Lab 2.1: Development Environment Setup

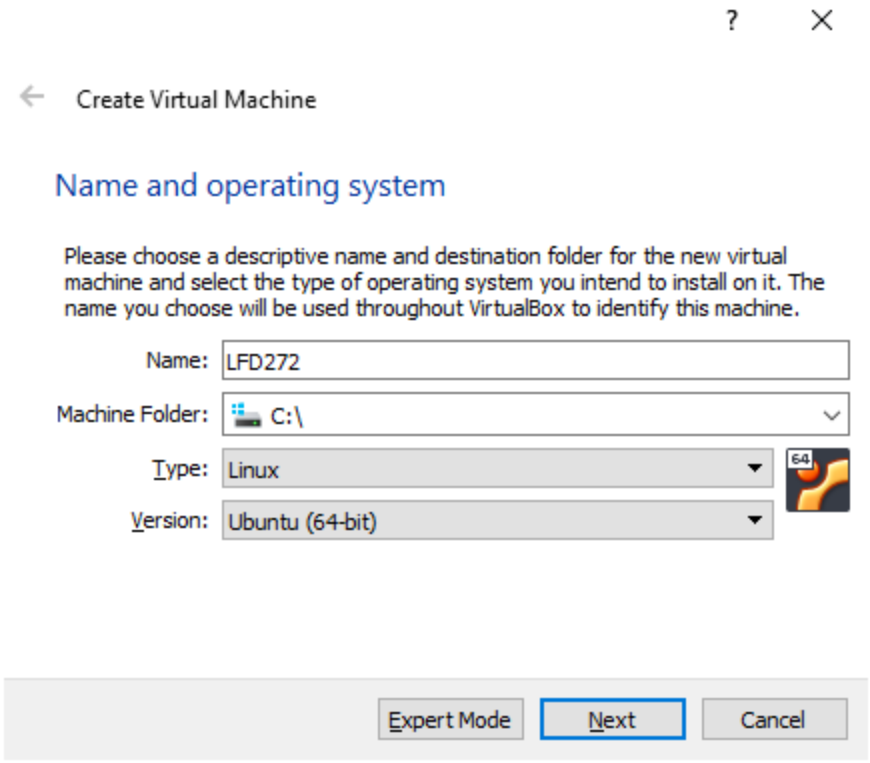
In this chapter, we will go through the necessary steps to set up the development environment for the course.

All the instructions in the chapter were tested by the course creators. To avoid any irregularities, we recommend using the resulting configuration for the duration of the course. Nevertheless, if you want to substitute the software used or tune the hardware requirements, you can do so at your own risk.

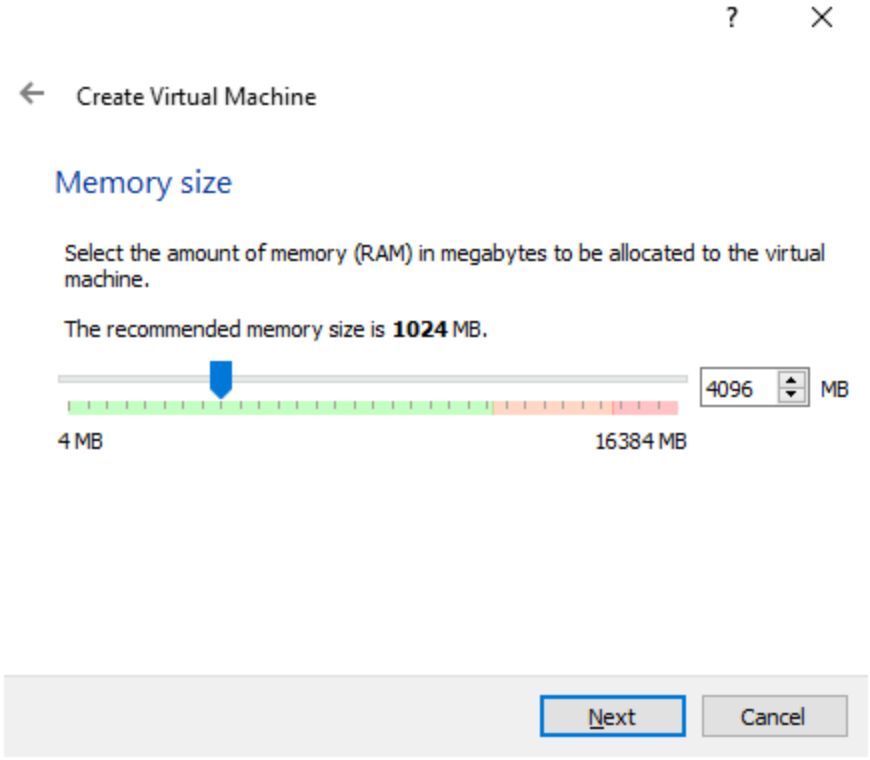
Create a virtual machine

Note: In this section, you can find the instructions on how to create a virtual machine (VM) on your computer. It is also possible to host a VM on a preferred cloud provider where the process described below is fully or partially automated.

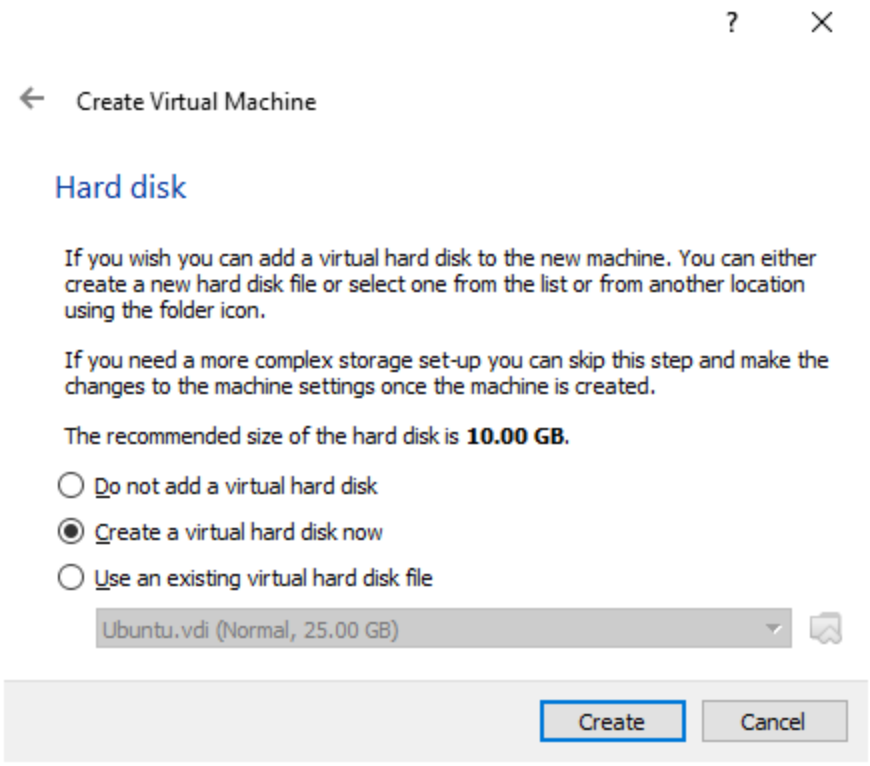
1. Download and install the Oracle VM VirtualBox platform package for your operating system.
2. Download a desktop image of Ubuntu 20.04.2.0 LTS (Focal Fossa).
3. Create a new virtual machine:
    a. Run Oracle VM VirtualBox Manager.
    b. Click on the **New** (or **Ctrl+N**).
    c. In the **Name and operating system** window, enter any VM's name and folder, and choose the **Linux** type and the **Ubuntu (64-bit)** version.



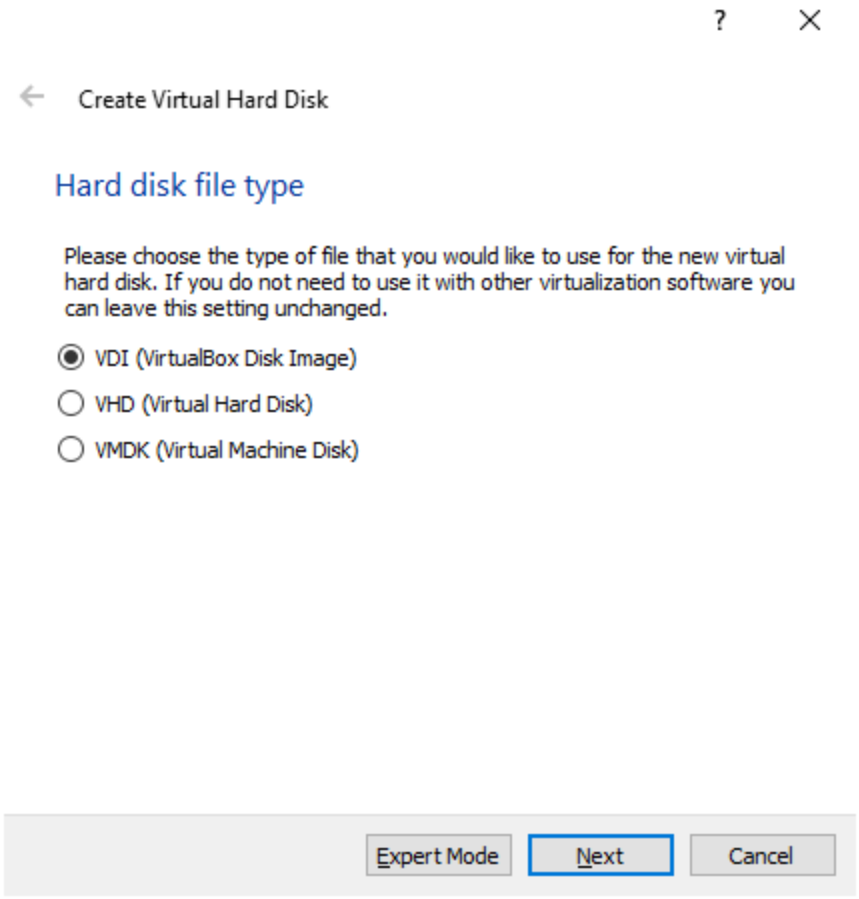
d. Next, select a memory size of 4096 MB.



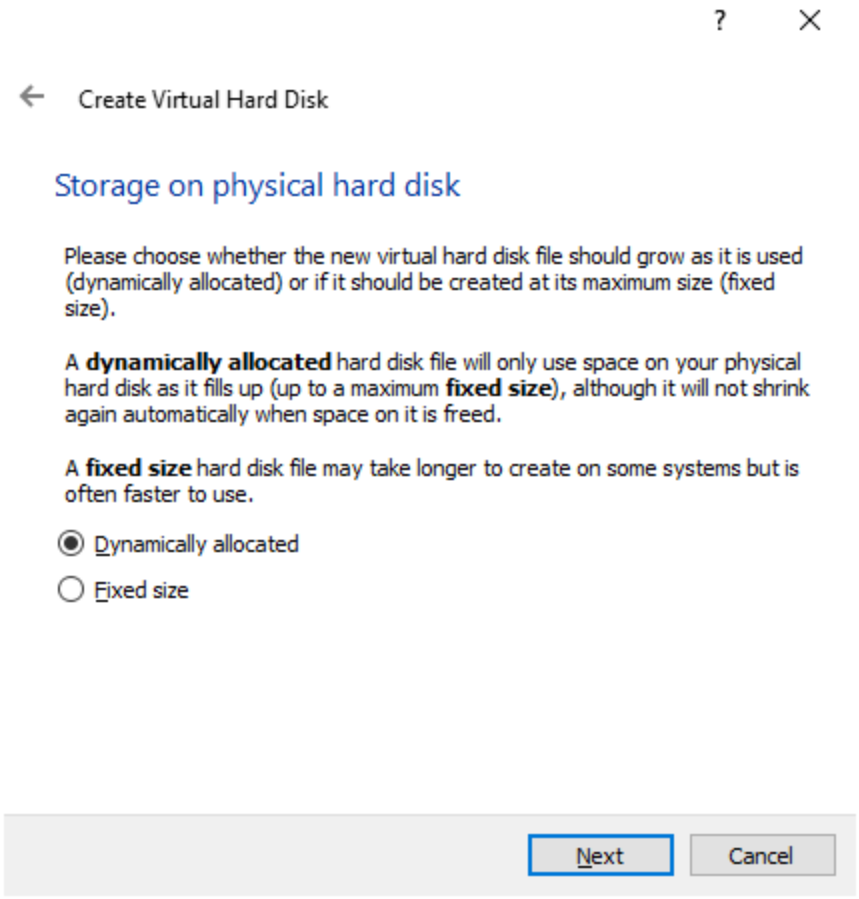
e. Then, create a hard disk for your virtual machine.



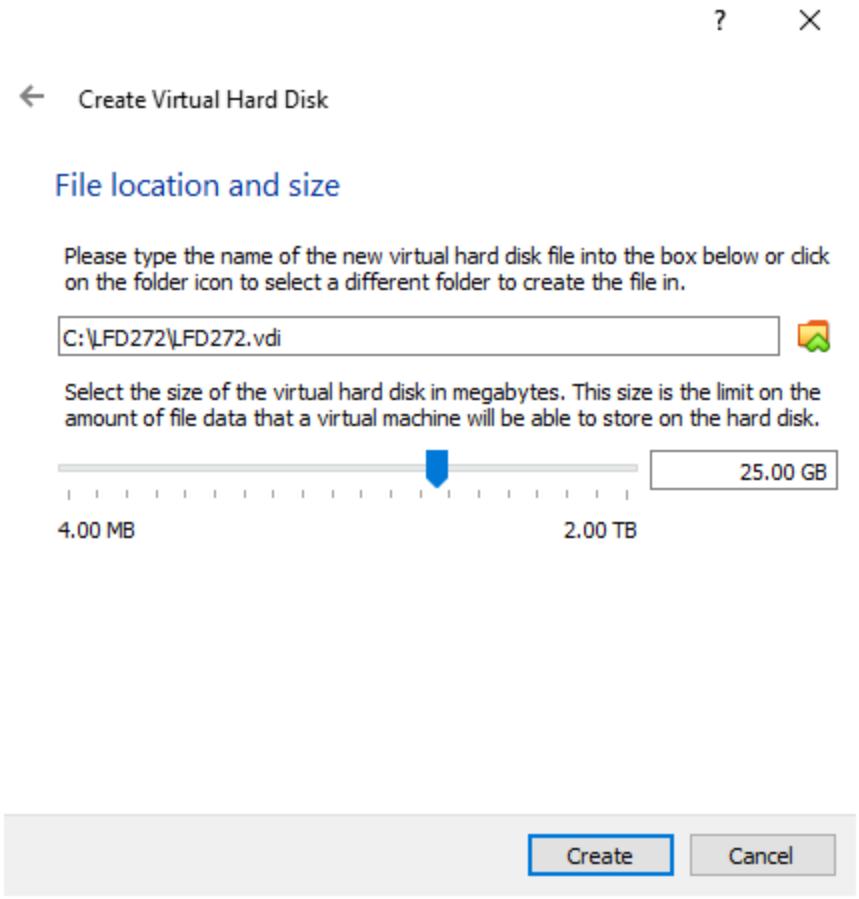
i. Select the **VDI** hard disk file type.



ii. Choose the **dynamically allocated** storage type.



iii. Finally, specify the disk file location and size (25 GB is recommended) and click on the **Create** button.

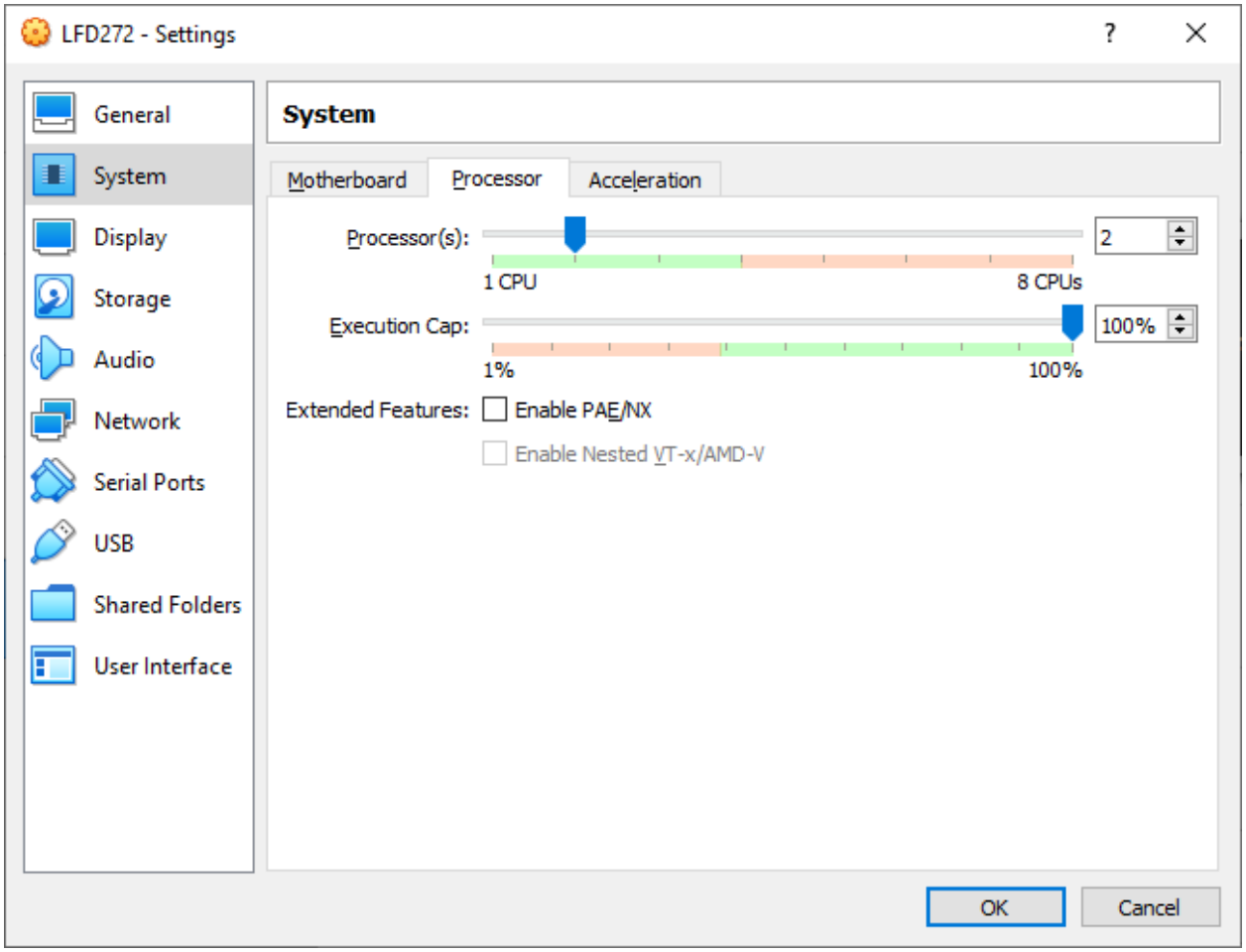


f. Once the hard disk file is created, click on the **Create** button to create a virtual machine.

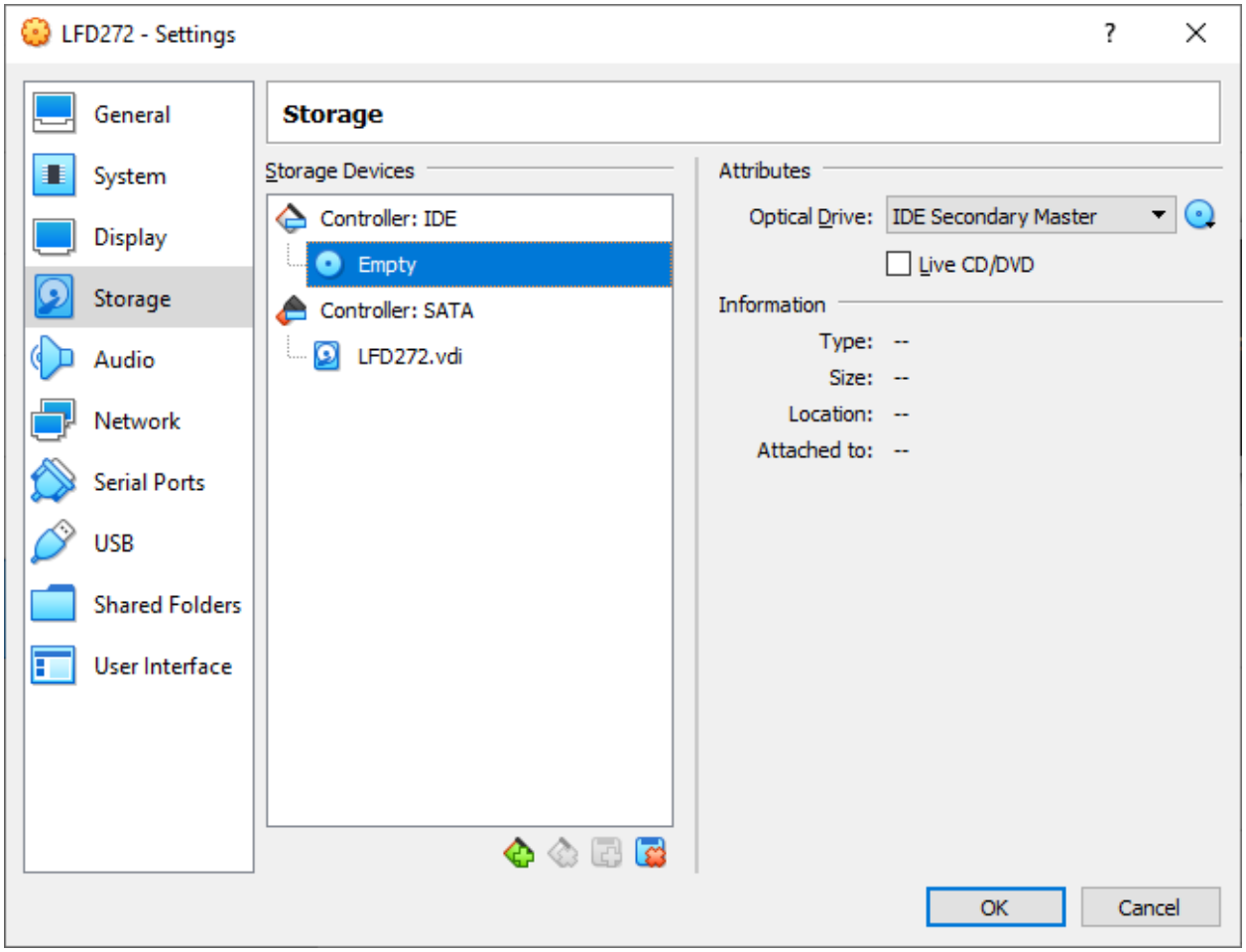
4. Configure a VM:

   a. Select a VM from the list in VirtualBox Manager and open the **Settings** menu (**Ctrl+S**).

   b. In the **Settings** window, go to **System** and select the **Processor** tab. Set the number of processors to 2.

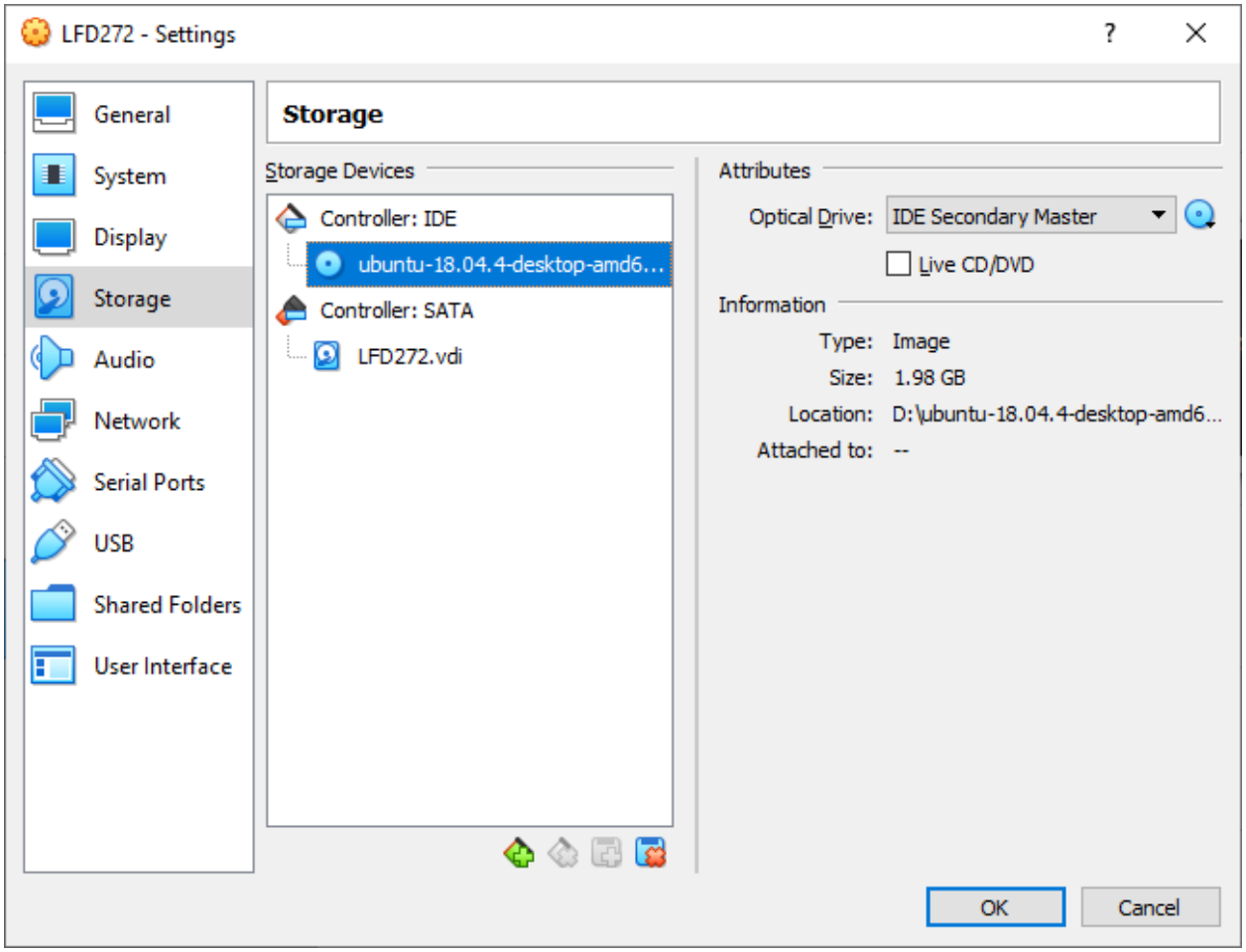


c. Next, go to **Storage** and set up the IDE controller.

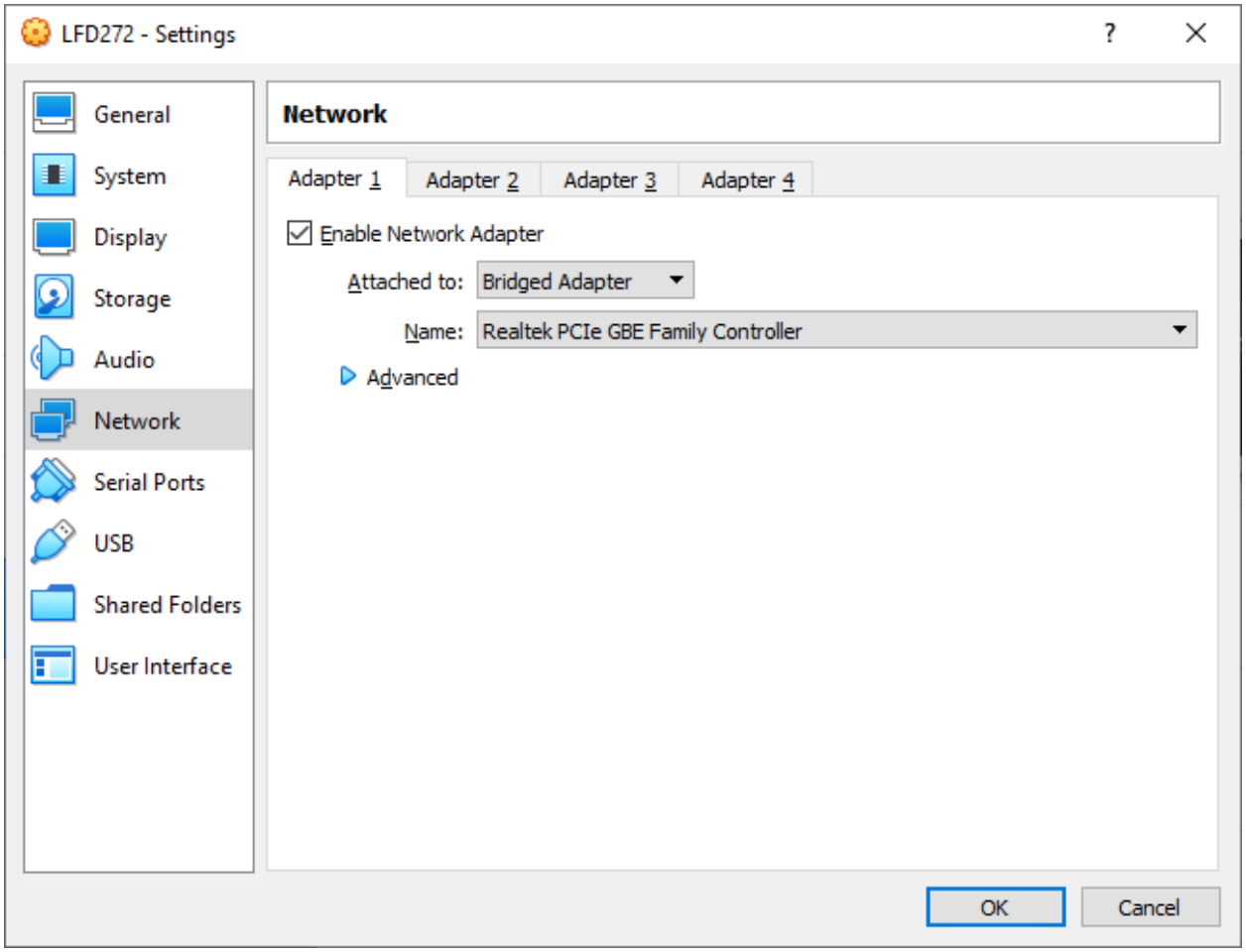


i. In the **Attributes** section on the right side of the window, click on the **CD** icon, and choose the virtual optical disk file.

ii. In the file system explorer, choose the .iso Ubuntu file that was downloaded in step 2.



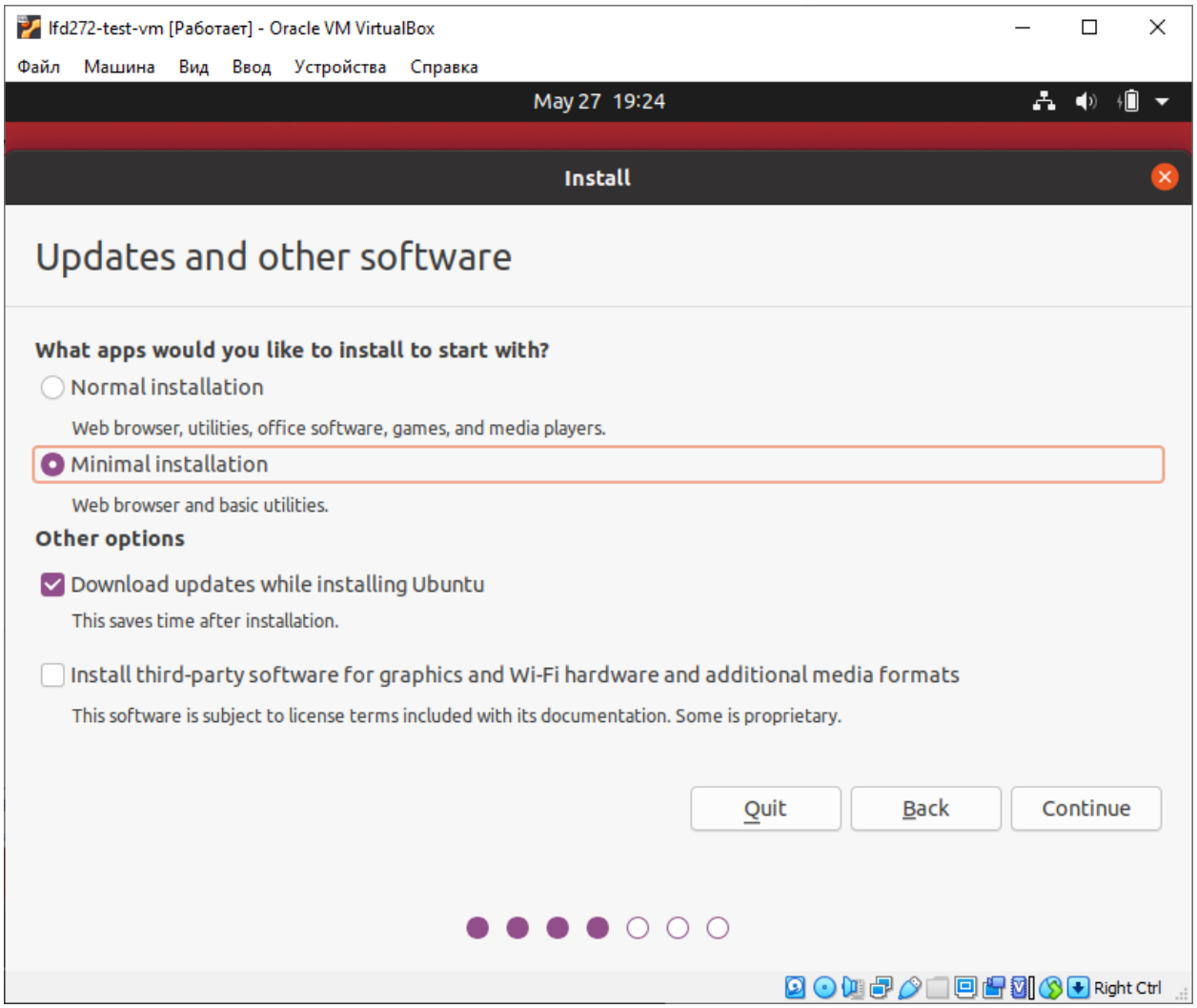
d. Go to **Network.** In the **Adapter 1** tab, choose **NAT** for the **Attached to** parameter.

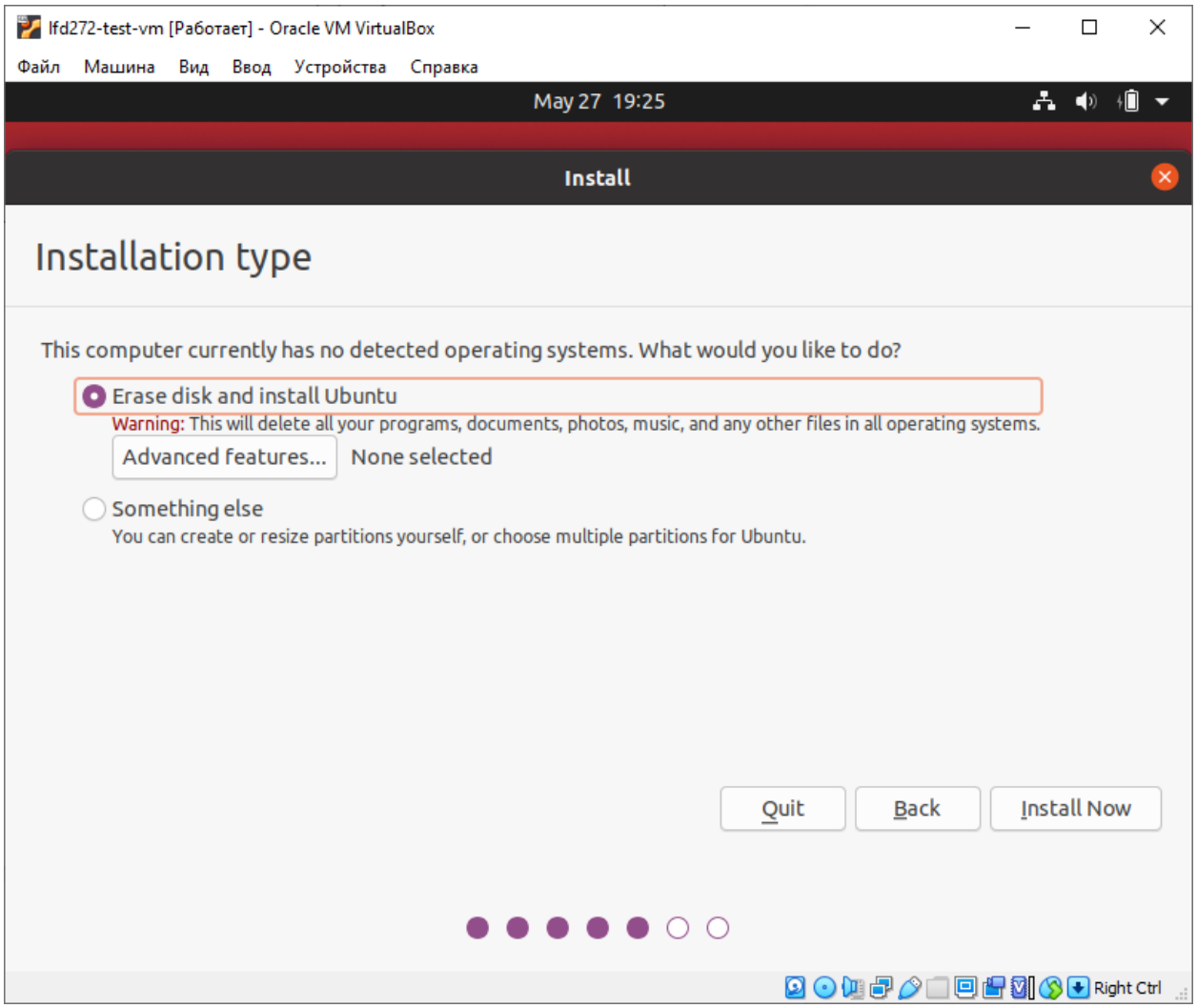


e. Click on the **OK** button to accept the changes.

5. Start the VM and go through the Ubuntu installation process.

Note: It is recommended to install the minimal version of the operating system. In addition, if you created the virtual hard drive file as shown above, you can safely select the **Erase disk and install Ubuntu** installation type.

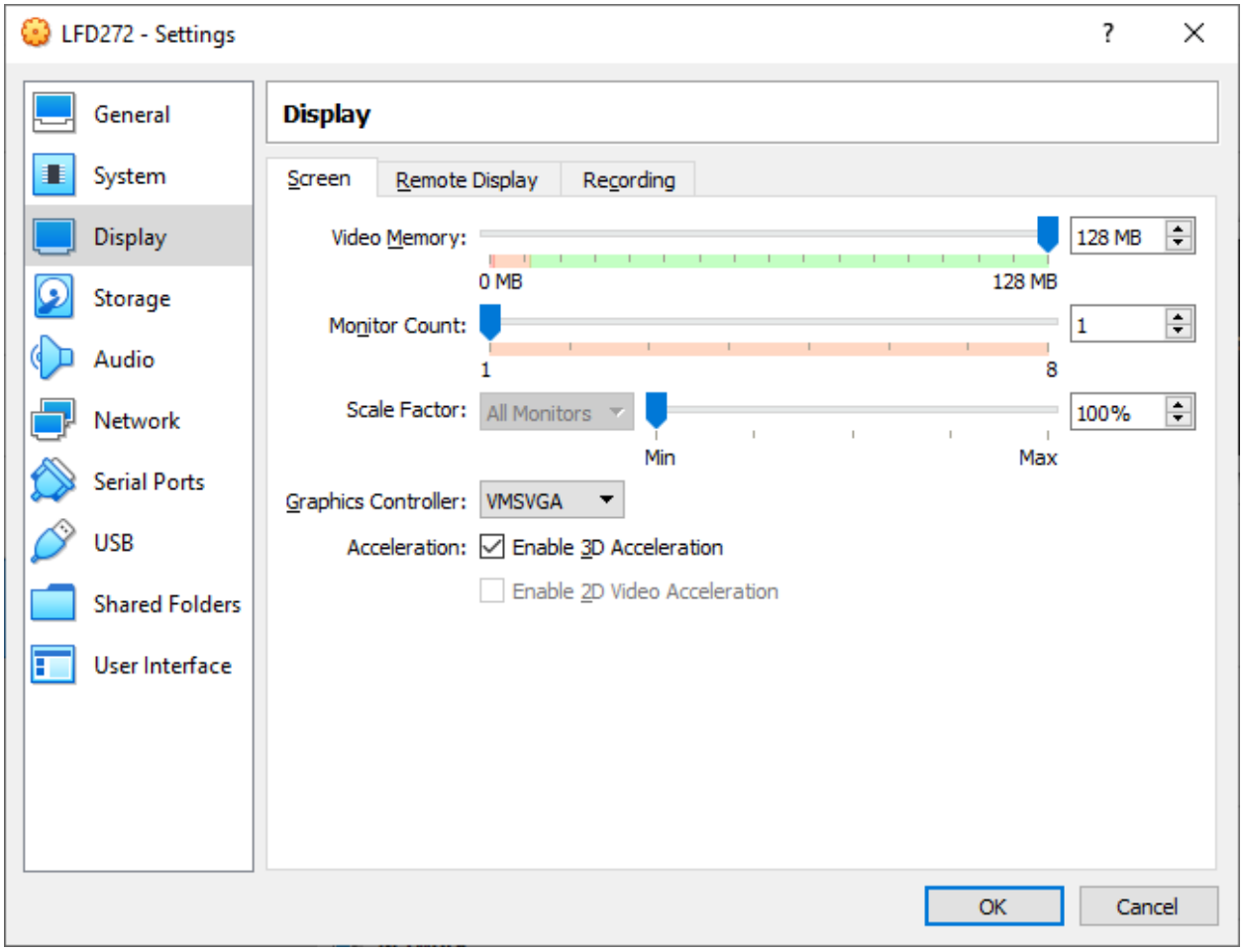




6. Enable full screen mode for the VM (optional).

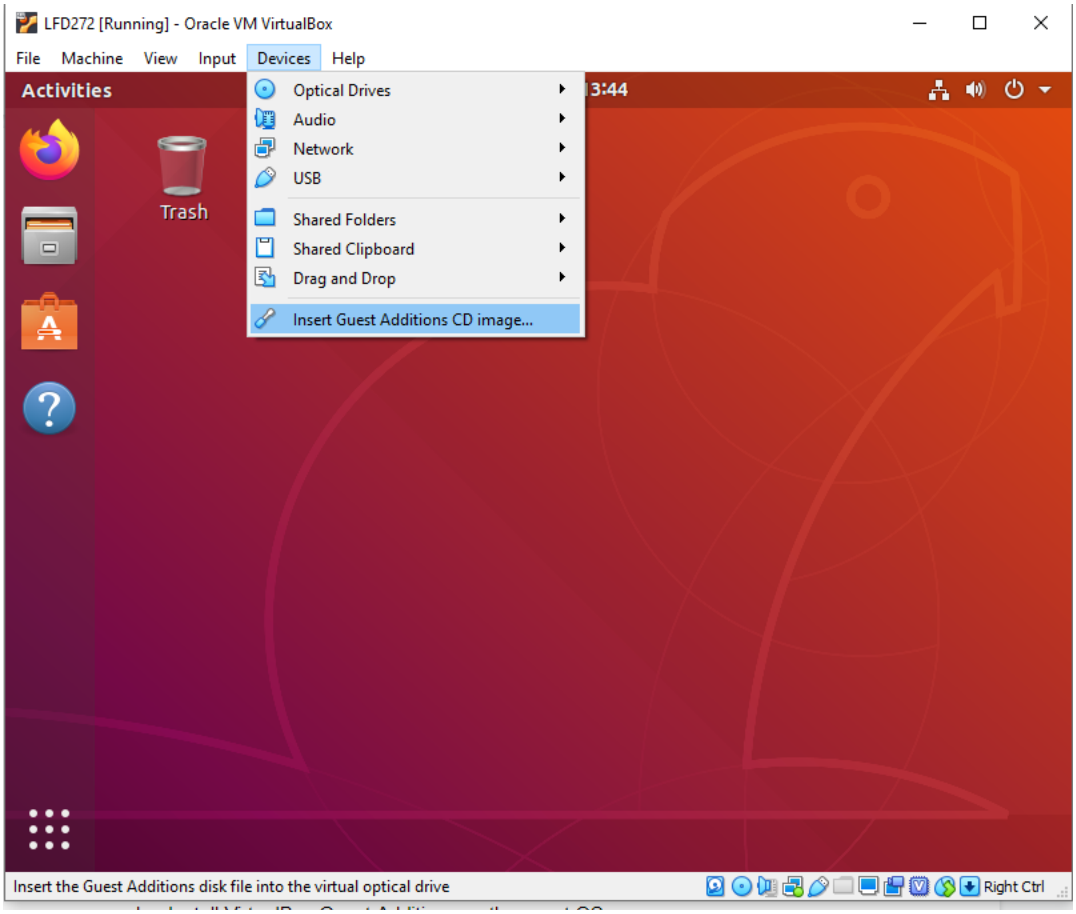
By default, the interface of the guest operating system (OS) will be displayed in windowed mode. The size of the window is not ideal and if you maximize the window, the resolution of the guest OS user interface will be the same. In most cases, working in a guest OS with low resolution is uncomfortable. In this section, you will learn how to make the VirtualBox full screen for the Ubuntu guest OS.

a. When your VM is powered off, go to **Settings** and set the video memory to 128 MB if possible. Select the **VMSVGA** graphics controller. Click on the **OK** button to save the changes.

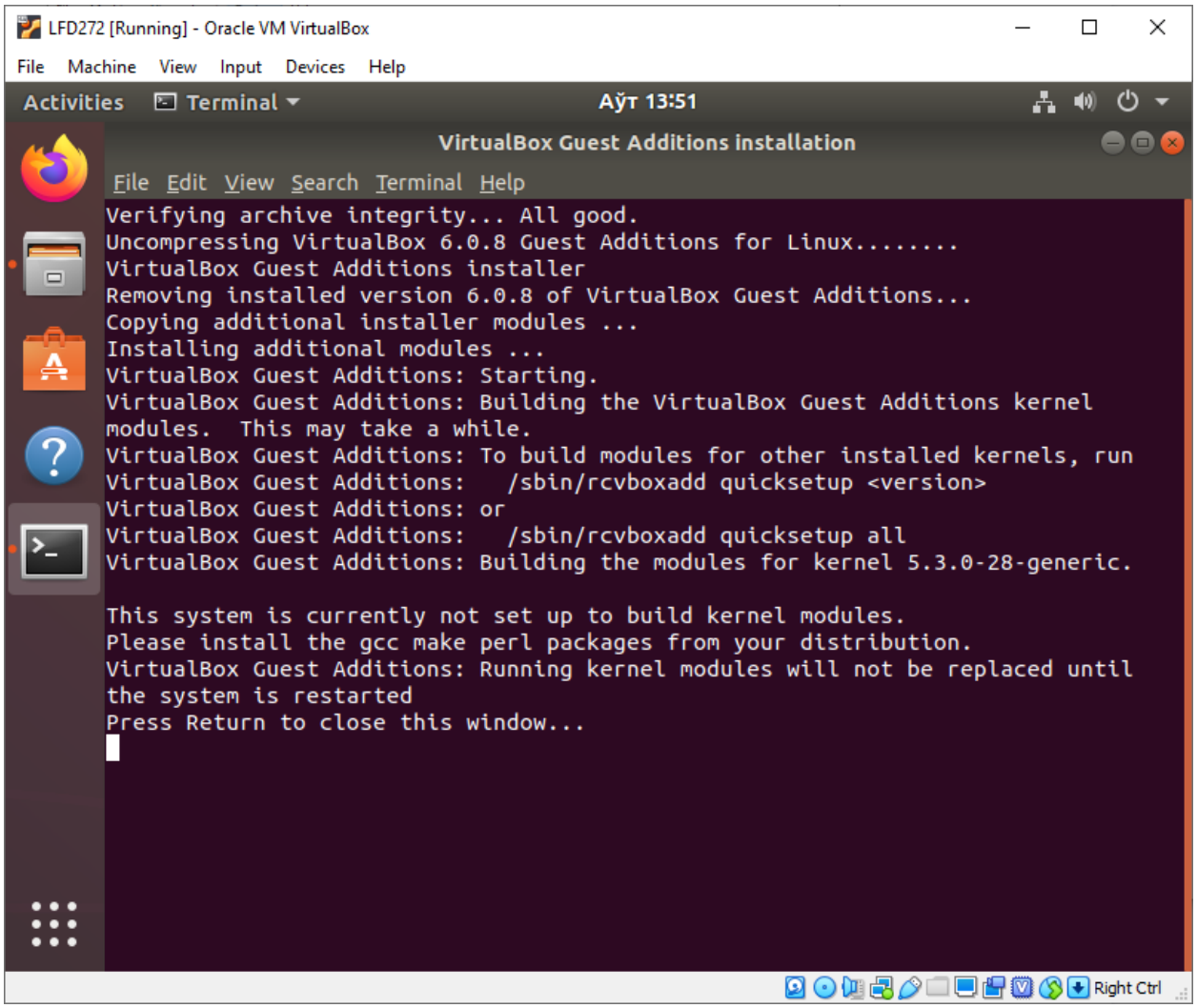


b. Install VirtualBox Guest Additions on the guest OS.

   i. Start the machine. In the VM window, go to **Devices,** click on **Insert Guest Additions CD image** to insert the ISO disk into the virtual CD drive of the VM.



ii. In the autorun window, click on the **Run** button and enter a password to confirm running the installer.



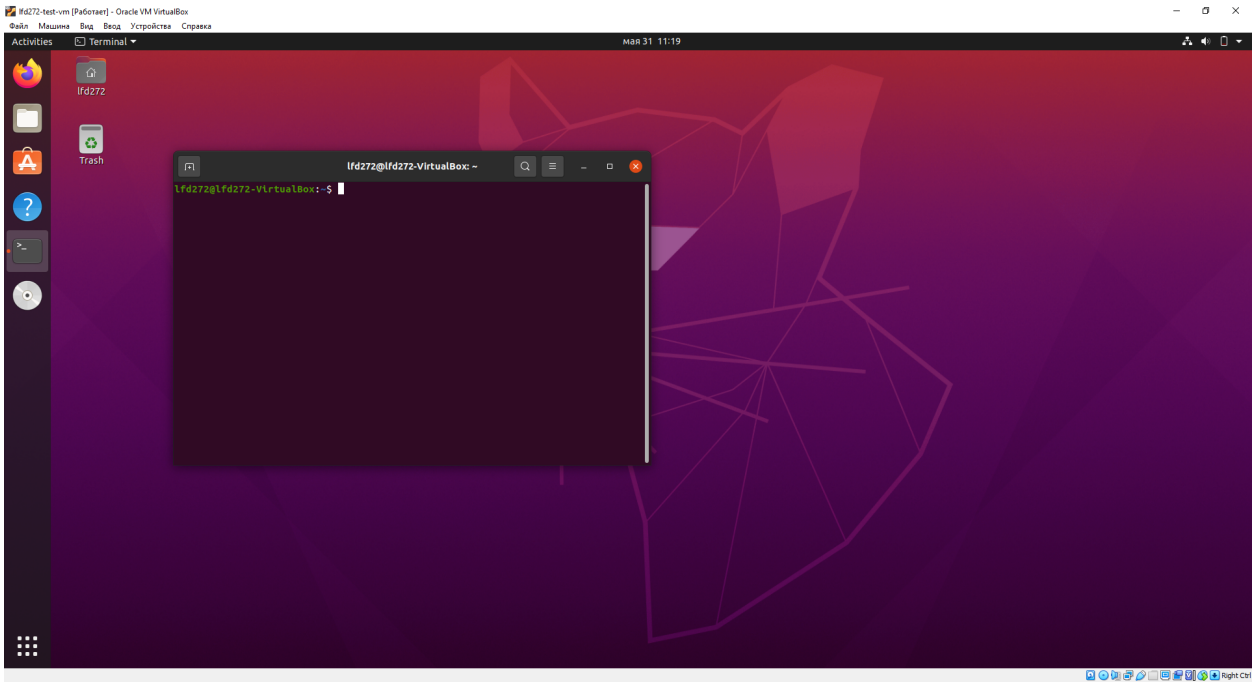
iii. Install the missing software packages.

```
# sudo apt update
# sudo apt-get install -y gcc make perl
```

iv. Go to a folder containing the VirtualBox Guest Additions files and run **`VBoxLinuxAdditions.run`**.

```
# cd /media/$(whoami)/VBox_GAs_6.1.18
# sudo ./VBoxLinuxAdditions.run
```

v. Reboot the VM. Now, you can enter full screen mode or resize the VM window, updating the guest OS interface accordingly.



Install software dependencies

git

To install `git`, run the following command in the terminal window.

```
# sudo apt install -y git
```

Docker

To properly install `Docker`, follow the official installation guide.
Make sure that the `Docker` version is `18.03` or greater by running

```
# docker --version
```

(Optional) If you do not want to preface the `docker` command with `sudo`, follow the Manage Docker as a non-root user guide.

docker-compose

To install `docker-compose`, run

```
# sudo apt install -y docker-compose
```

or follow the installation guide to obtain more recent versions.

Make sure that the `docker-compose` version is `1.14.0` or greater by running

```
# docker-compose --version
```

Node.js

To install Node.js version 12.x, run the following commands.

```
# curl -sL https://deb.nodesource.com/setup_12.x | sudo -E bash -
# sudo apt-get install -y nodejs
```

Golang (optional)

The course uses Node.js as the main programming language, but it can still be useful to have Golang installed to practice with examples provided by Hyperledger Fabric developers.

Download Golang version `1.15.x` or greater from the official website.
Navigate to the directory with the downloaded archive and extract it into the `/usr/local` directory.

```
# sudo tar -C /usr/local -xzf go1.16.linux-amd64.tar.gz
```

Note: In the command above, replace the archive name with the actual downloaded file name.

Add `/usr/local/go/bin` to the `PATH` environment variable. You can do this by adding this line to your `/etc/profile` (for a system-wide installation) or `$HOME/.profile`:

```
export PATH=$PATH:/usr/local/go/bin
```

Apply the changes.

```
# source $HOME/.profile
```

`build-essential`

`build-essential` is a package containing tools critical for building different software modules. To install the package, run the following commands.

```
# sudo apt update
```

```
# sudo apt install -y build-essential
```

Hyperledger Fabric samples, binaries, and Docker images

Run the Hyperledger Fabric bootstrap script from the `$HOME/go/src/github.com/hyperledger` folder.

```
# mkdir -p $HOME/go/src/github.com/hyperledger
# cd $HOME/go/src/github.com/hyperledger
# curl -sSL http://bit.ly/2ysbOFE | bash -s -- 2.2.2 1.4.9
```

The bootstrap script above clones the `fabric-samples` repository, downloads all necessary `Docker` images, and places the platform-specific binaries into the `bin` folder in the `fabric-samples` repo.

In the `fabric-samples`, checkout to the `v2.2.2` tag.

```
# cd fabric-samples
# git checkout v2.2.2
```

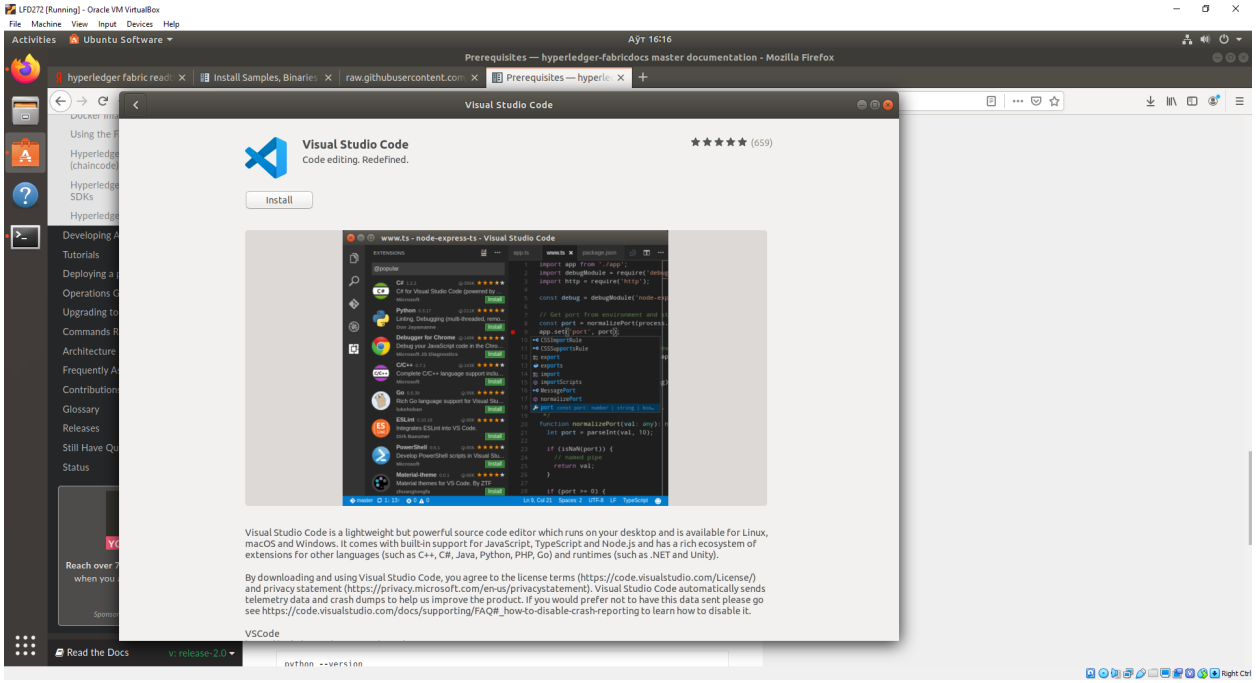
Next, clone the `fabric` repository and switch the branch to the `release-2.2`.

```
# cd ..
# git clone https://github.com/hyperledger/fabric
# cd fabric
# git checkout release-2.2
```

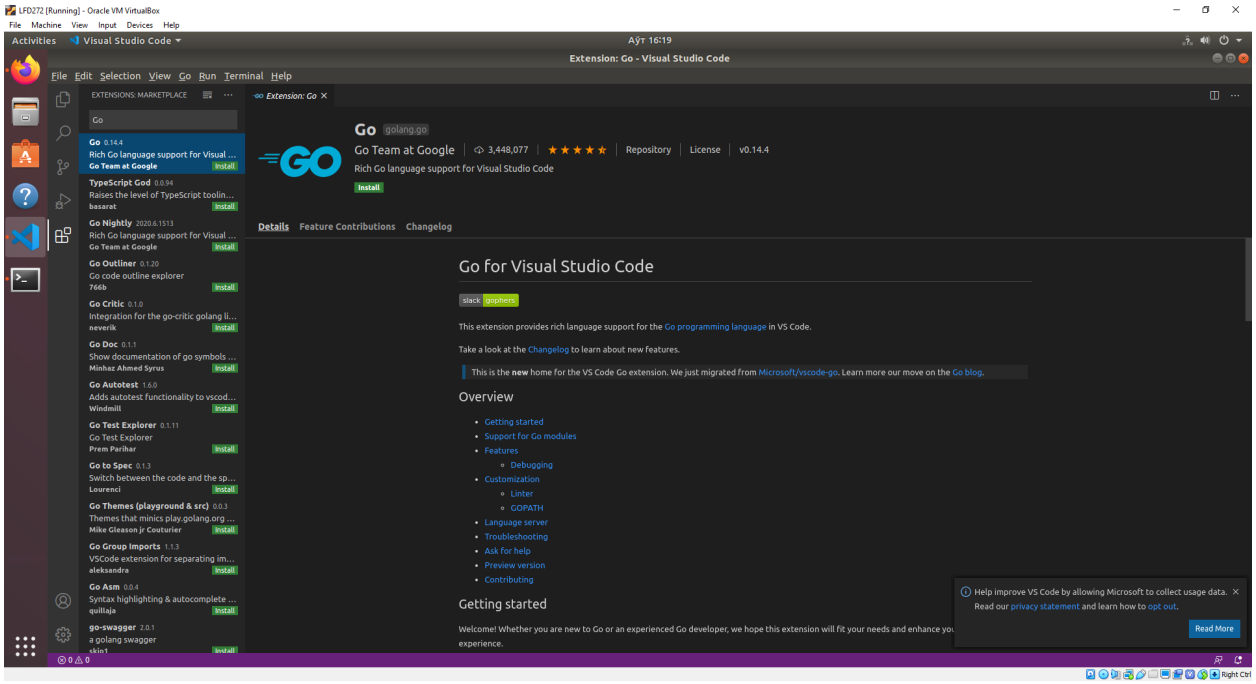
IDE

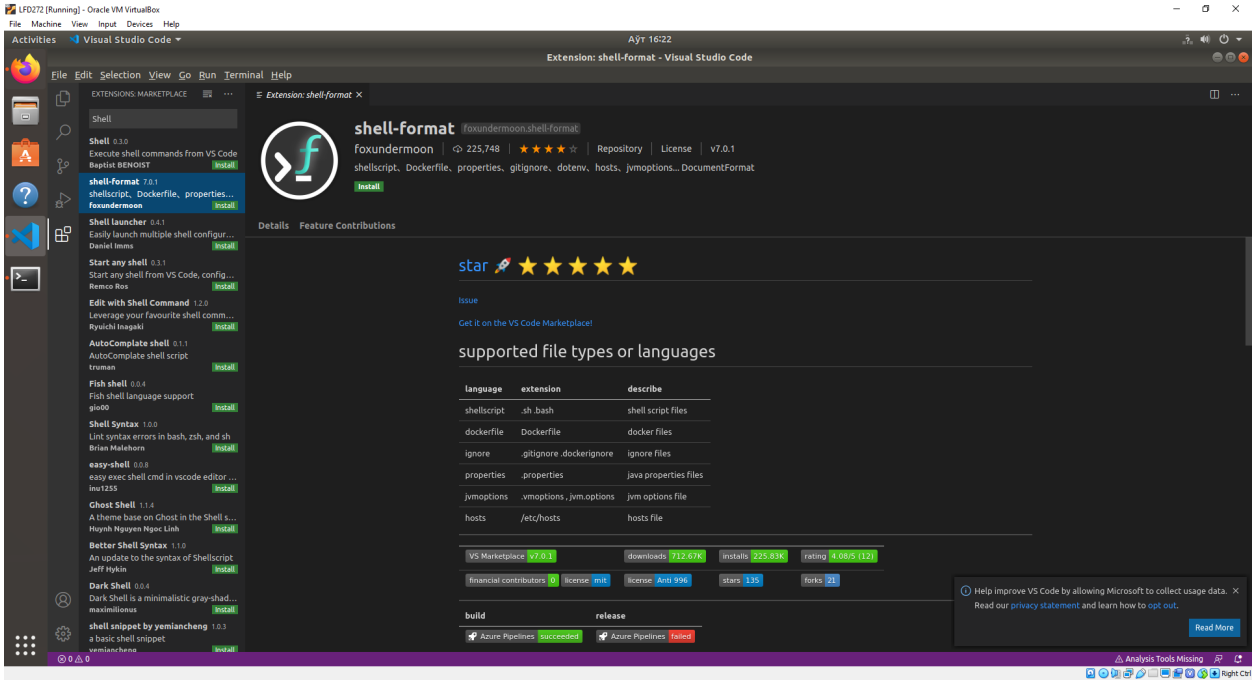
Note: IDE is a matter of choice. In this section, we describe how to install the Visual Studio Code inside of a VM. However, you can use any preferred tools, establish Secure Shell access between your host machine and a VM, mount workspaces, etc.

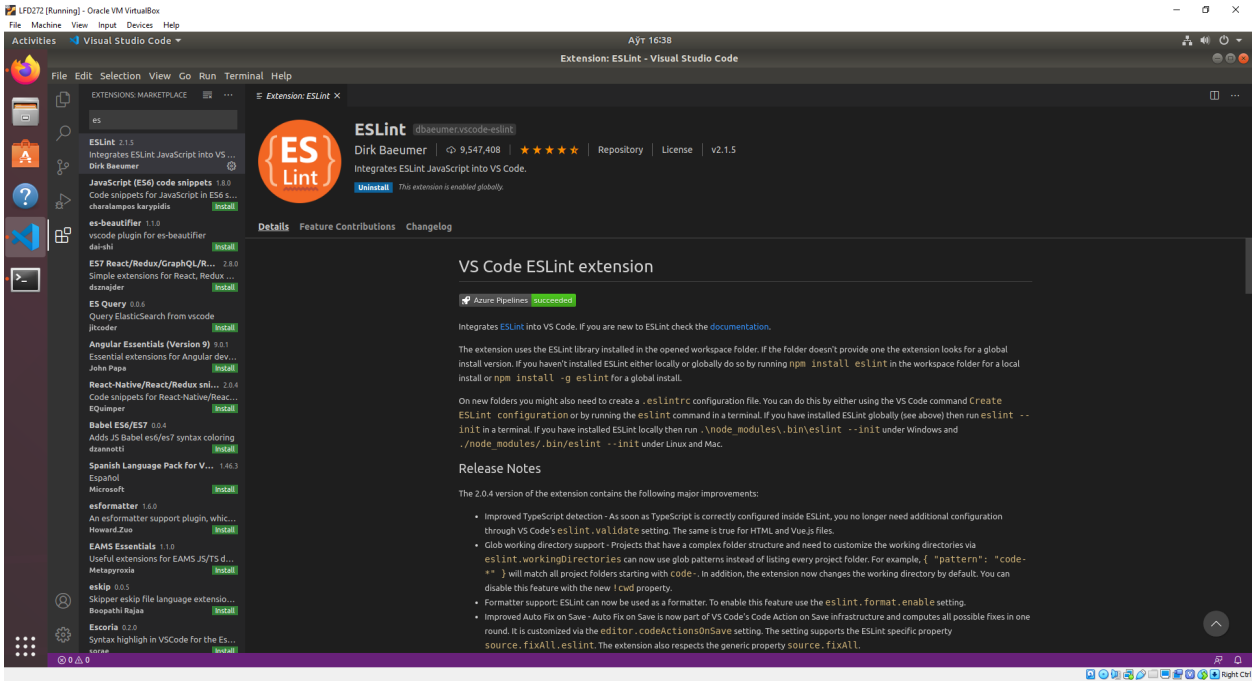
1. Go to Ubuntu Software, find Visual Studio Code there, and click install.

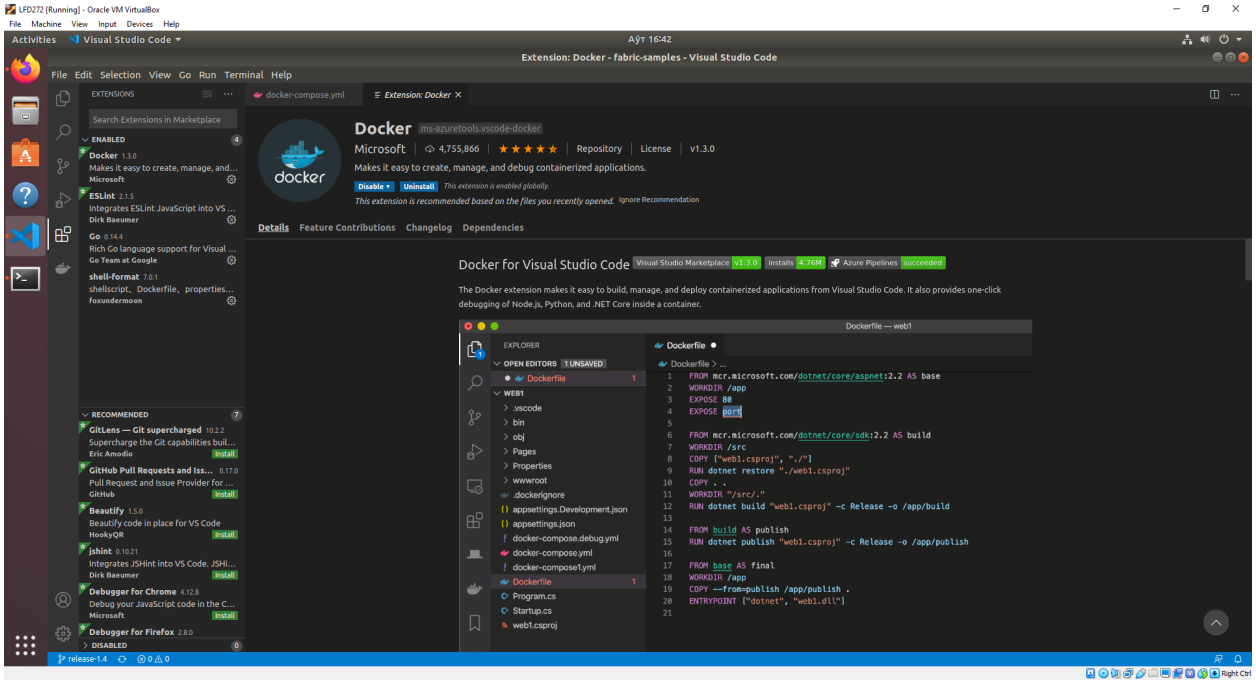


2. Open Visual Studio Code and go to the Extensions (**Ctrl+Shift+X**) tab on the left side of the window. Find and install the plugins for Go, JavaScript, Shell script, and Docker.









Run the Hyperledger Fabric network

`test-network`

Spin up the network

1. Navigate to the `test-network` folder.

   ```
   # cd ~/go/src/github.com/hyperledger/fabric-samples/test-network/
   ```

2. Spin up the network. As a default configuration, we will use a network with a predefined `mychannel` channel, CouchDB-based world state, and certificate authorities as identity providers.

   ```
   # ./network.sh up createChannel -ca -s couchdb
   ```

Add Org3 (optional)

To add the third organization to the network, follow the steps below.

1. Spin up the network as shown above.

2. Navigate to the `test-network/addOrg3` folder.

```
# cd
~/go/src/github.com/hyperledger/fabric-samples/test-network/addOrg3
```

3. Extend the network.

```
# ./addOrg3.sh up -ca -s couchdb
```

Clean network artifacts

To clean network artifacts, follow the steps below.

1. Navigate to the `test-network` folder.

```
# cd ~/go/src/github.com/hyperledger/fabric-samples/test-network/
```

2. Clean Org3 artifacts, if necessary.

```
# ./addOrg3/addOrg3.sh down
```

3. Clean remaining network artifacts.

```
# ./network.sh down
```