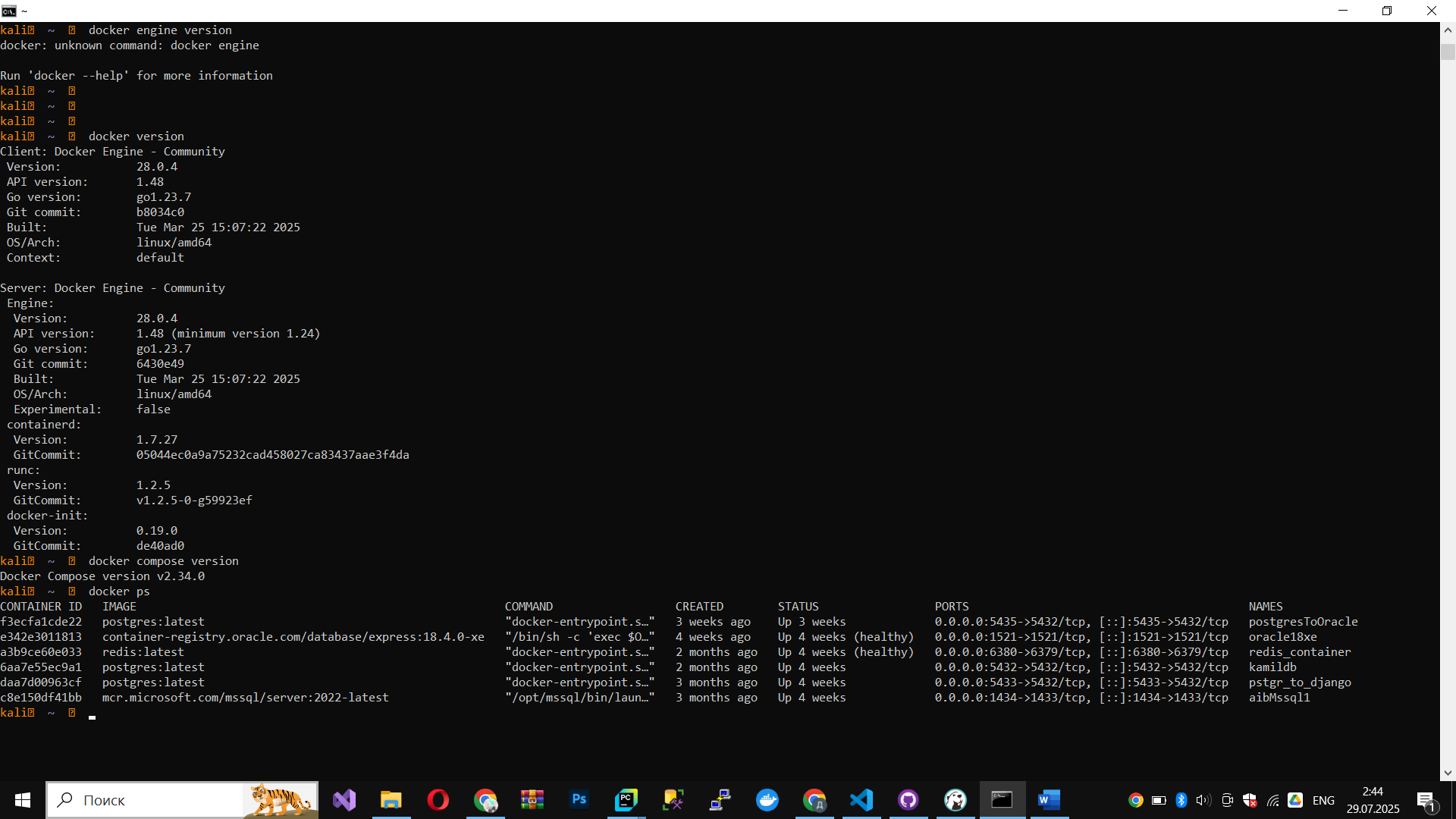


Рисунок 1 – Проверка установки docker engine и docker compose.

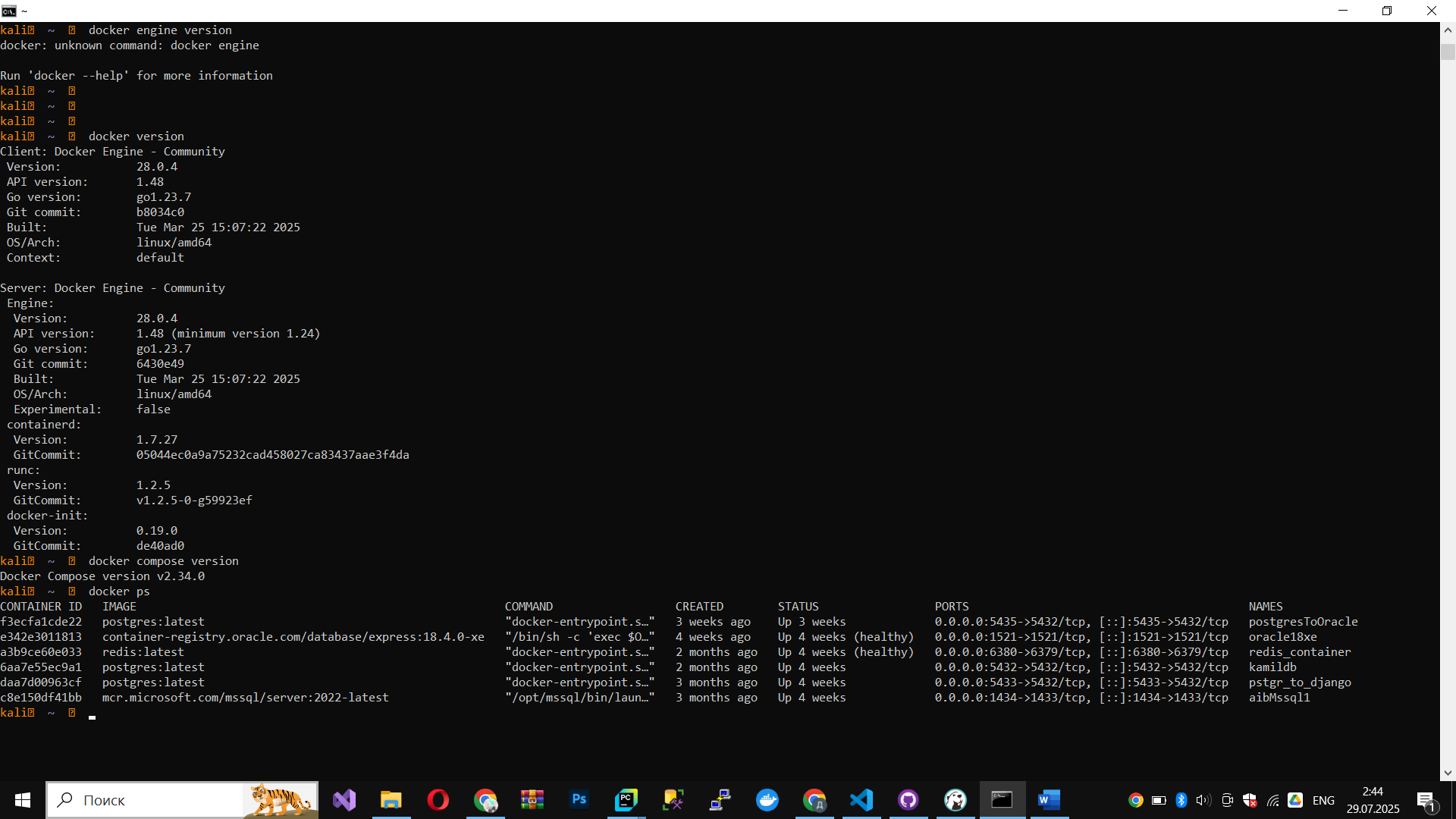


Рисунок 2 – Проверка запущенных docker контейнеров

В случае, если программы ранее не были установлены, их можно установить с помощью команд:

Для установки docker engine - «sudo apt-get update

sudo apt-get install docker.io

sudo systemctl enable --now docker»

Для установки docker compose: «sudo curl -L "https://github.com/docker/compose/releases/latest/download/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose

sudo chmod +x /usr/local/bin/docker-compose»

## 2 Что такое couchdb

Ранее я не имел опыт взаимодействия с этой программой, но судя по приписке “db”, это должна быть СУБД.

Apache CouchDB – свободно распространяемая СУБД с открытым исходным кодом. Тип – NoSql, документно-ориентированная. По описанию из сети, наводит на мысль о схожести с MongoDB. СУБД с низкой популярностью. Хранит данные в формате JSON. Не требует жесткой схемы данных – это позволяет работать с данными разнообразной структуры. Взаимодействие с данной СУБД происходит через Http запросы, из чего следует высокое удобство в веб-разработке.

Исходя из описания, можно выделить значительное преимущество данной СУБД – способность работать в условиях перебоев сети.

Вывод: СУБД, удобная в работе в тех приложений, в которых требуется периодическая с синхронизация данных, как к примеру мобильные приложения.

## 3 Клонирование репозитория с github

Для работы с этим проектом, я создал репозиторий github <https://github.com/Danil-512/Test-task-Devops>

Задача – клонировать его в директорию /home/danil/Desktop/

## 3 Написание нужного файла yaml compose и запуск контейнеров

Для хранения проекта и файла конфигурации yaml, создам директорию на рабочем столе kali. В нее клонирую репозиторий с Github. Клонирование репозитория представлено на рисунке 3.

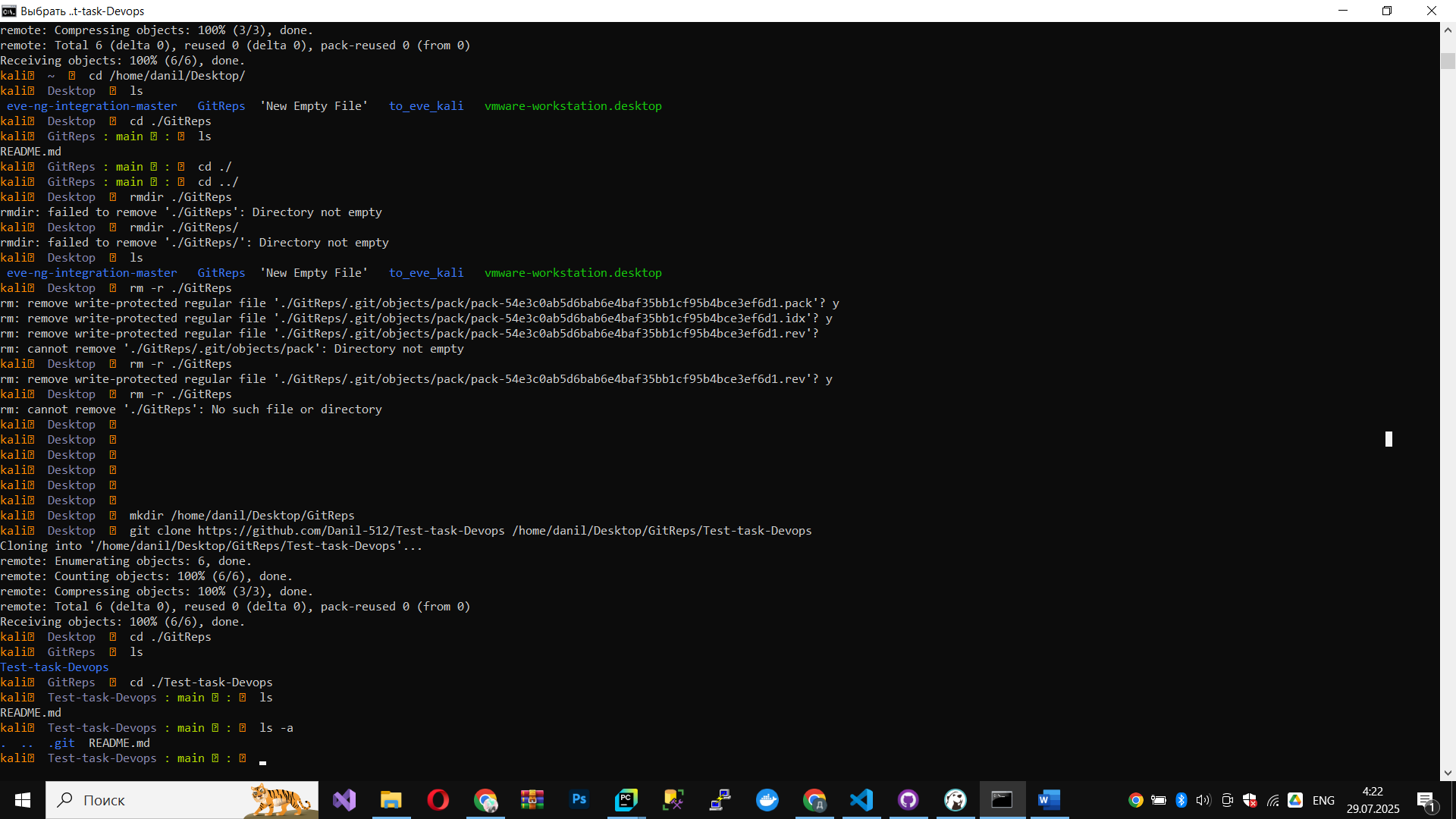


Рисунок 3 – Клонирование репозитория с Github

Далее необходимо создать файл конфигурации docker compose в формате yaml. Имя файла - docker-compose.yaml. Создание файла представлено на рисунке 4.

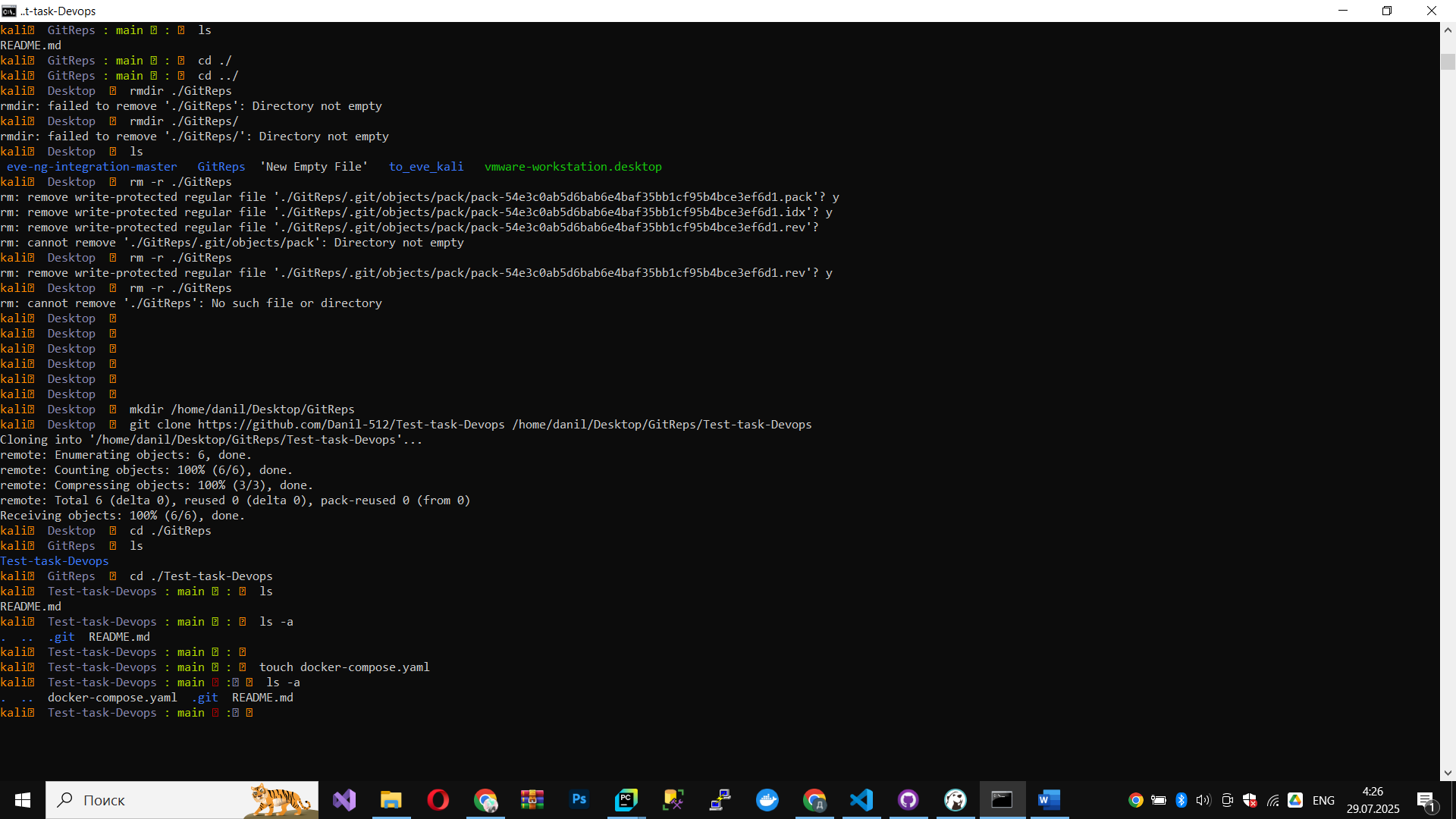


Рисунок 4 – Создание файла docker-compose.yaml

Файл конфигурации docker-compose представлен на рисунке 5 с подробным описанием каждой строки.

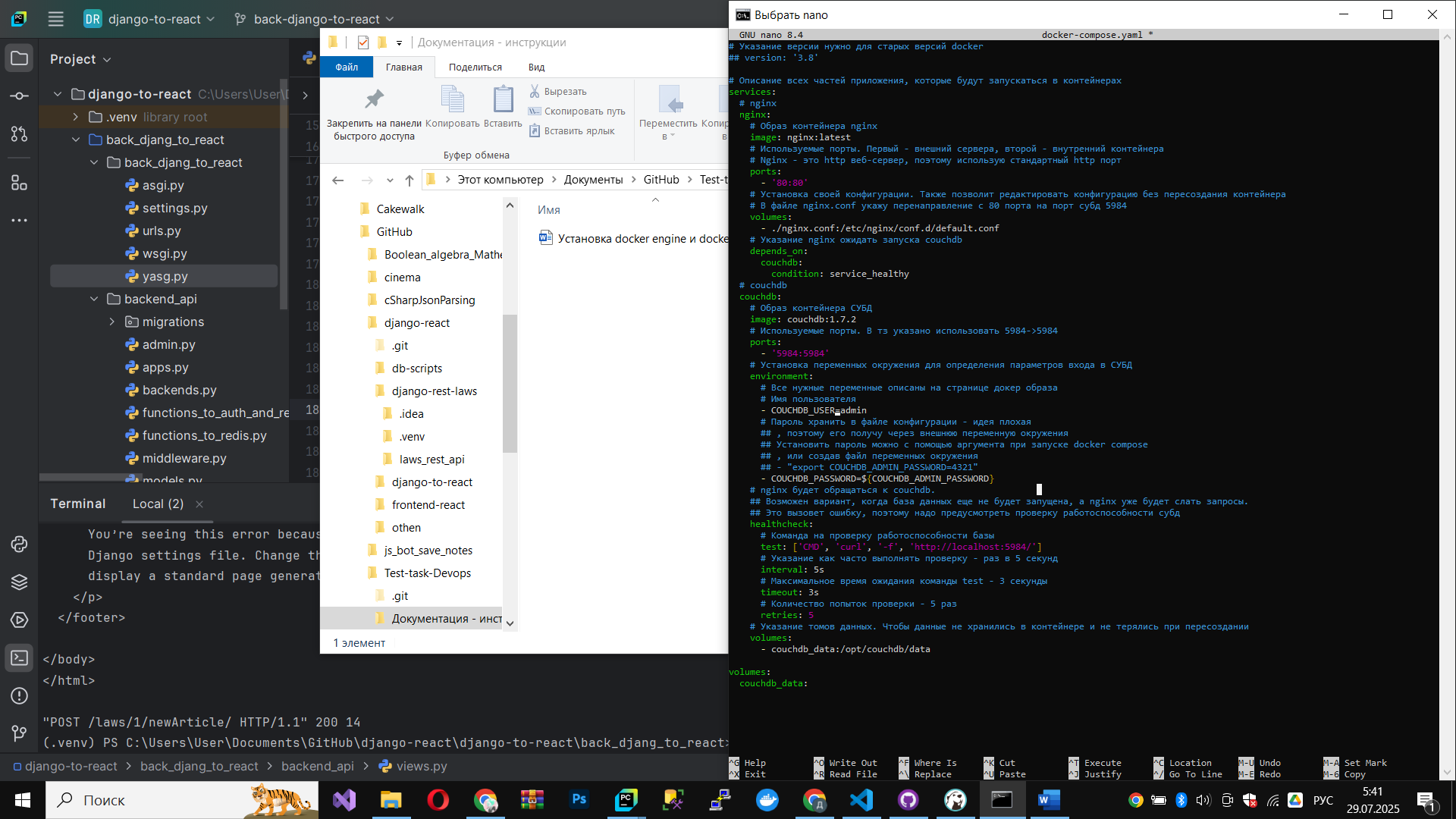


Рисунок 5 – файл docker-compose

Создание переменной окружения для пароля, проверка файла конфигурации compose и создание файла конфигурации nginx.conf представлено на рисунке 6

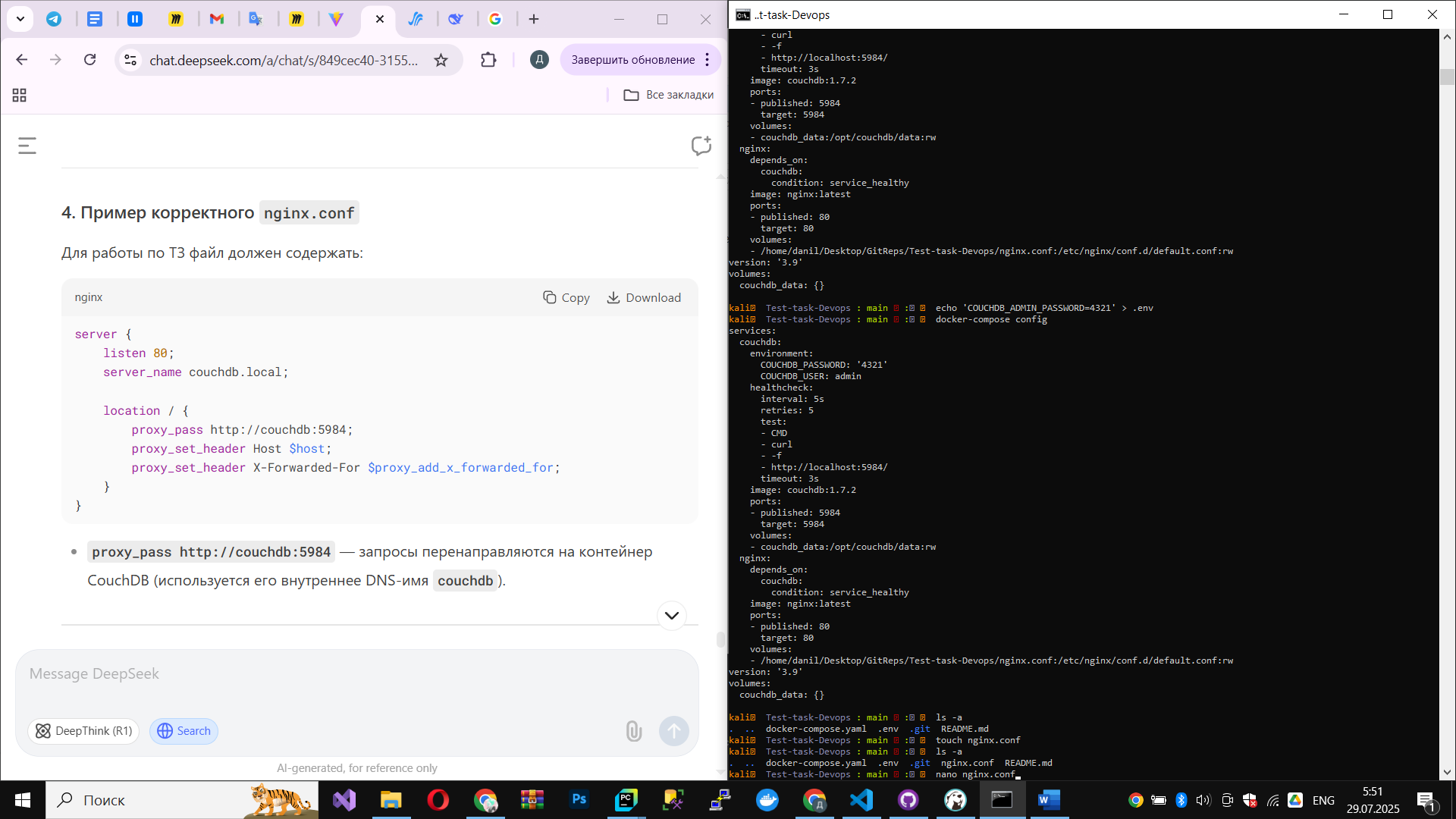


Рисунок 6 – Проверка конфигурации, создание переменной окружения

Далее необходимо настроить конфигурацию nginx в файле nginx.conf для перенаправления от порта 80 к порту 5984. Настройка nginx.conf представлена на рисунке 7.

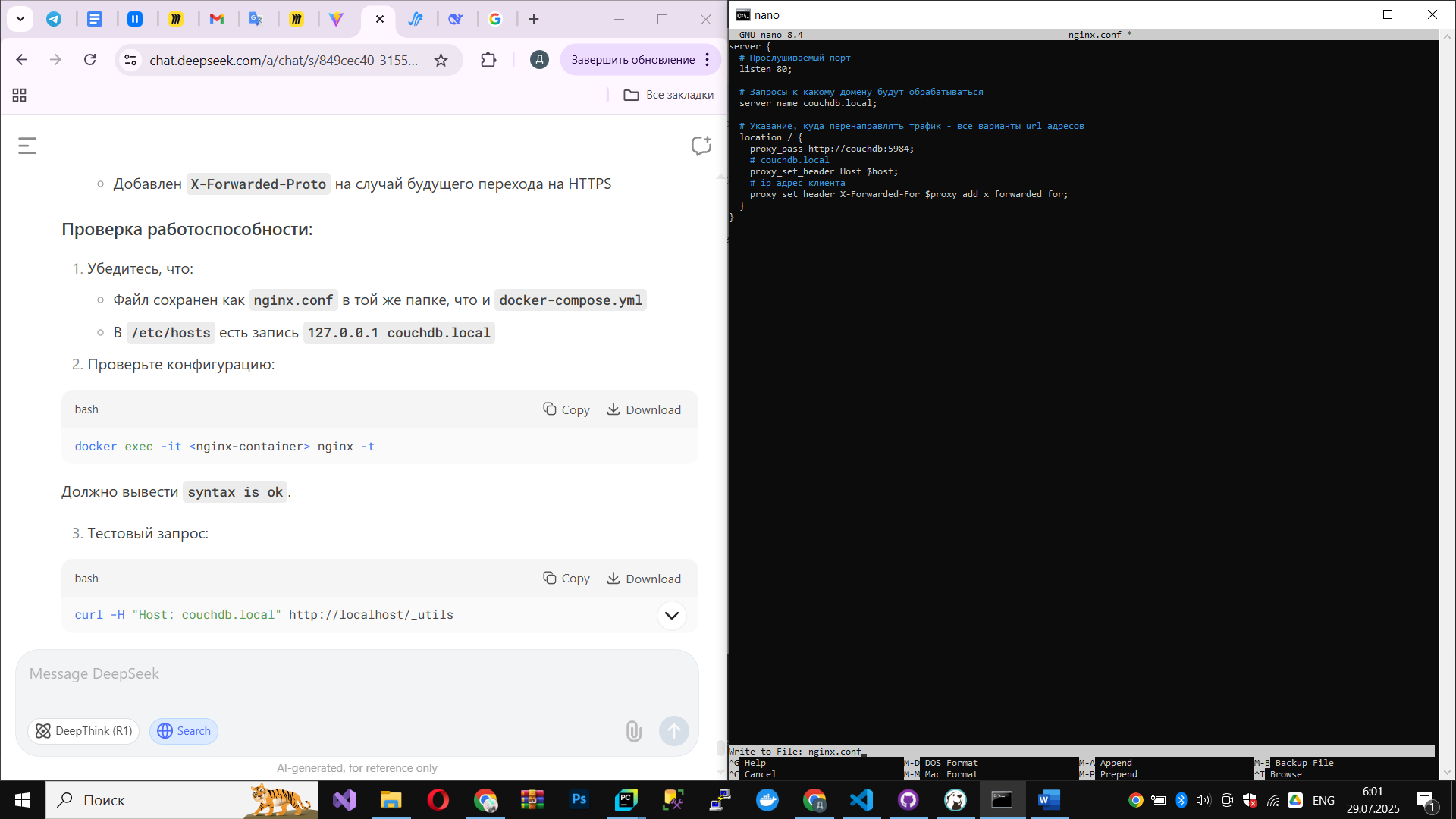


Рисунок 7 – Файл конфигурации nginx.conf

Далее на сервере необходимо прописать в системный файл hosts доменное имя couchdb.local. Это требуется по тз и без этой настройки, адрес couchdb.local будет запрашиваться из внешней сети. Данная настройка представлена на рисунке 8.

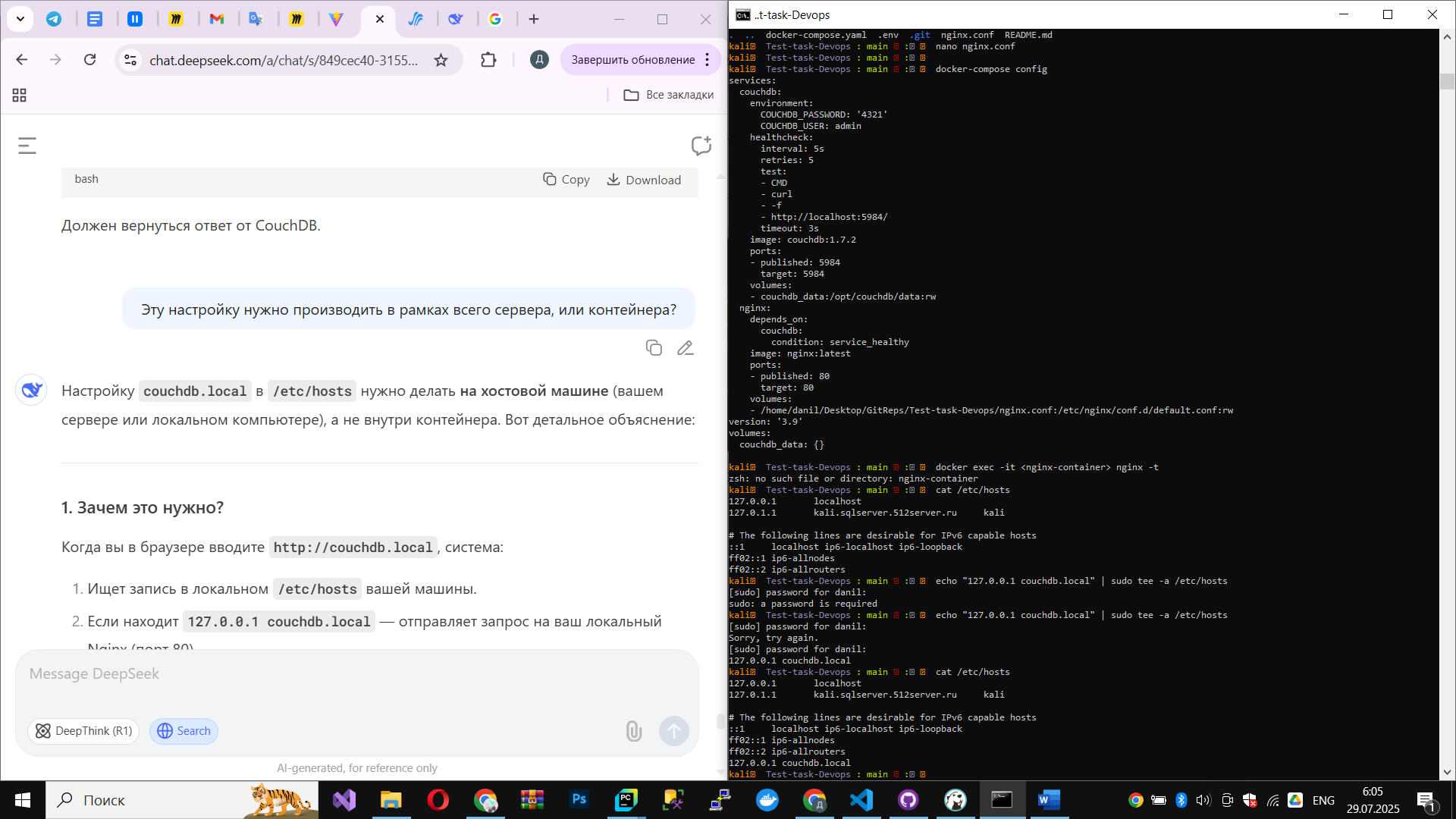


Рисунок 8 – Добавление доменного имени в hosts

Все настройки произведены, можно запускать контейнеры. Запуск происходит с помощью команды «docker compose up --build». Запуск представлен на рисунке 9.

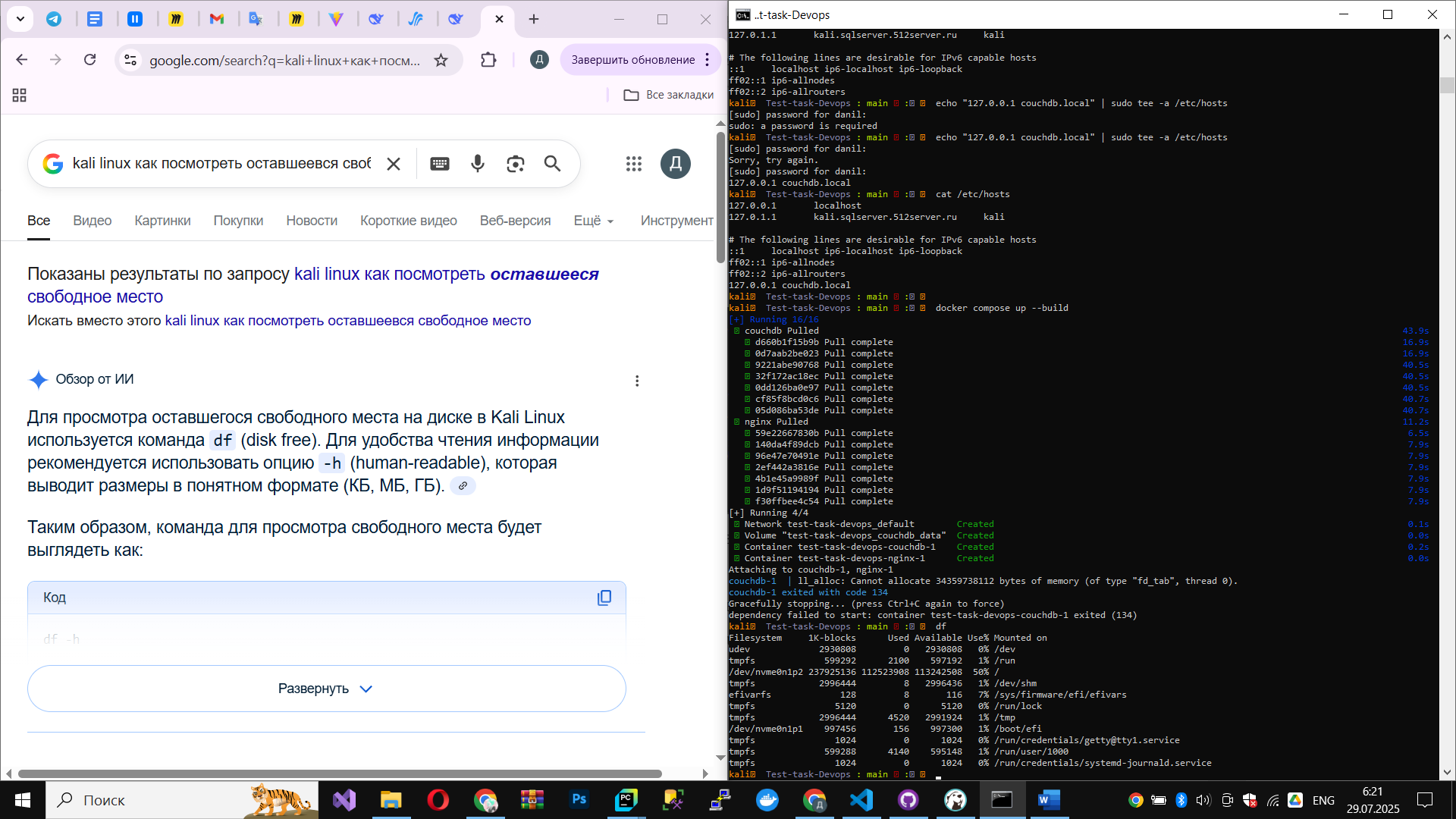


Рисунок 9 – Неудачный запуск контейнеров

Запуск не удался. Была получена ошибка о недостатке места. CouchDb запрашивает 32 гб свободного места, сервер такое пространство предоставить не может. Выход – указать ограничение места в файле конфигурации docker compose. Измененный файл конфигурации представлен на рисунке 10.

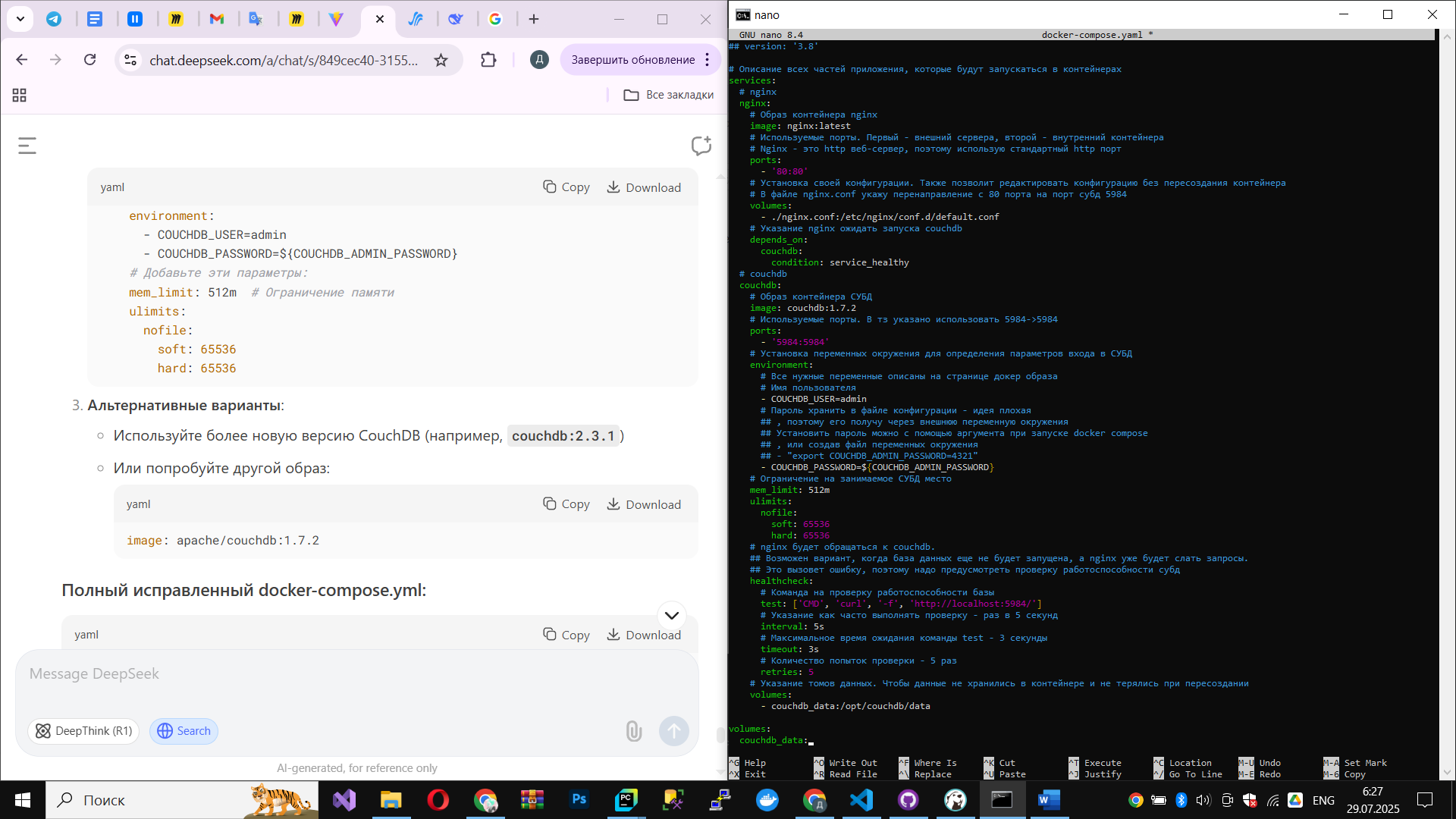


Рисунок 10 – Измененный файл конфигурации с ограничением места для СУБД

Далее с помощью запуска сборки контейнеров через «docker-compose down -v && docker-compose up --build», получилось успешно запустить docker-compose. Работа контейнеров представлена на рисунке 11.

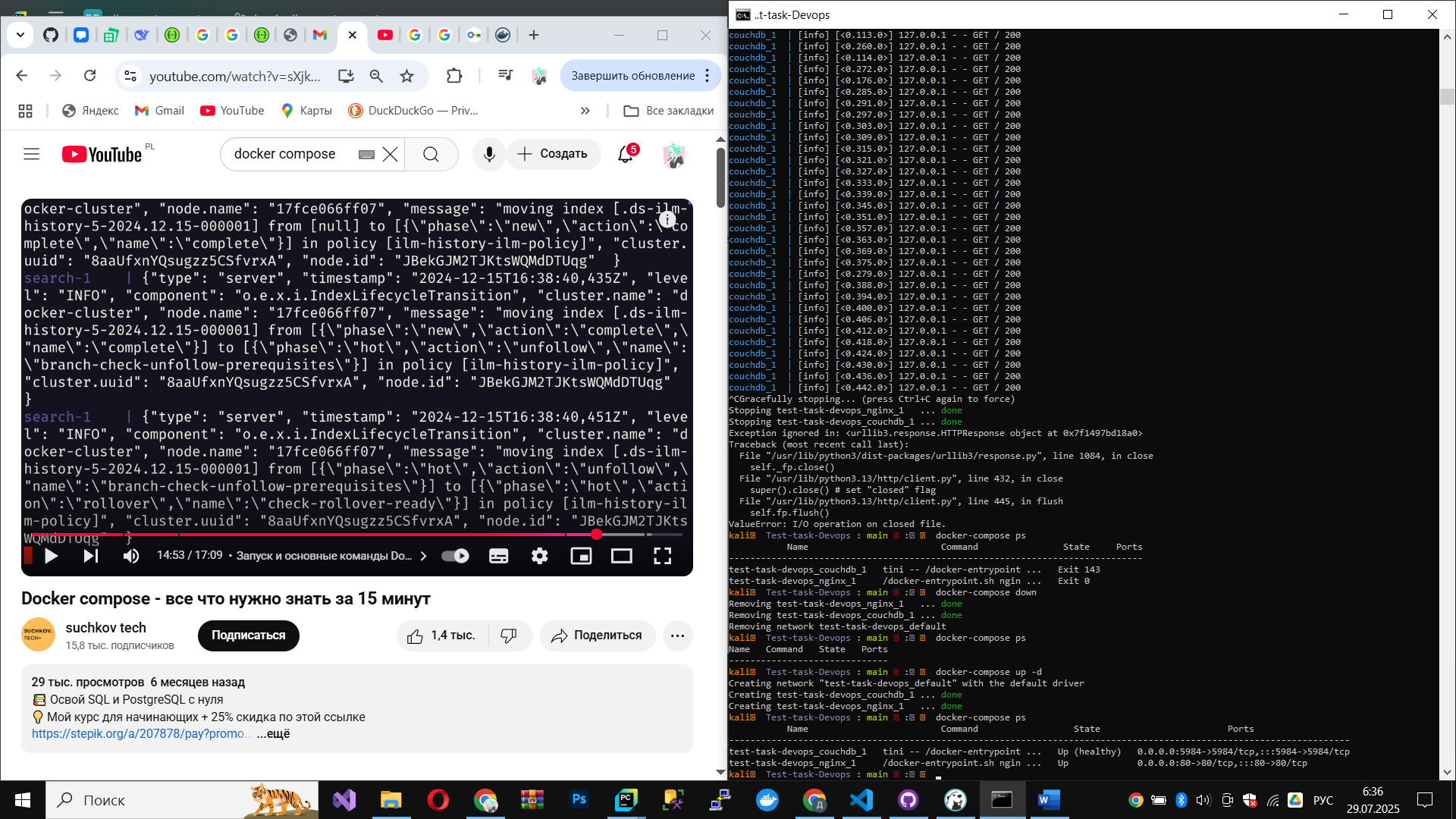


Рисунок 11 – Успешно запущенные контейнеры.

Проверка обращения по адресу couchdb.local представлена на рисунке 12.

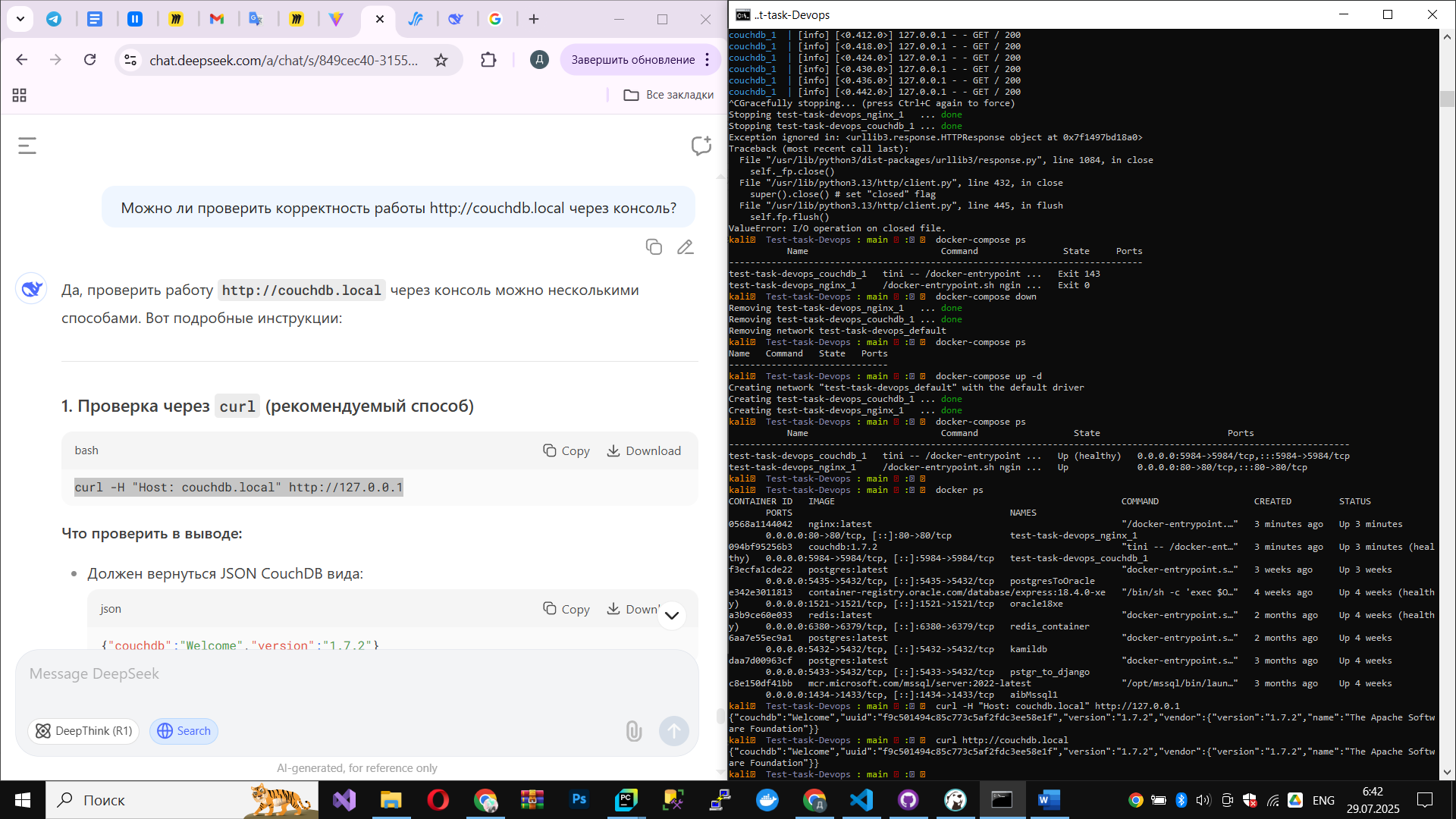


Рисунок 12 – Проверка работы СУБД и nginx

Исходя из консольного вывода на рисунке 12, можно сделать вывод, что контейнеры с couchdb и nginx успешно запущены и работают, nginx перенаправляет трафик с порта 80 на порт 5984 СУБД, СУБД дает корректный ответ.

## 4 Написание Bash-скрипта для смены паролей.

«Учетную запись и пароль администратора скрипт должен брать из переменных окружения» - означает что и логин и пароль администратора должны быть записаны в переменных окружения. А у меня на данный момент в переменных окружения только пароль.

Заменяю строчку «- COUCHDB\_USER=admin» на строчку «- COUCHDB\_USER=${COUCHDB\_ADMIN\_USER}» в docker-compose.yaml и добавляю строку с логином администратора в файл с переменными окружения командой «echo 'COUCHDB\_ADMIN\_USER=admin' > .env». После изменения файла конфигурации и добавления переменной окружения, необходимо пересоздать контейнеры, это действие представлено на рисунке 13

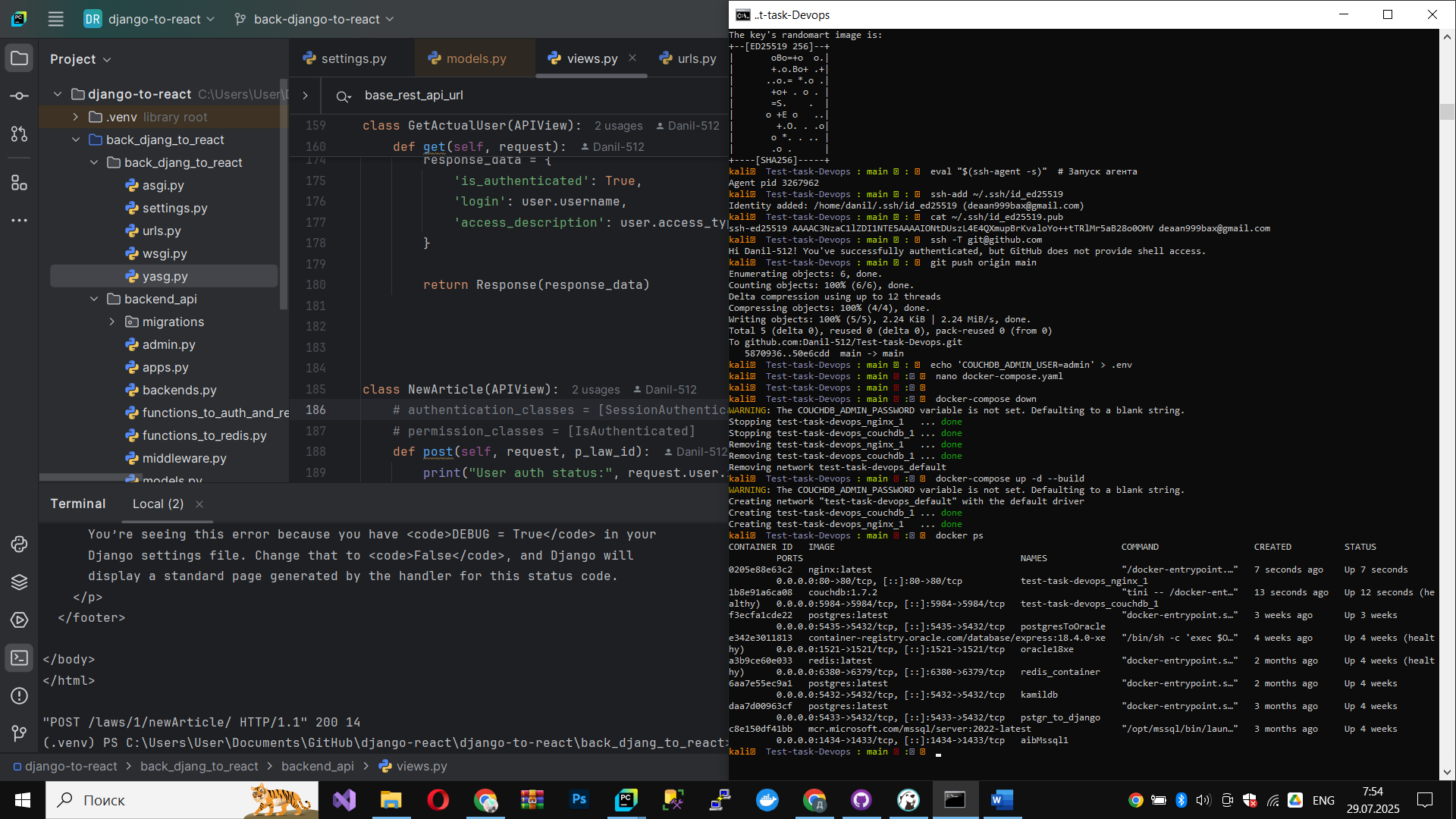


Рисунок 13 – Пересоздание контейнеров после обновления конфигурации

Все переменные окружения на месте, можно приступать к написанию bash-скрипта. Написанный bash скрипт с подробными комментариями представлен на рисунке 14.

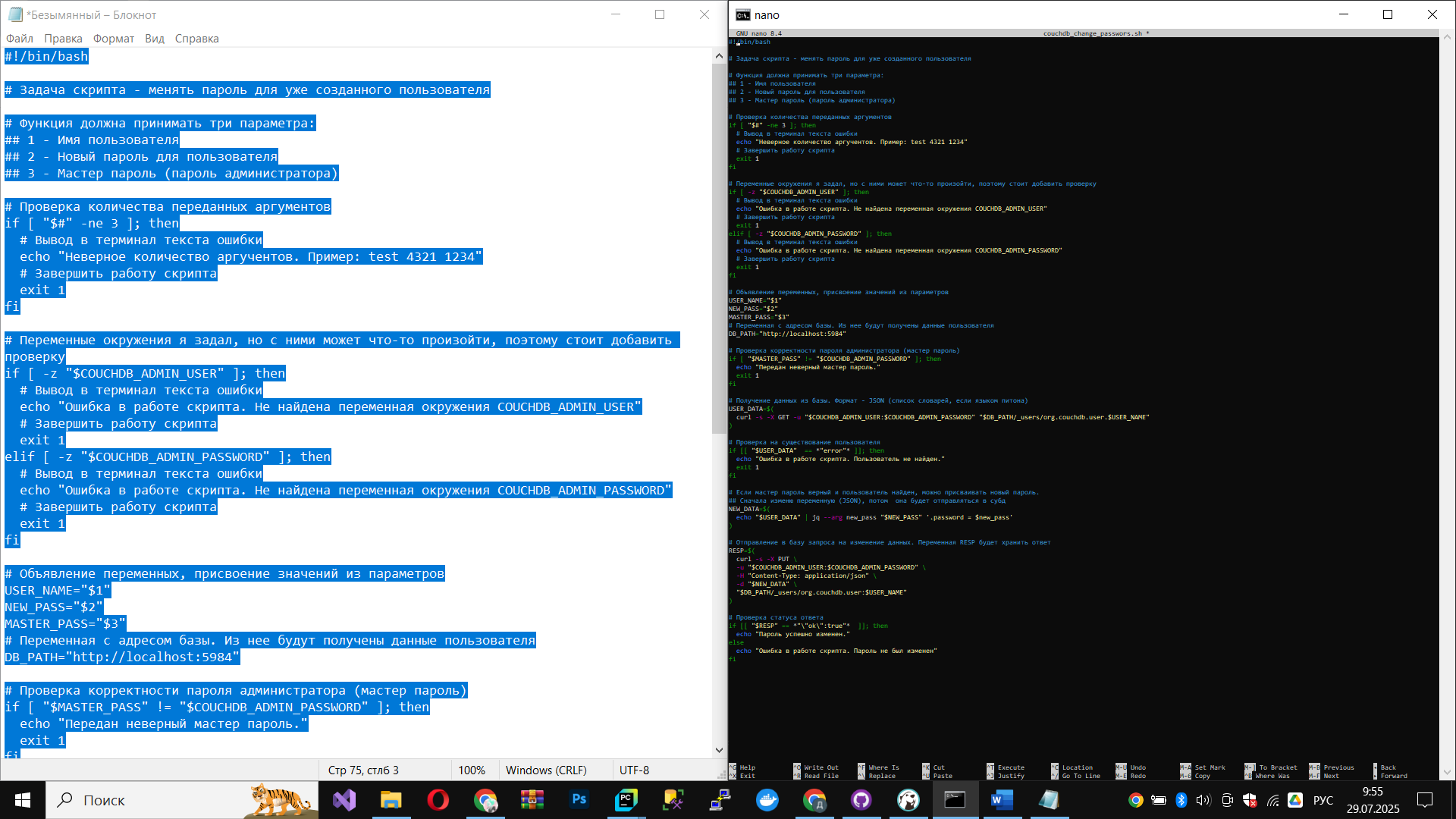


Рисунок 14 – Bash скрипт для изменения пароля пользователя

Чтобы скрипт корректно запускать, его нужно сделать исполняемым с помощью команды «chmod +x couchdb\_change\_password.sh».

Для тестирования скрипта, создам нового пользователя test1. Создание тестового пользователя и тестирование работы скрипта представлено на рисунке 15. При тестировании скрипта столкнулся с проблемой, что не подгружались переменные окружения в скрипте, поэтому добавил в начало скрипта после #bash код «# Загрузка переменных окружения

if [ -f ".env" ]; then

set -o allexport

source .env

set +o allexport

else

echo "Не найден файл с переменными окружения"

exit 1

fi»

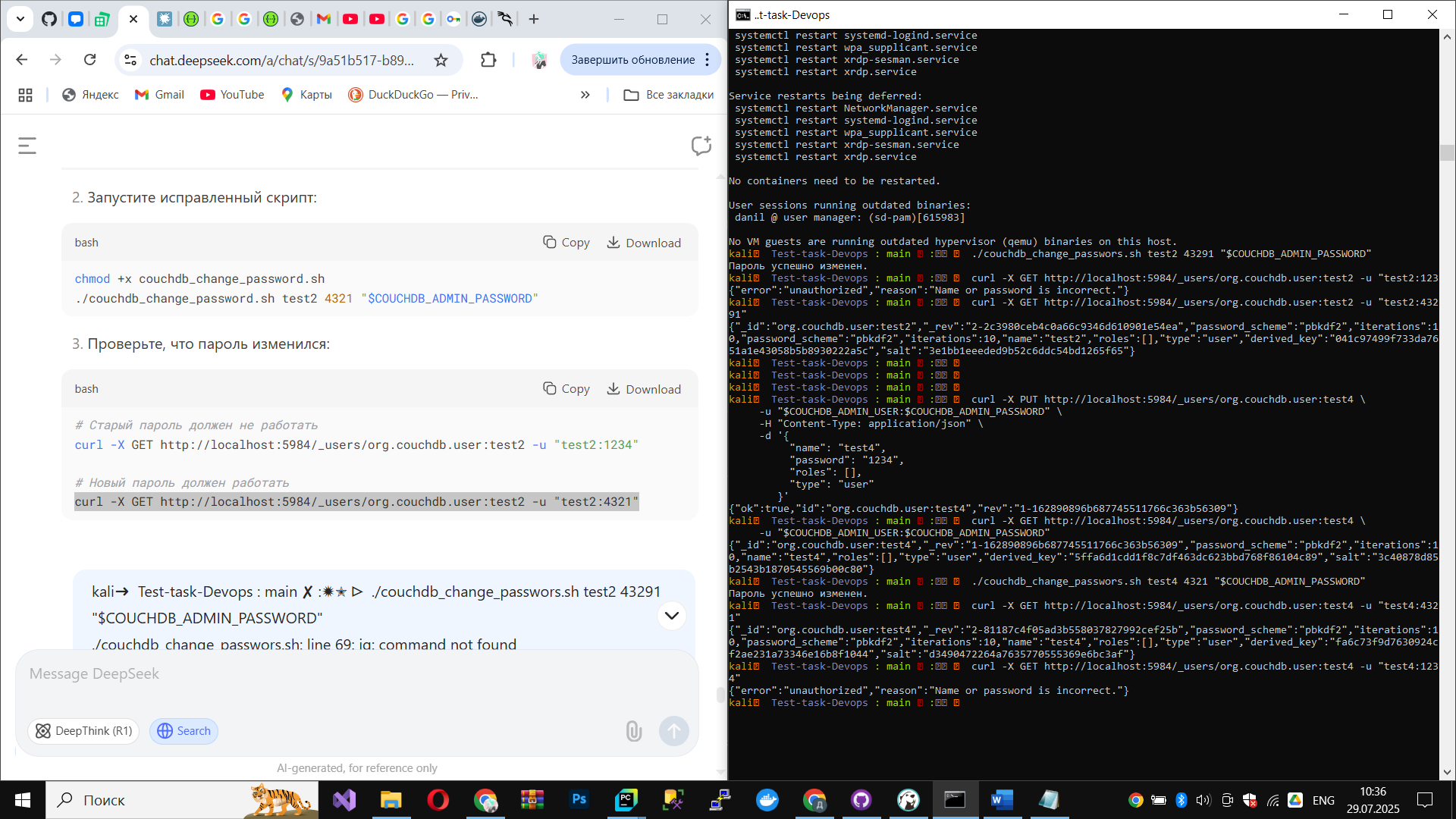


Рисунок 15 – Успешное исполнение скрипта

Исходя из консольного вывода, можно понять, что скрипт для смены пароля успешно сработал.

Вывод: Файлы конфигурации успешно написаны, контейнеры запущены, nginx работает, перенаправляя трафик с порта 5984 на http порт 80, скрипт для смены пароля пользователя написан и успешно работает.

Все задачи исполнены, все материалы выложены в github репозиторий.