МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное   
образовательное учреждение высшего образования  
«Самарский национальный исследовательский университет   
имени академика С.П. Королева»

(Самарский университет)

Институт информатики, математики и электроники

Факультет информатики  
Кафедра суперкомпьютеров и общей информатики

**Отчет по лабораторной работе №2**

Дисциплина: «Методология разработки DevOps»

Тема: **«Helm. JupyterHub»**

Выполнил: Мелешенко И.С.

Группа: 6233-010402D

Самара 2022

**ЗАДАНИЕ**

1. Install Helm, Jupyterhub according with https://github.com/jupyterhub/zero-to-jupyterhub-k8s https://z2jh.jupyter.org/en/stable/
2. Try to run test-notebooks as in following video [11:18] [ Kube 99 ] Zero to Jupyterhub in Kubernetes | Getting Started Guide <https://youtu.be/Da1qn7-RHvY?t=676>
3. Make report with screens of:
4. JupiterHub notebook print("Hello world") or some simple command for your choice executed.

СОДЕРЖАНИЕ

[Задание 0. Установка локального кластера Kubernetes с помощью minikube. 4](#_Toc119255032)

[Задание 0.1 Скачивание локального кластера Kubernetes с помощью minikube 4](#_Toc119255033)

[Задание 0.2 Установка локального кластера Kubernetes с помощью minikube 5](#_Toc119255034)

[Задание 0.3 Запуск локального кластера Kubernetes с помощью minikube 6](#_Toc119255035)

[Задание 0.4 Просмотр версии локального кластера Kubernetes с помощью minikube 7](#_Toc119255036)

[Задание 1. Установка helm. 9](#_Toc119255037)

[Задание 2. Установка JupyterHub. 10](#_Toc119255038)

[Задание 2.1 Создание конфигурационного файла 10](#_Toc119255039)

[Задание 2.2 Добавление репозитория JupyterHub 10](#_Toc119255040)

[Задание 2.3 Запуск локального кластера Kubernetes 11](#_Toc119255041)

[Задание 2.4 Настройка конфигурационного файла 12](#_Toc119255042)

[Задание 2.5 Поиск общедоступного IP-адреса 14](#_Toc119255043)

[Задание 2.6 Запуск JupyterHub 15](#_Toc119255044)

[Задание 2.7 Работа с notebook в JupyterHub 18](#_Toc119255045)

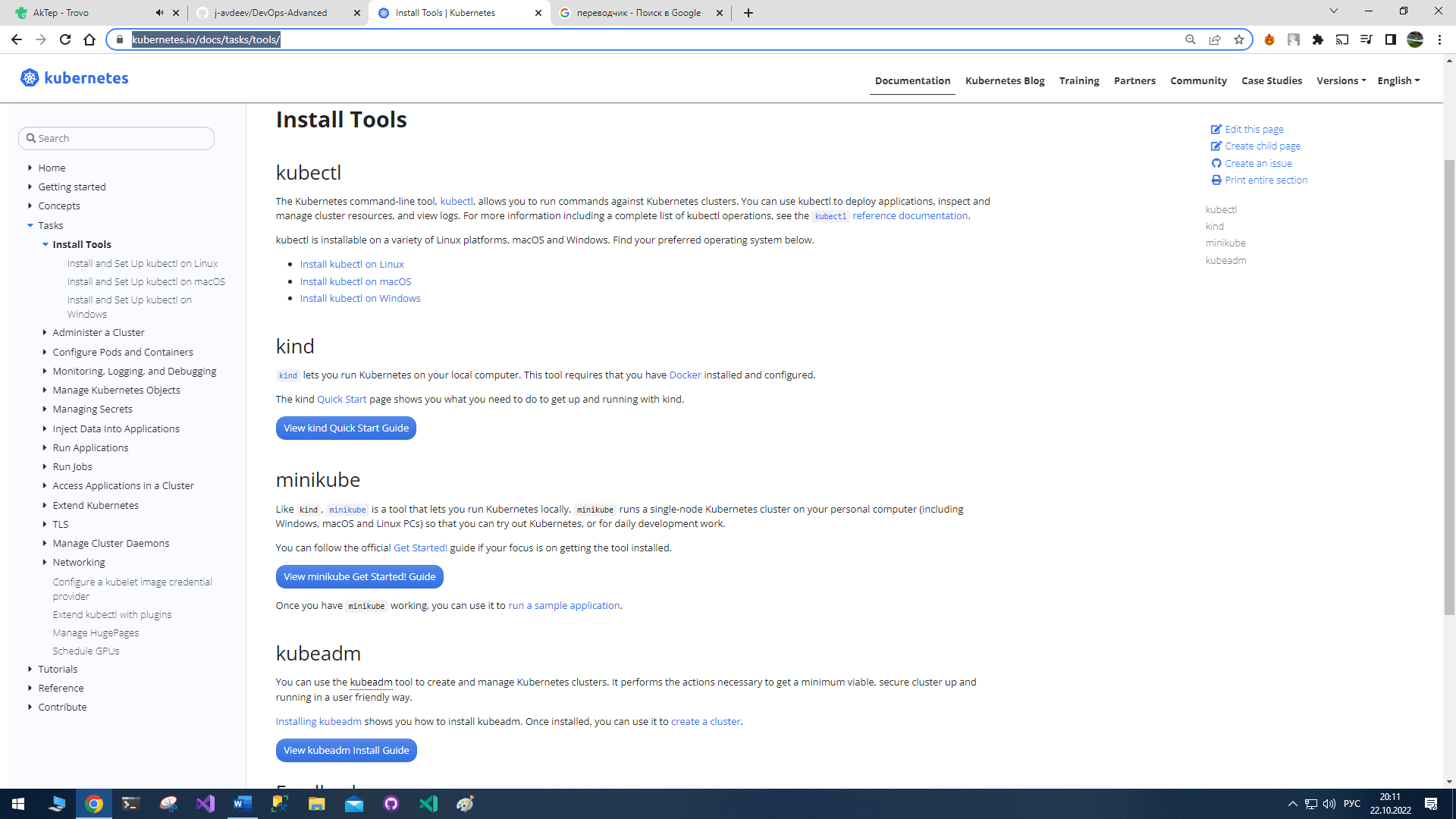
[Заключение 20](#_Toc119255046)

# **Задание 0. Установка локального кластера Kubernetes с помощью minikube.**

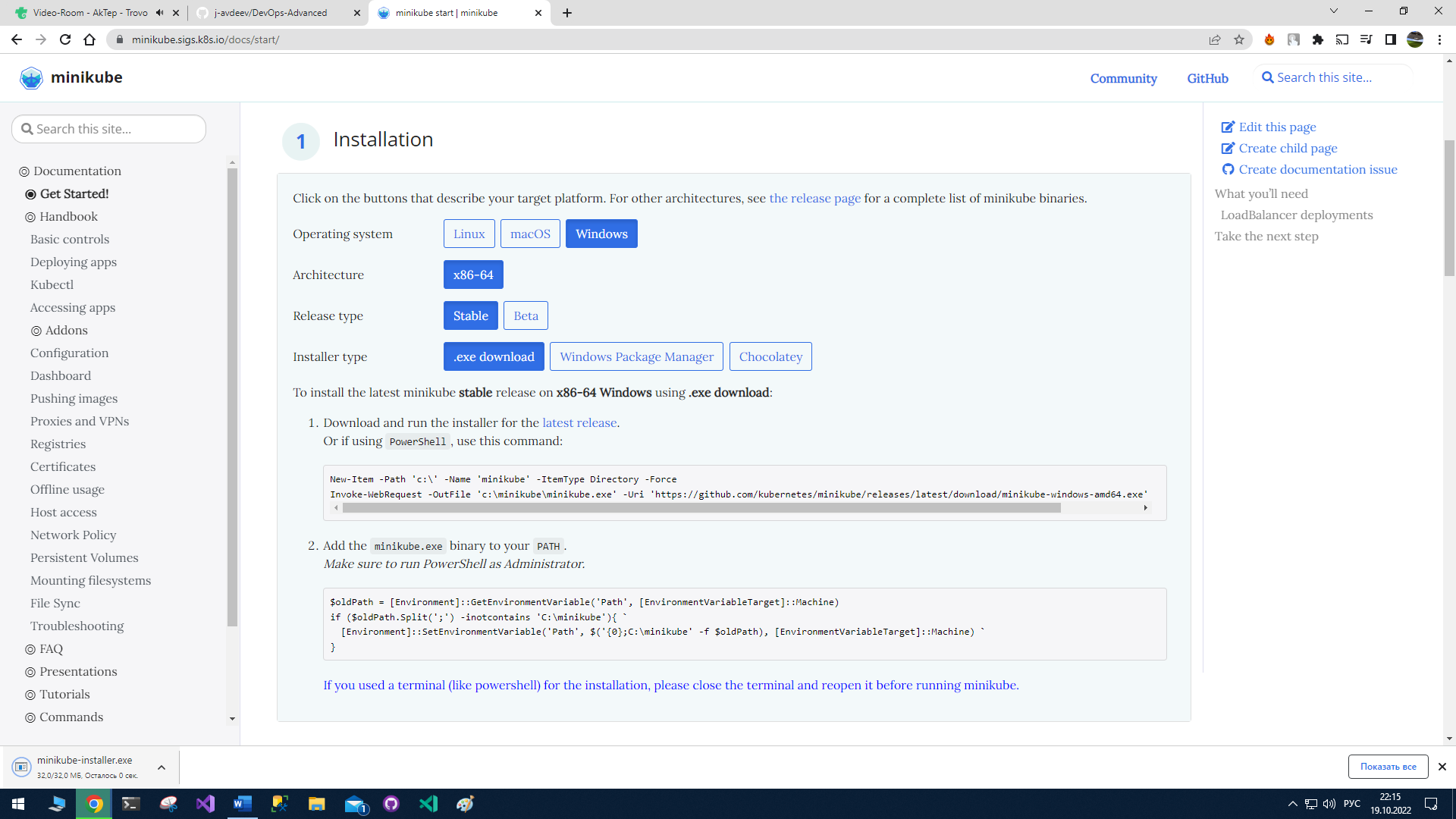
## ***Задание 0.1 Скачивание локального кластера Kubernetes с помощью minikube***

Скачиваем и устанавливаем кластер Kubernetes с помощью minikube с официального сайта по ссылке: <https://kubernetes.io/docs/tasks/tools/>

Выбираем minikube, как показано на рисунке.

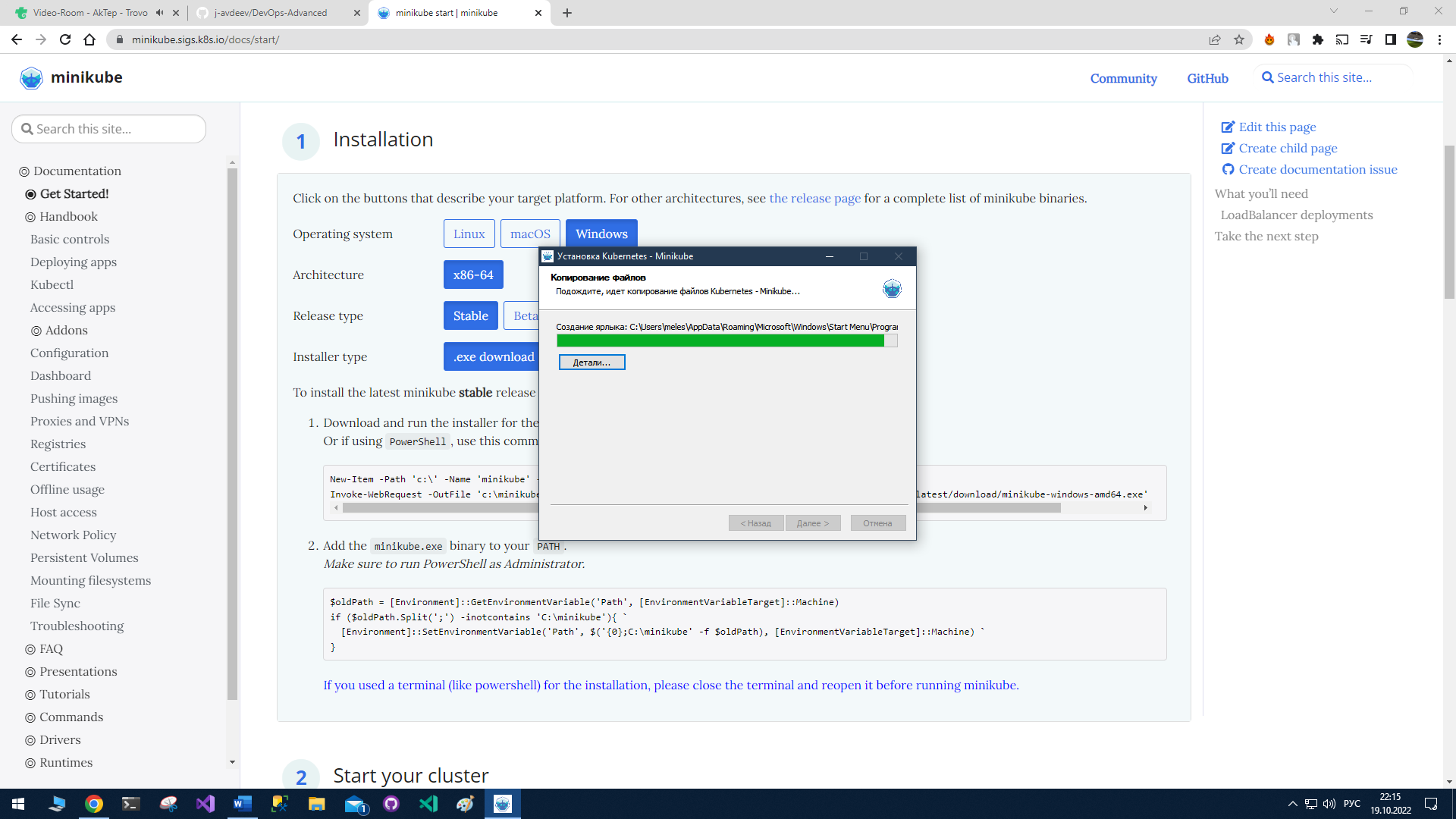
  
Рисунок 1 – Скачивание кластера Kubernetes

Переходим на туториал по установке кластера. Для скачивания, выбираем параметры своего ПК, на который будет устанавливаться кластер. Для своего ПК, я выбрал следующие параметры, как показано на рисунке ниже.

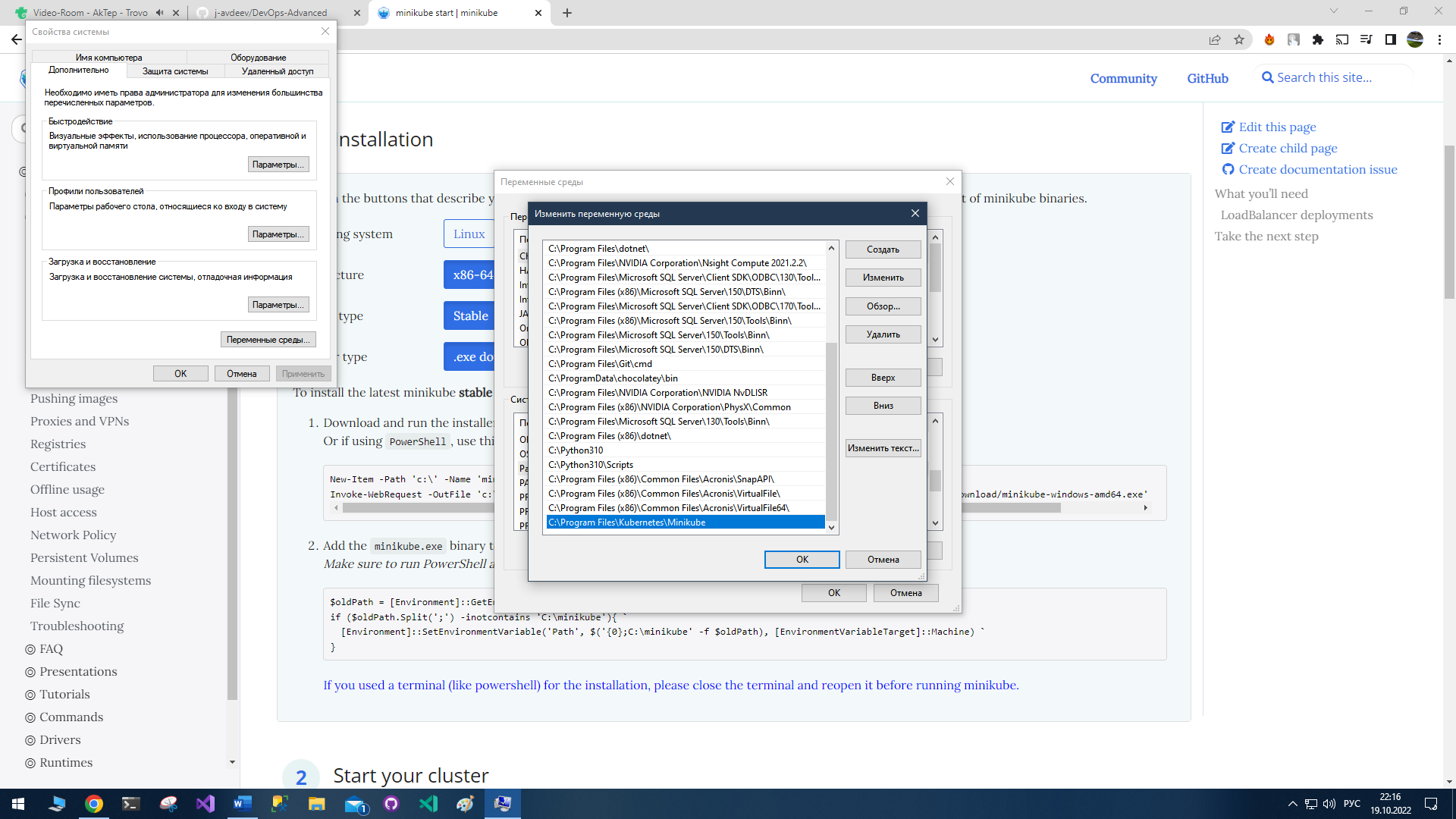
  
Рисунок 2 – Настройка пакета установки

## ***Задание 0.2 Установка локального кластера Kubernetes с помощью minikube***

После скачивания устанавливаем кластер Kubernetes с помощью minikube.

  
Рисунок 3 – Установка кластера Kubernetes с помощью minikube

После завершения установки кластера, проверим пути переменной PATH.

  
Рисунок 4 – Проверка переменной PATH

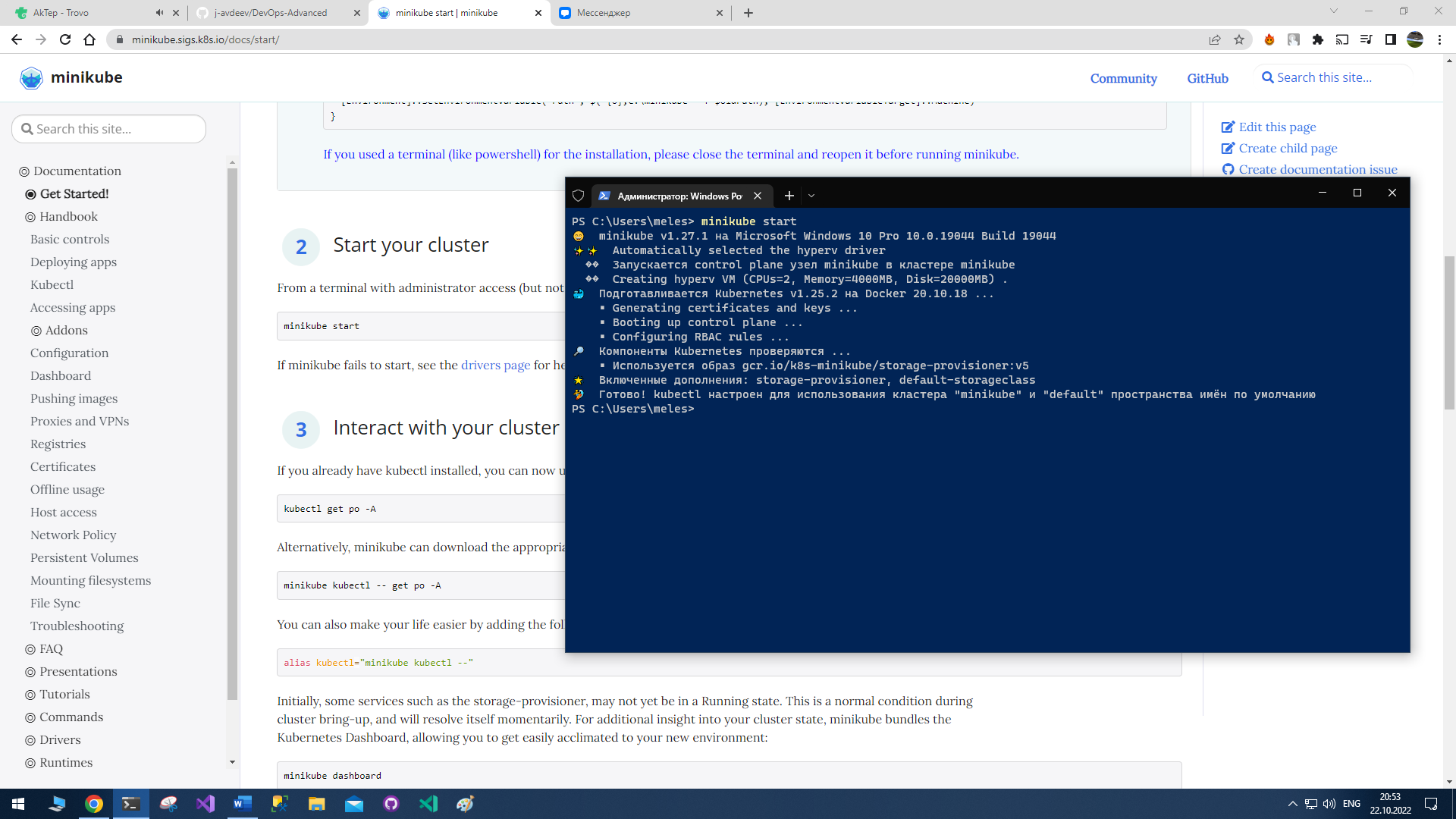
## ***Задание 0.3 Запуск локального кластера Kubernetes с помощью minikube***

После завершения процесса установки, перейдем в командную строку и продолжим работу там.

Для запуска локального кластера воспользуемся командой

~$ minikube start

Результат выполнения показан на рисунке ниже:

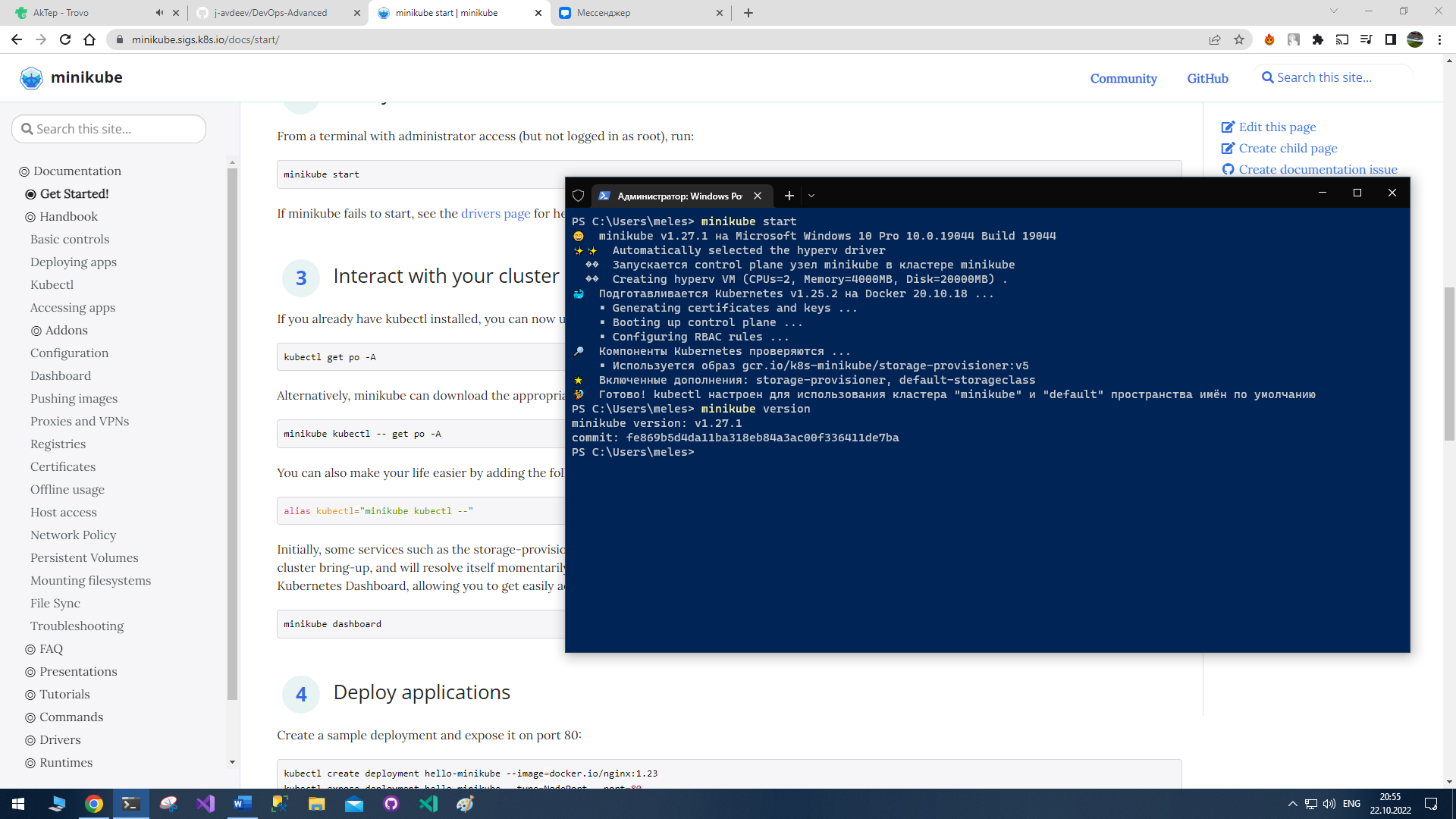
  
Рисунок 5 – Запуск локального кластера Kubernetes с помощью minikube

## ***Задание 0.4 Просмотр версии локального кластера Kubernetes с помощью minikube***

Для просмотра версии установленного кластера на ПК, воспользуемся следующей командой:

~$ minikube version

Результат выполнения данной команды показан на рисунке:

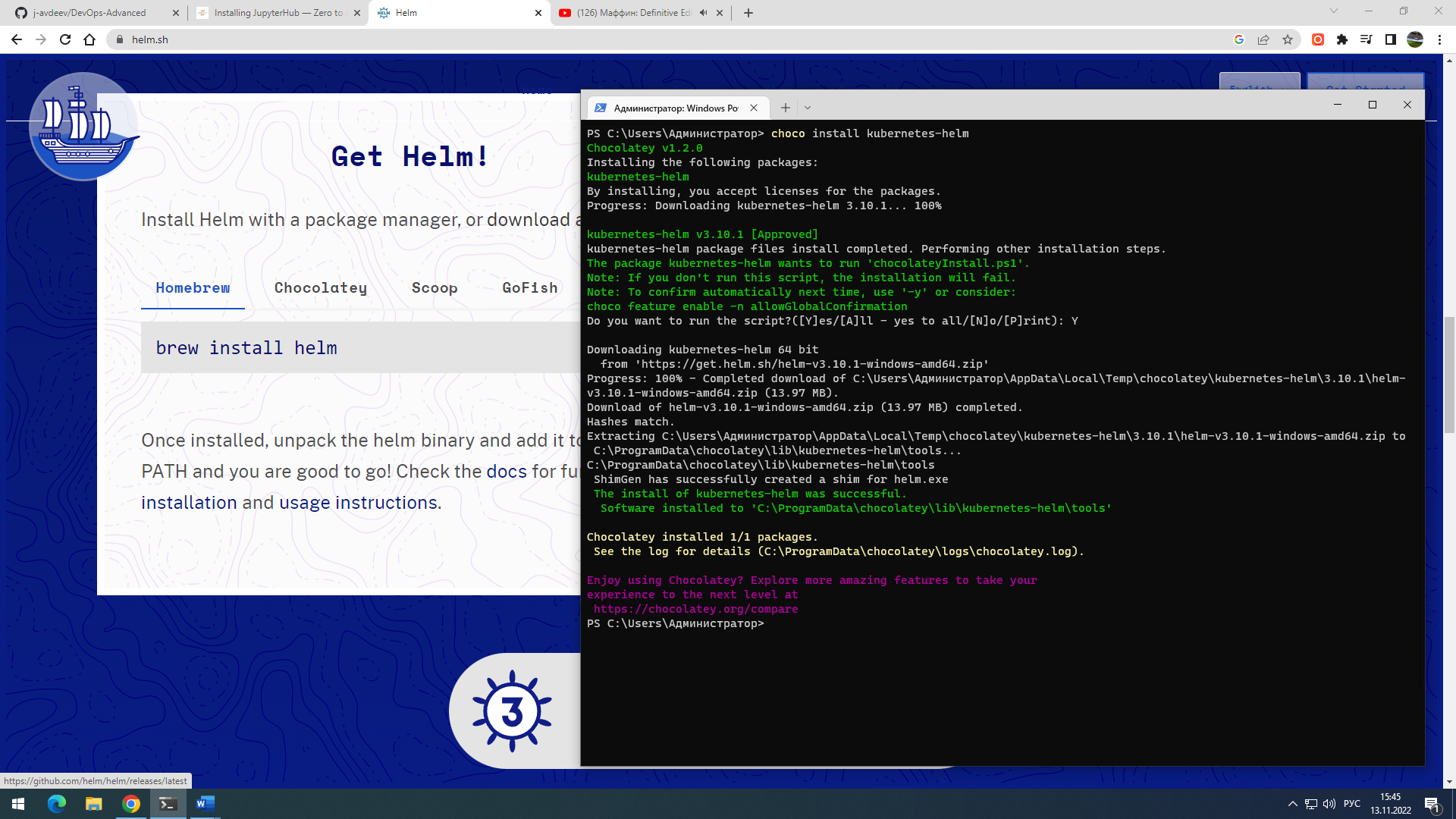
  
Рисунок 6 – Просмотр версии кластера

# **Задание 1. Установка helm.**

Что такое helm?

Helm - это менеджер пакетов, который помогает разработчикам "легко управлять и развертывать приложения на кластере Kubernetes”. Его можно установить несколькими способами, я воспользовался самым простым и быстрым. На официальном сайте, по ссылке <https://helm.sh/>, указаны комнды для Windows 10, для установки Helm. Я использовал слудующую:

~$ choco install kubernetes-helm

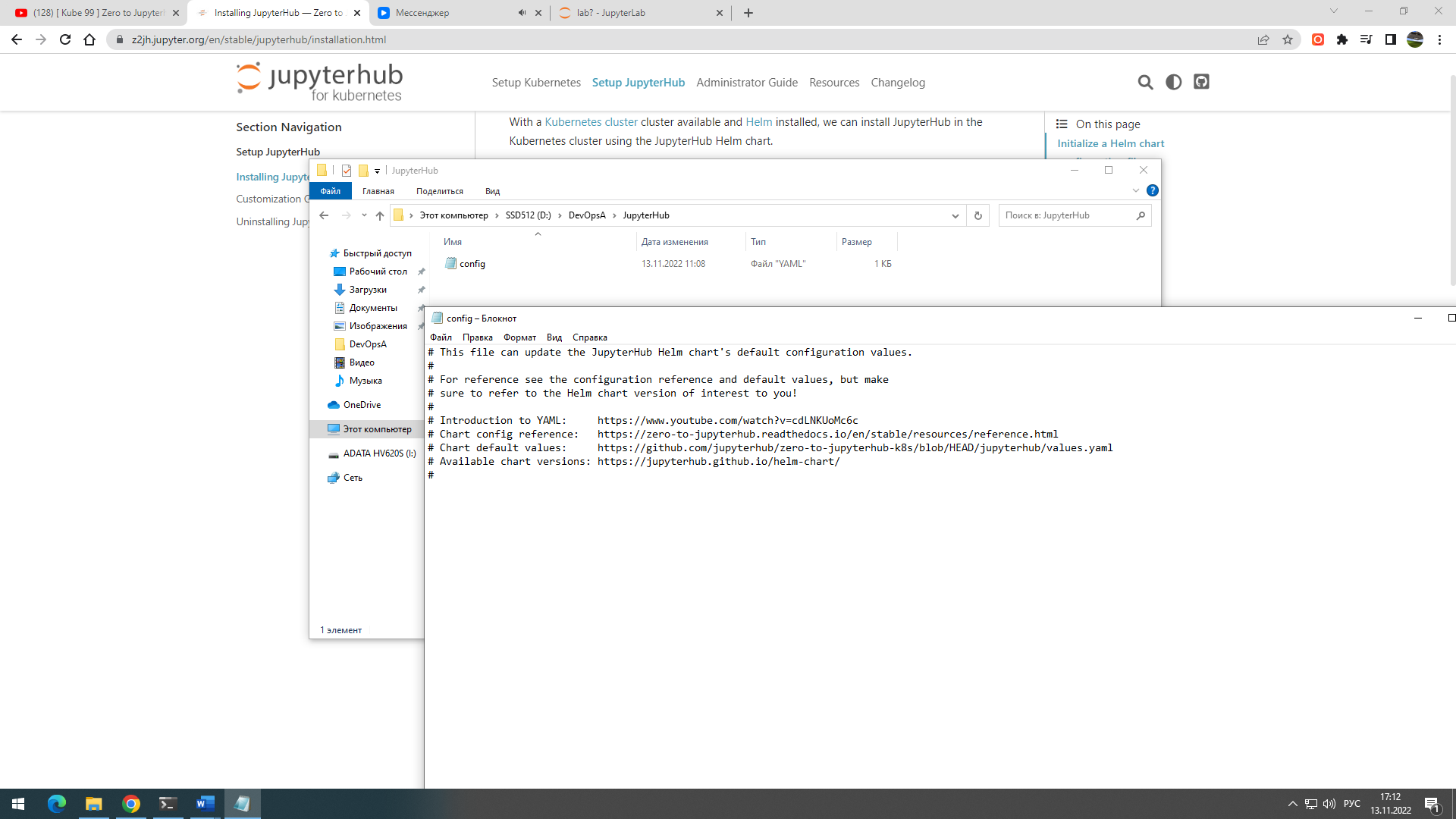
  
Рисунок 7 – Установка Helm

После установки менеджера пакетов Helm, мы можем приступить к установке jupyterhub-а.

# **Задание 2. Установка JupyterHub.**

***Задание 2.1 Создание конфигурационного файла***

Перед тем, как начать установку JupyterHub, нам необходимо создать конфигурационный файл *config.yaml.* Однако, начиная с версии 1.0.0, не нужна никакая конфигурация для начала работы JupyterHub, поэтому просто создаем файл *config.yaml* с некоторыми полезными комментариями, например как на рисунке ниже.

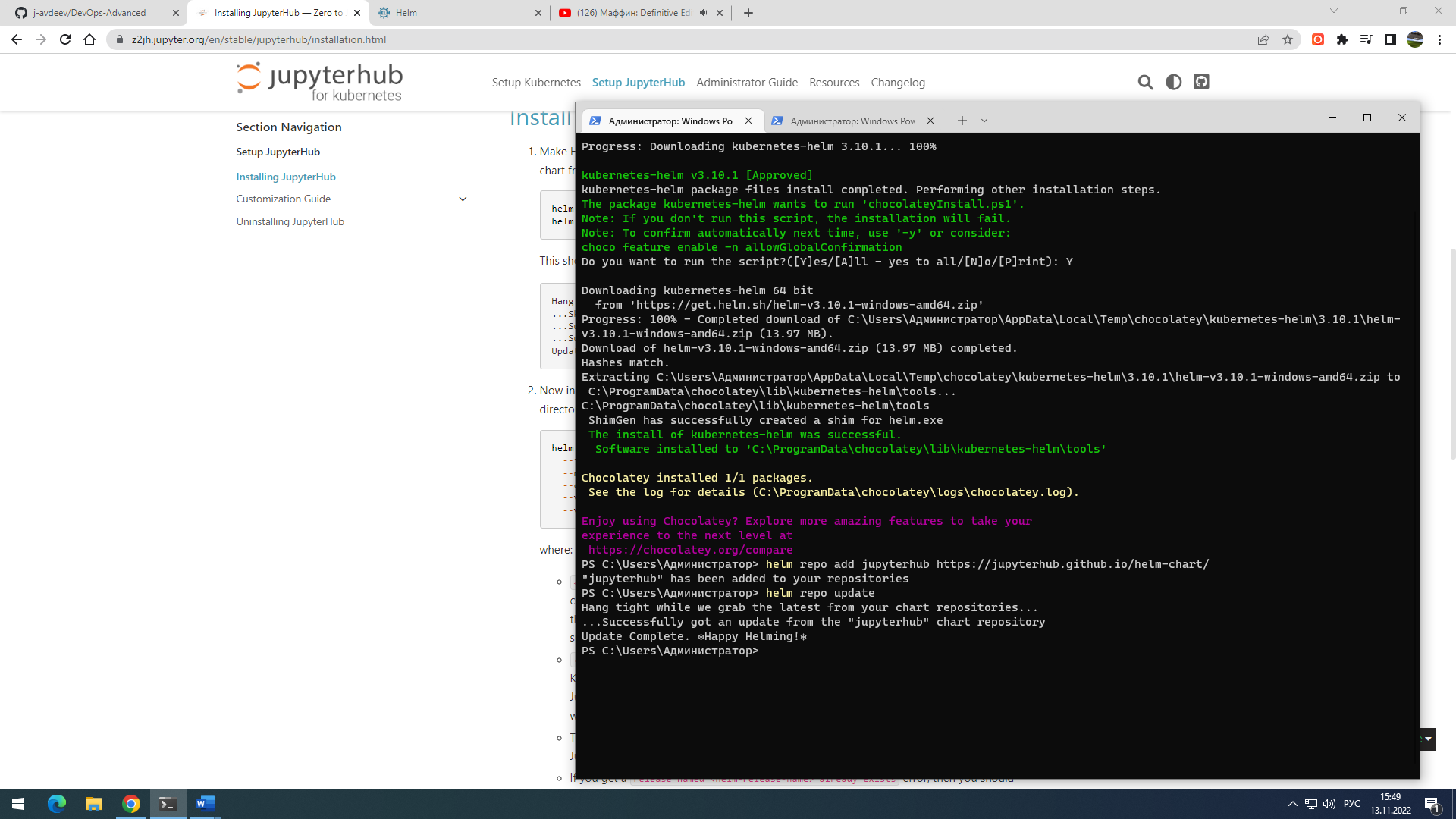
  
Рисунок 8 – Создание конфигурационного файла

***Задание 2.2 Добавление репозитория JupyterHub***

При помощи установленного менеджера пакетов Helm, добавляем к себе репозиторий JupyterHub, при помощи следующих команд:

~$ helm repo add jupyterhub https://jupyterhub.github.io/helm-chart/

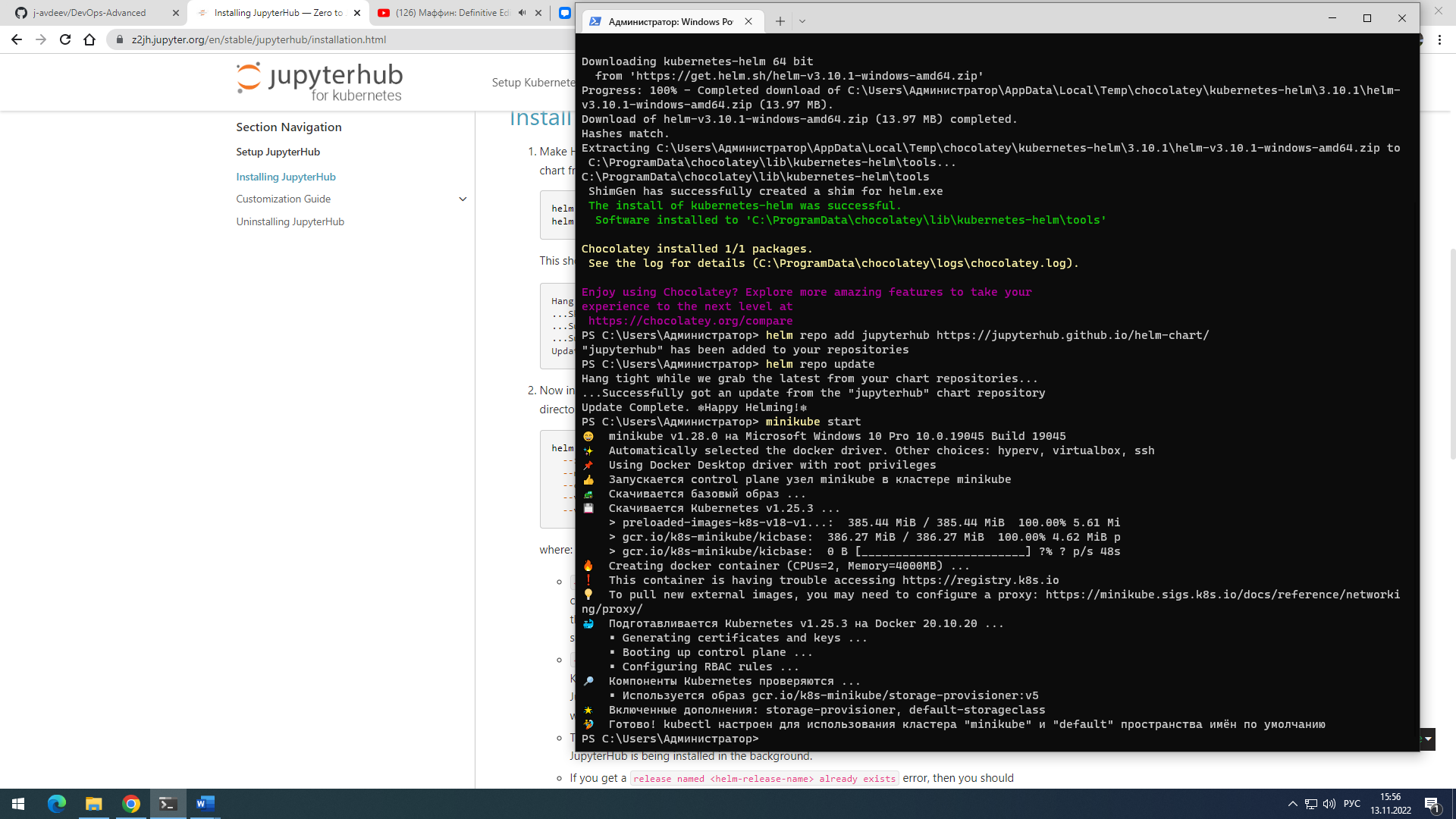
~$ helm repo update

  
Рисунок 9 – Добавление репозитория JupyterHub

***Задание 2.3 Запуск локального кластера Kubernetes***

Для продолжения работы нам потребуется установленный ранее локальный кластер Kubernetes, запустим его командой:

~$ minikube start

  
Рисунок 10 - Запускаем локальный кластер Kubernetes

***Задание 2.4 Настройка конфигурационного файла***

После того как запустили кластер, в новом терминале перейдем в папку, с конфигурационным файлом и произведем его настройку при помощи следующей команды:

helm install <helm-release-name> jupyterhub/jupyterhub \

--namespace <k8s-namespace> \

--create-namespace \

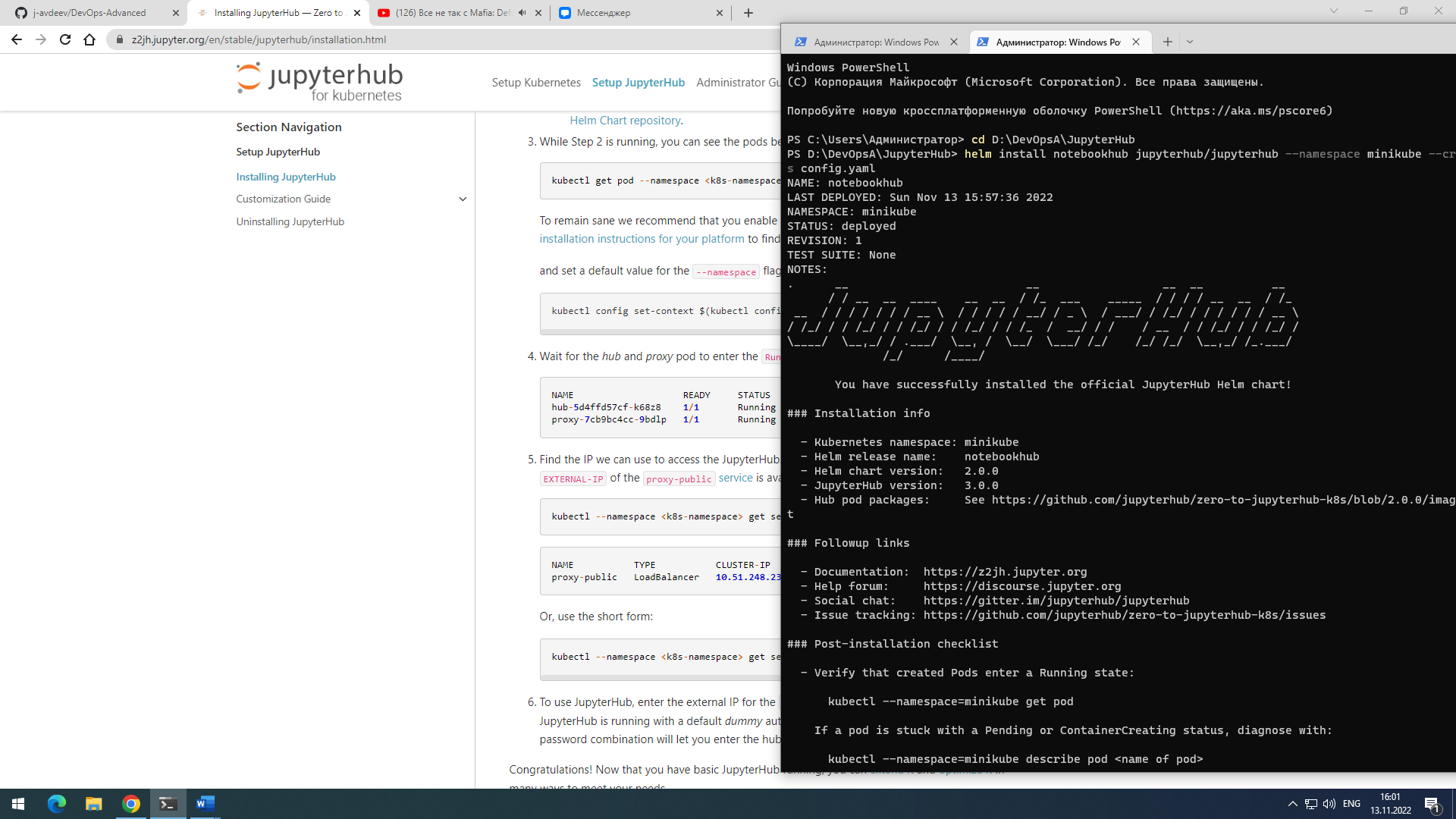
--values config.yaml

где:

* <helm-release-name> относится к имени выпуска Helm, идентификатору, необходимому для различия установленных диаграммы.
* <k8s-namespace> относится к пространству имен Kubernetes, идентификатору, используемому для группировки ресурсов Kubernetes, в данном случае всех ресурсов Kubernetes, связанных с диаграммой JupyterHub.

В моем случае эта команда выглядит следующим образом:

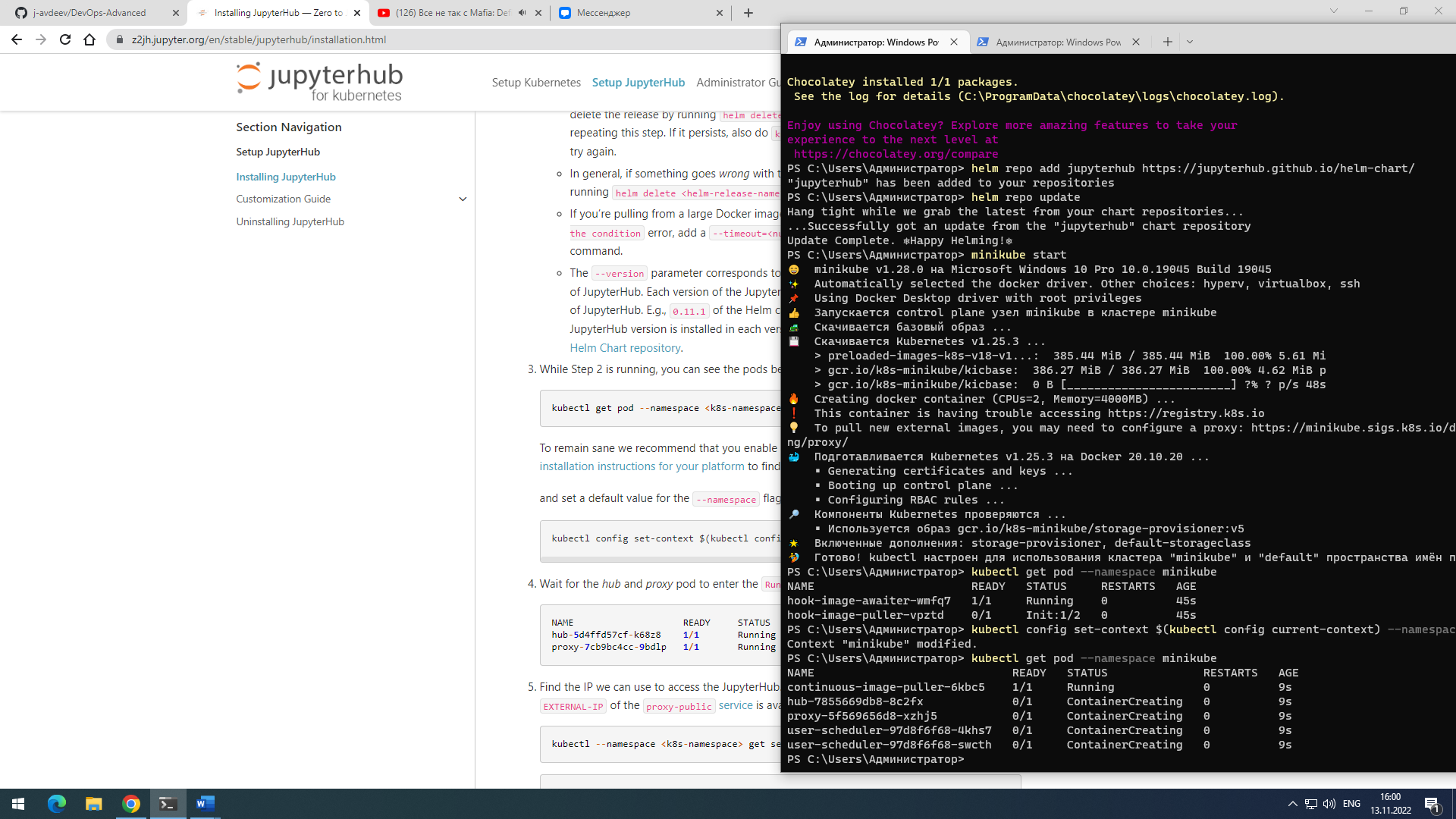
~$ helm install notebookhub jupyterhub/jupyterhub --namespace minikube --create-namespace --values config.yaml

  
Рисунок 11 – Настройка конфигурационного файла

Поскольку настройка требует некоторого времени, то процесс мы можем отследить при помощи команды ниже, а также произвести установку по умолчанию для флага пространства имен, для чего возвращаемся в предыдущий терминал:

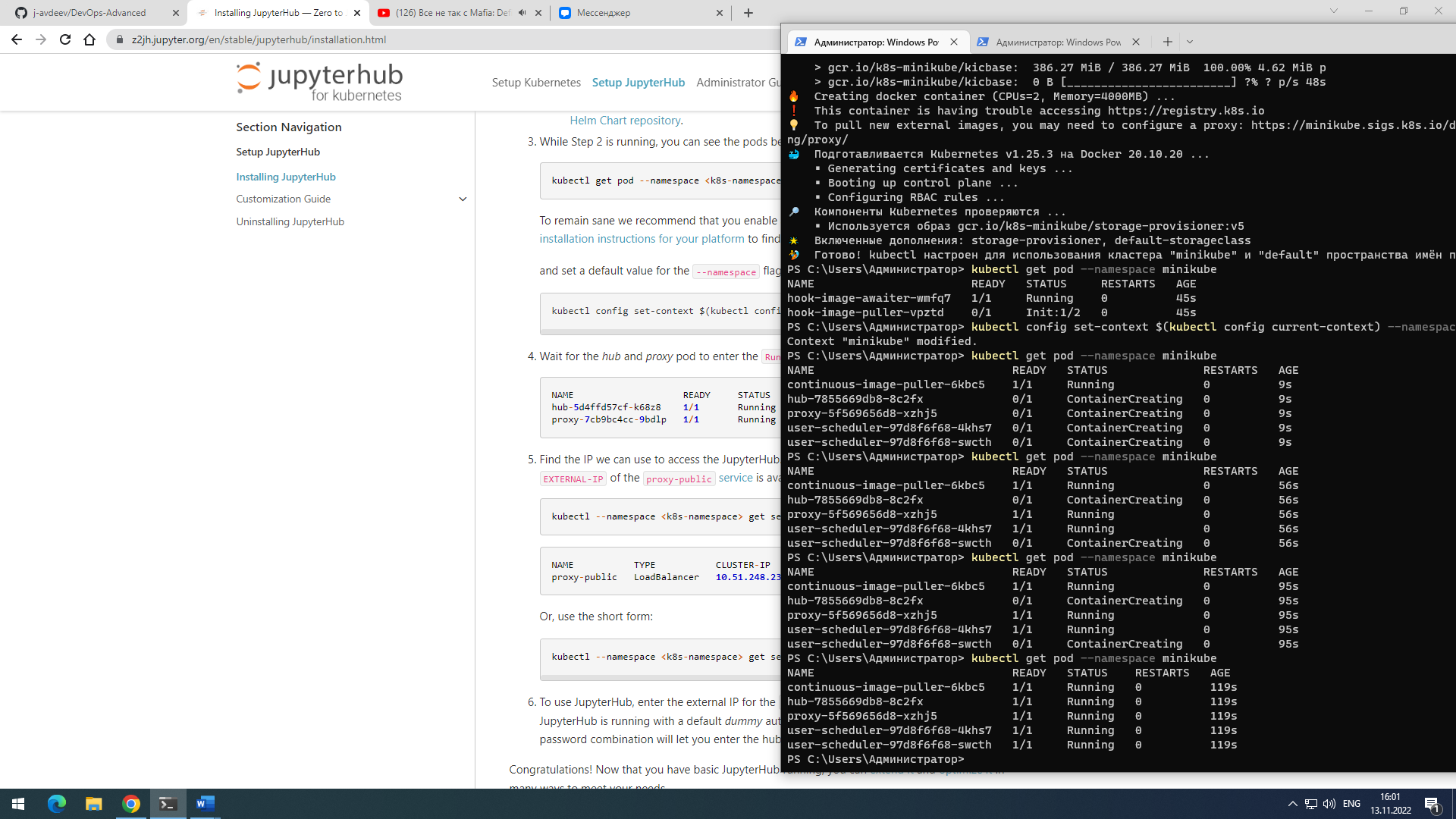
~$ kubectl get pod --namespace minikube

~$ kubectl config set-context $(kubectl config current-context) --namespace minikube

  
Рисунок 12 – Проверка настройки файла

По окончании настройки конфигурационного файла мы увидим, переход модулей hub и proxy передут в состояние запущен.

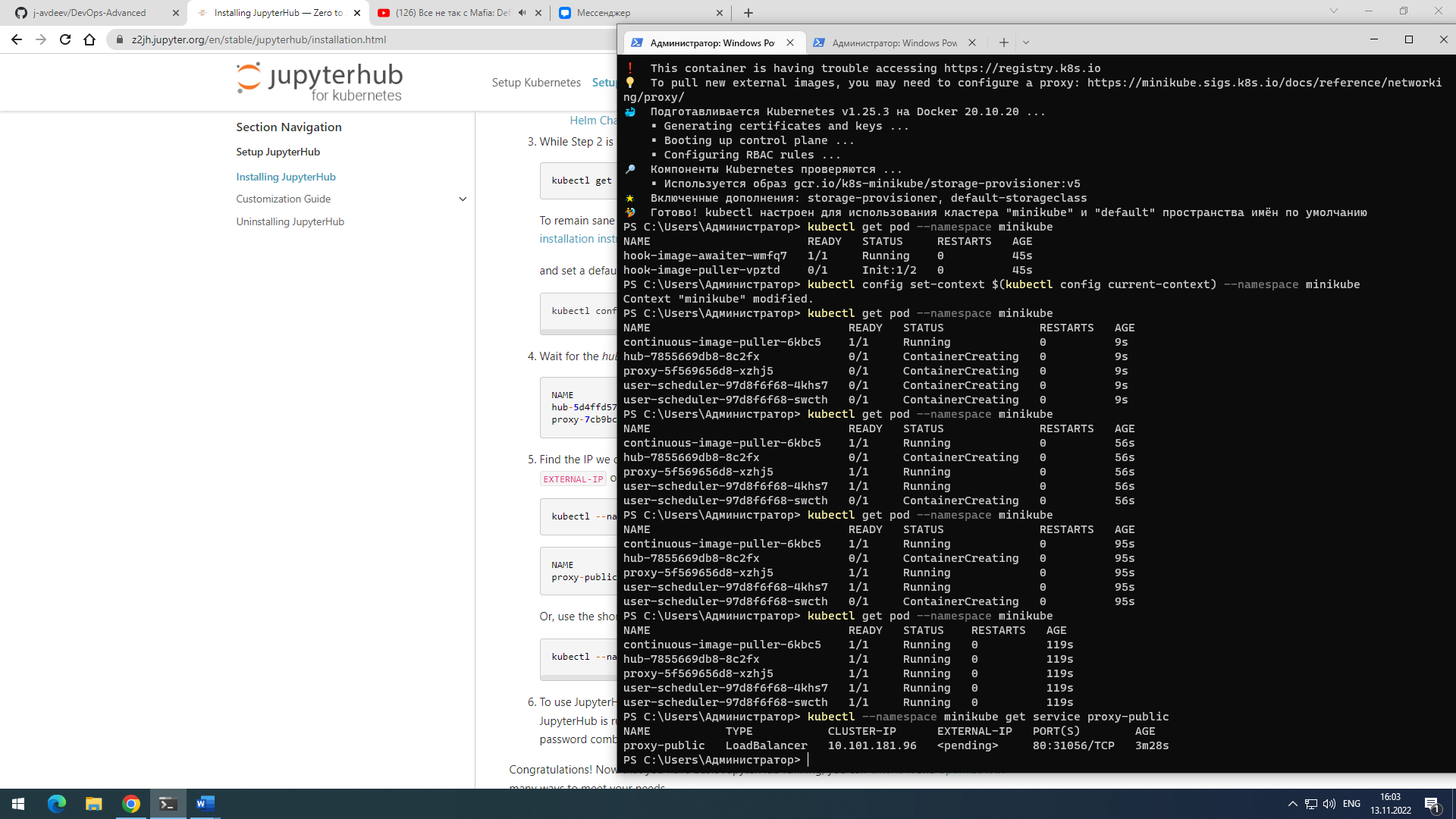
~$ kubectl get pod --namespace minikube

  
Рисунок 13 – Проверка настройки файла

***Задание 2.5 Поиск общедоступного IP-адреса***

Для продолжения работы необходимо найти IP-адрес, который можно использовать для доступа к JupyterHub. Для этого выполняем следующую команду и смотрим на EXTERNAL-IP proxy-public, как в примере вывода.

~$ kubectl --namespace minikube get service proxy-public

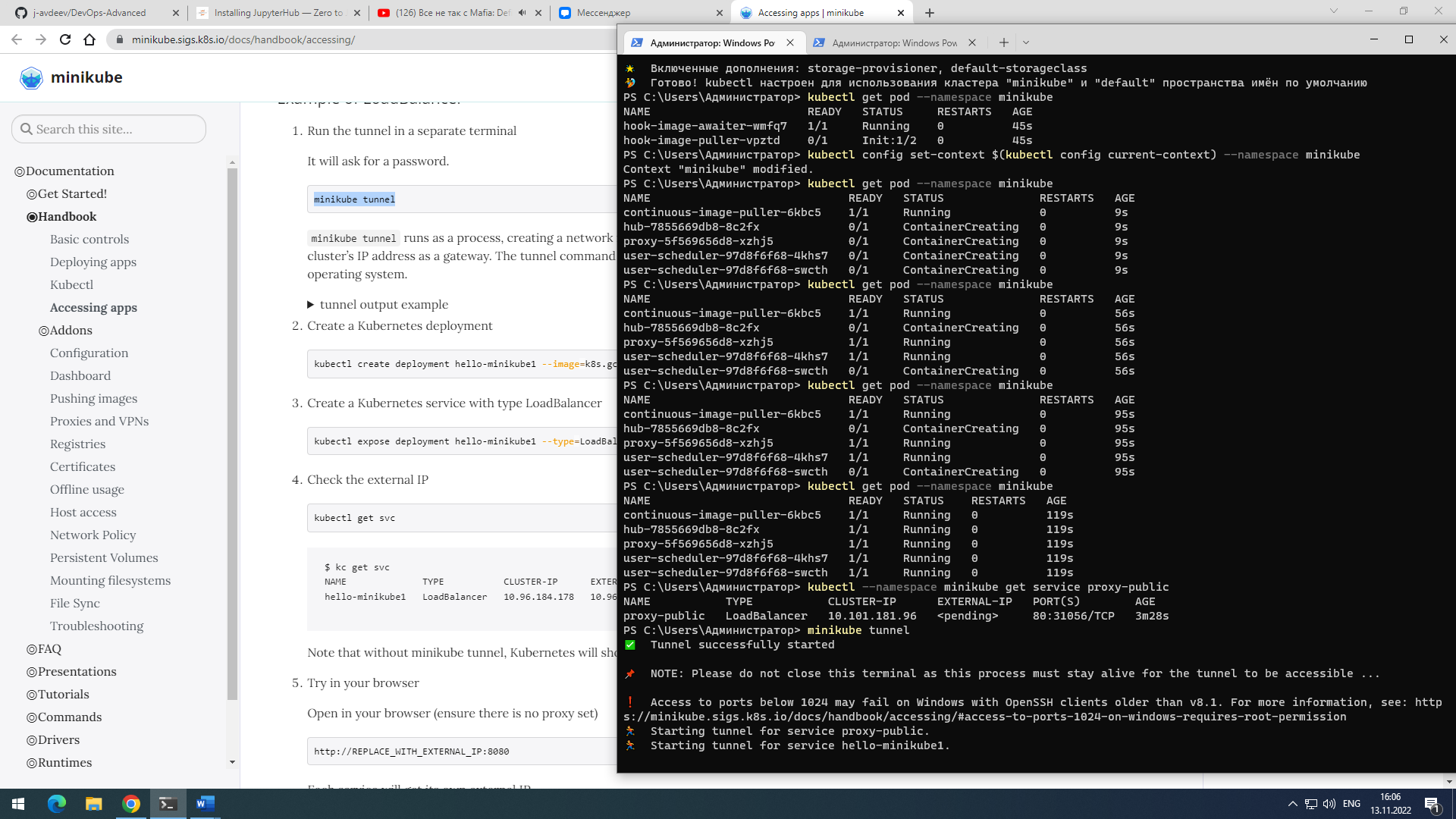
  
Рисунок 14 – Поиск IP адреса для JupyterHub

К сожалению, в столбце EXTERNAL-IP указано <pending> вместо IP-адреса, для продолжения работы нам необходимо произвести до настройку, для этого воспользуемся инструкцией по ссылке <https://minikube.sigs.k8s.io/docs/handbook/accessing/>.

***Задание 2.6 Запуск JupyterHub***

Запустим тоннель (аналог VPN) при помощи команды:

~$ minikube tunnel

  
Рисунок 15 – Запуск тоннеля

После запуска тоннеля переходим в новый терминал и последовательно выполняем три следующие команды:

Произведем развертывание Kubernetes:

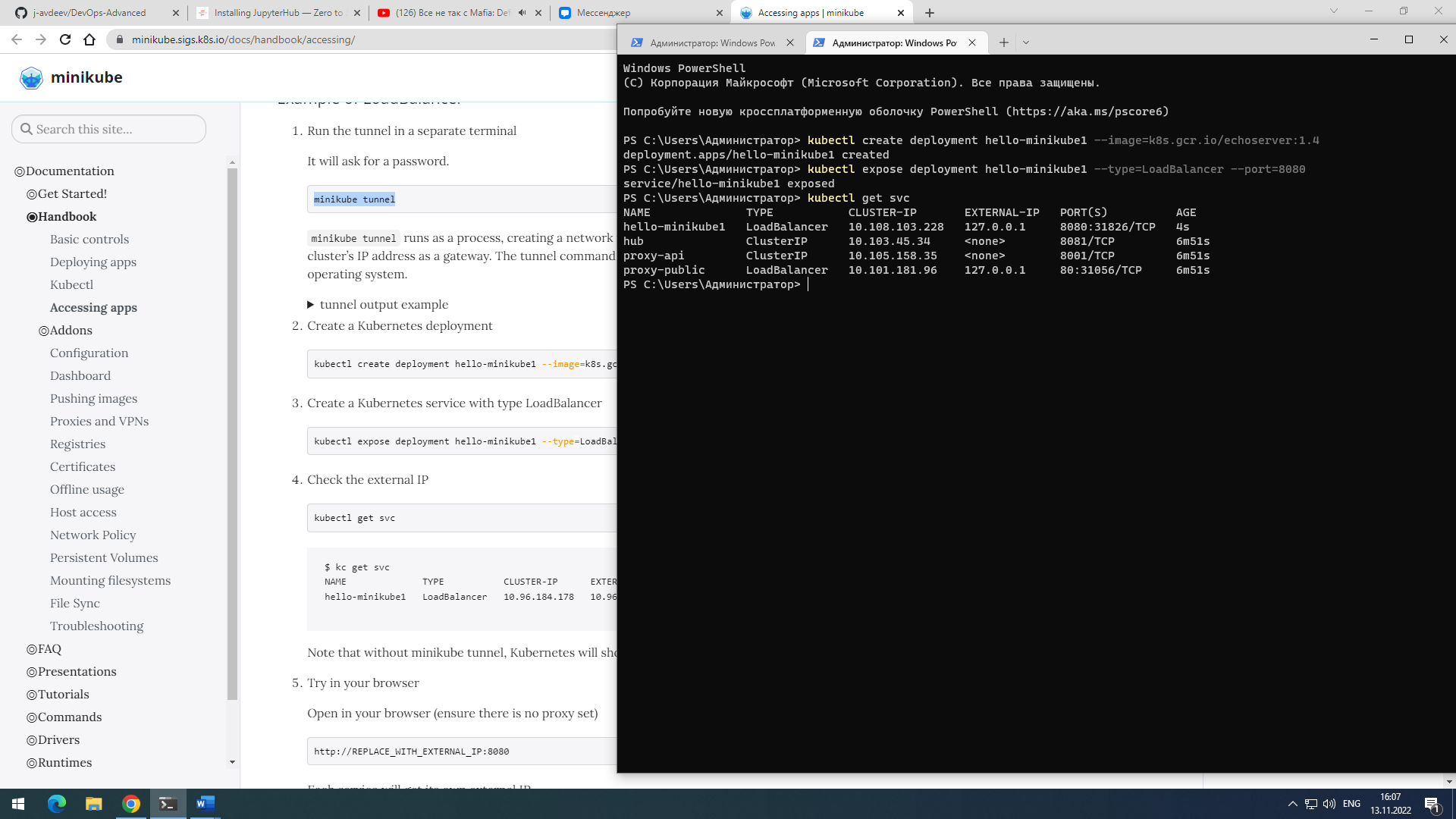
~$ kubectl create deployment hello-minikube1 --image=k8s.gcr.io/echoserver:1.4

Создадим сервис Kubernetes с типом LoadBalancer:

~$ kubectl expose deployment hello-minikube1 --type=LoadBalancer --port=8080

Проверим внешний IP:

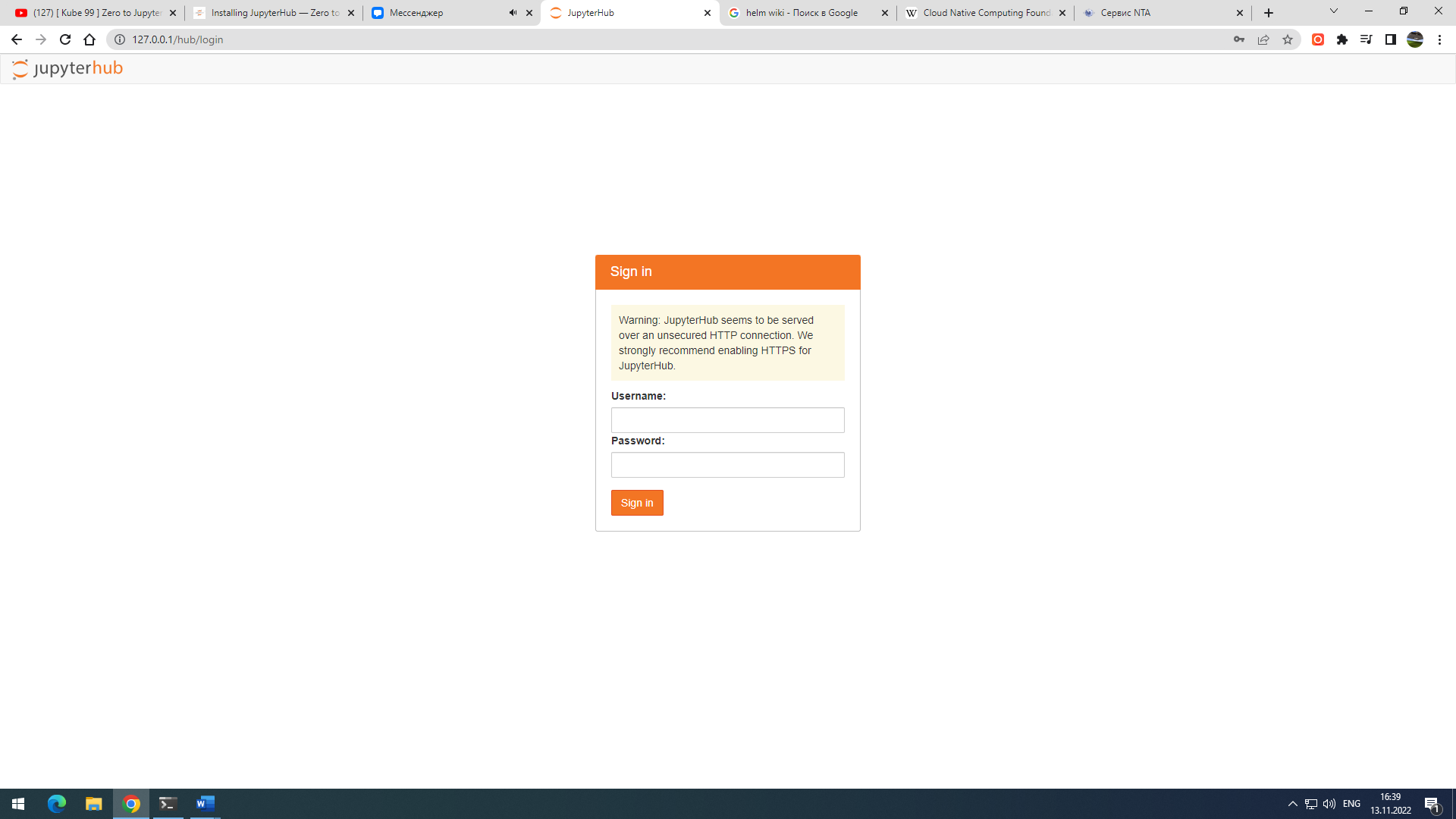
~$ kubectl get svc

  
Рисунок 16 – Проверка IP для JupyterHub

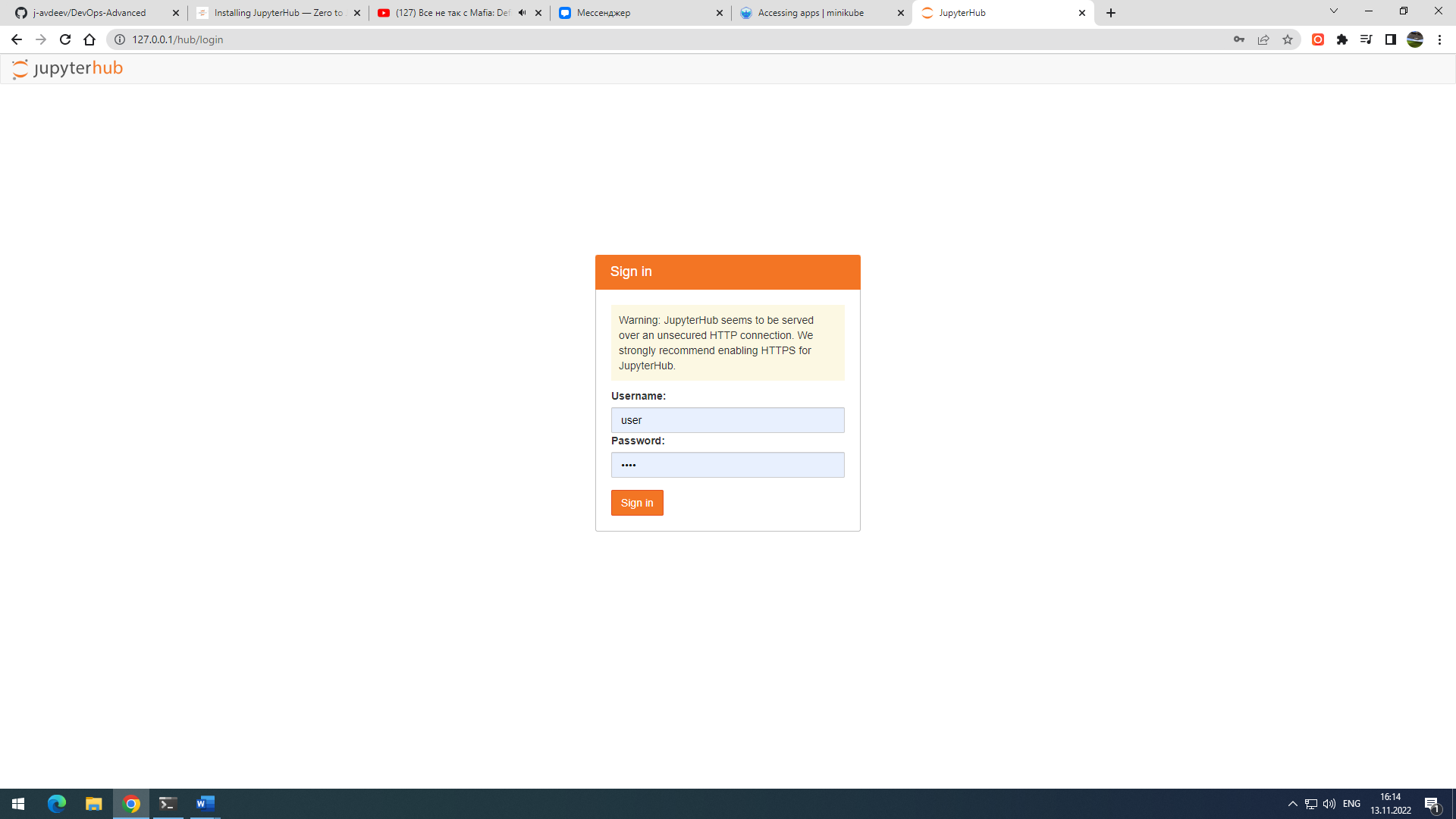
После того, как в графе EXTERNAL-IP для нашего приложения появился адрес:

<http://127.0.0.1>

Переходим по нему в браузер. В результате чего, попадаем на окно авторизации JupyterHub.

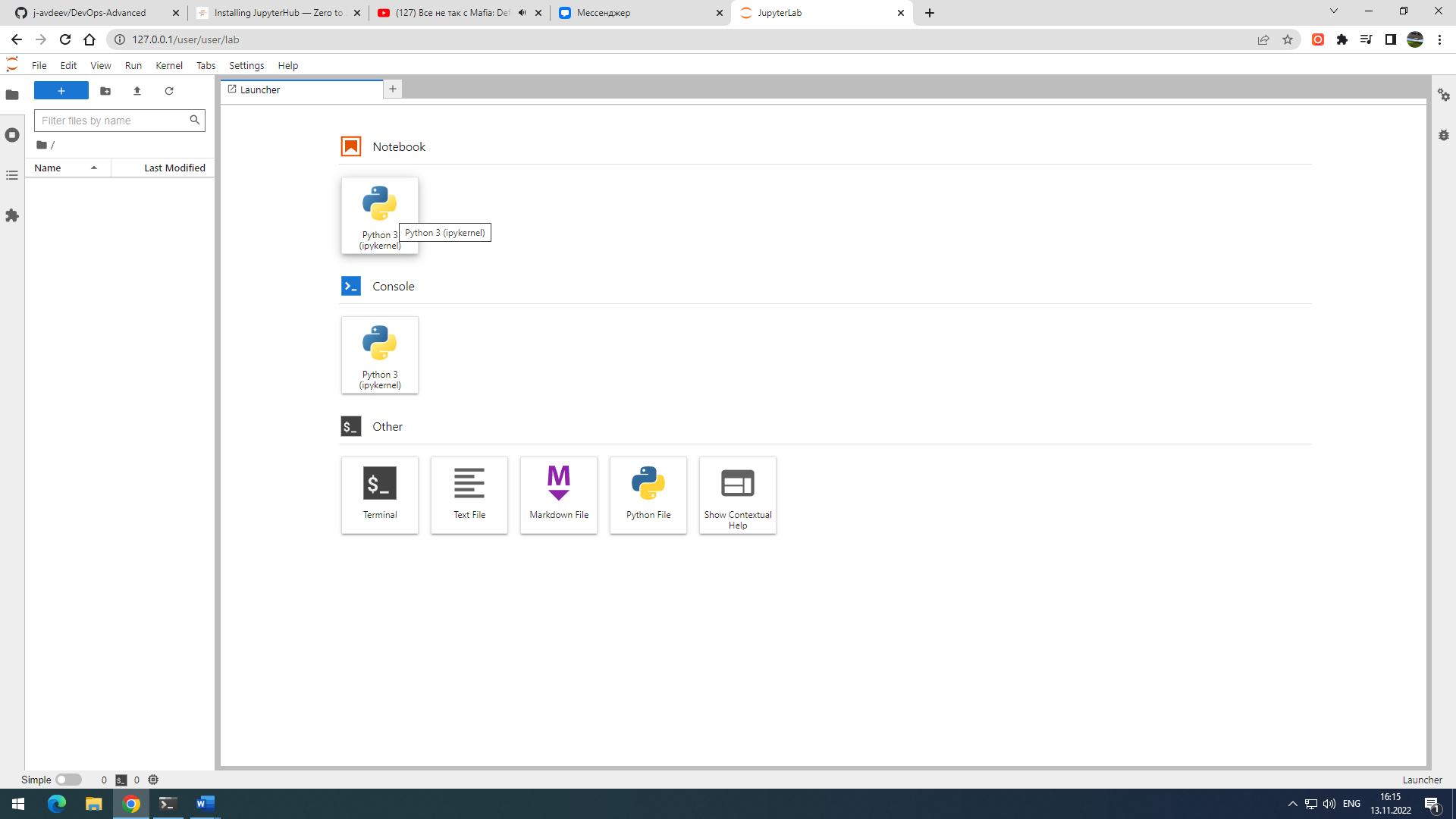
  
Рисунок 17 – Окно авторизации в JupyterHub

Для авторизации можно использовать любую комбинацию логина и пароля, я использовал user – user.

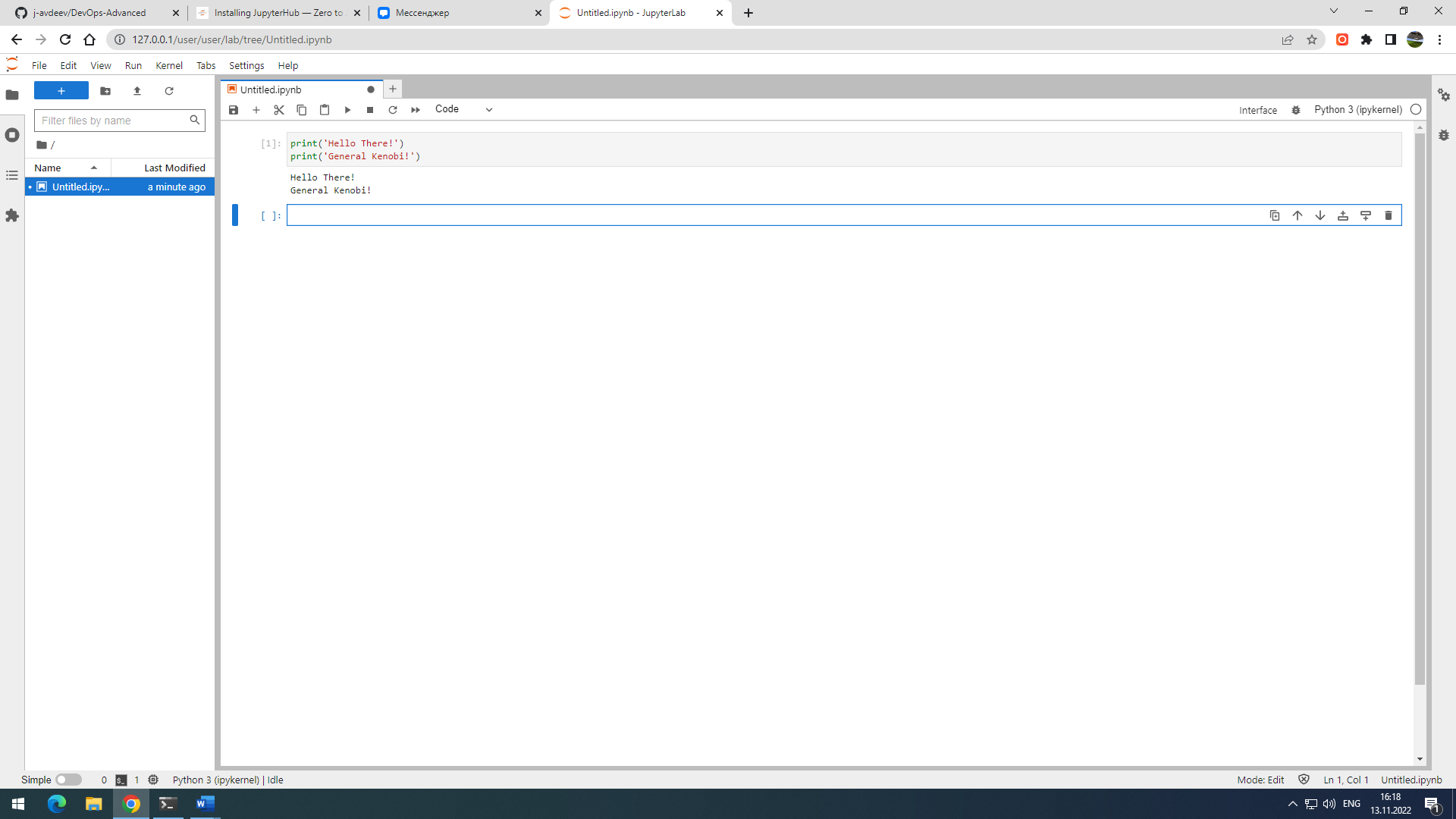
  
Рисунок 18 – Авторизация в JupyterHub

***Задание 2.7 Работа с notebook в JupyterHub***

После успешной авторизации, попадаем на главное окно, на котором нам предлагаются ноутбуки на выбор. Выбираем первый и продолжаем.

  
Рисунок 19 – Выбор notebook в JupyterHub.

В качестве теста, выполним пару простейших команд в notebook, результат работы приведен ниже.

  
Рисунок 20 – Отработавший notebook.

# **ЗАКЛЮЧЕНИЕ**

В результате выполнения данной лабораторный работы были получены первоначальны навыки работы с JupyterHub при помощи minikube и helm. Было совершено подключение к публичному прокси и написаны простейшие команды по выводу информации на notebook.