Guide to construct Ethereum Local Testnet

1. **Install Docker Engine on Ubuntu**

[OS requirements](https://docs.docker.com/engine/install/ubuntu/#os-requirements)

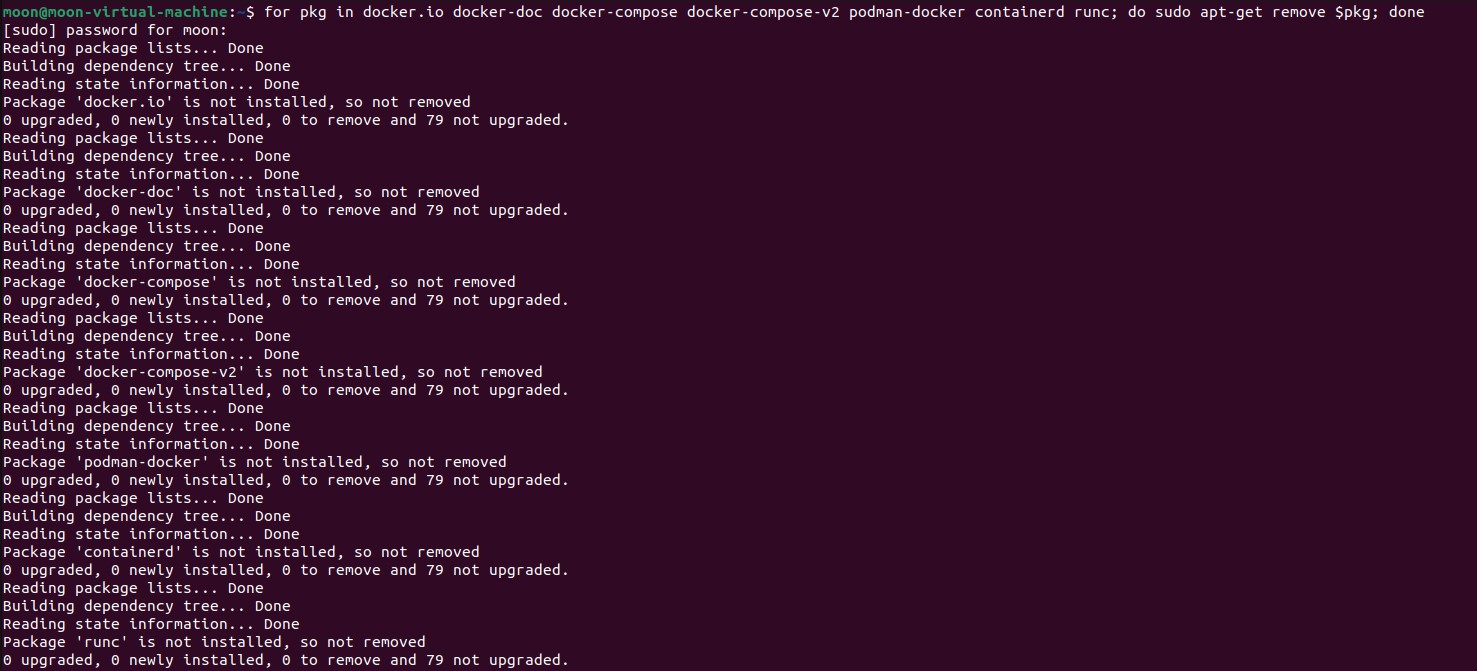
To install Docker Engine, you need the 64-bit version of one of these Ubuntu versions:

* Ubuntu Noble 24.04 (LTS)
* Ubuntu Jammy 22.04 (LTS)
* Ubuntu Focal 20.04 (LTS)

Docker Engine for Ubuntu is compatible with x86\_64 (or amd64), armhf, arm64, s390x, and ppc64le (ppc64el) architectures.

* Uninstall old versions

Run the following command to uninstall all conflicting packages:

`for pkg in docker.io docker-doc docker-compose docker-compose-v2 podman-docker containerd runc; do sudo apt-get remove $pkg; done`

* Install using the apt repository

1. Set up Docker's apt repository.

`# Add Docker's official GPG key:

sudo apt-get update

sudo apt-get install ca-certificates curl

sudo install -m 0755 -d /etc/apt/keyrings

sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/docker.asc

sudo chmod a+r /etc/apt/keyrings/docker.asc

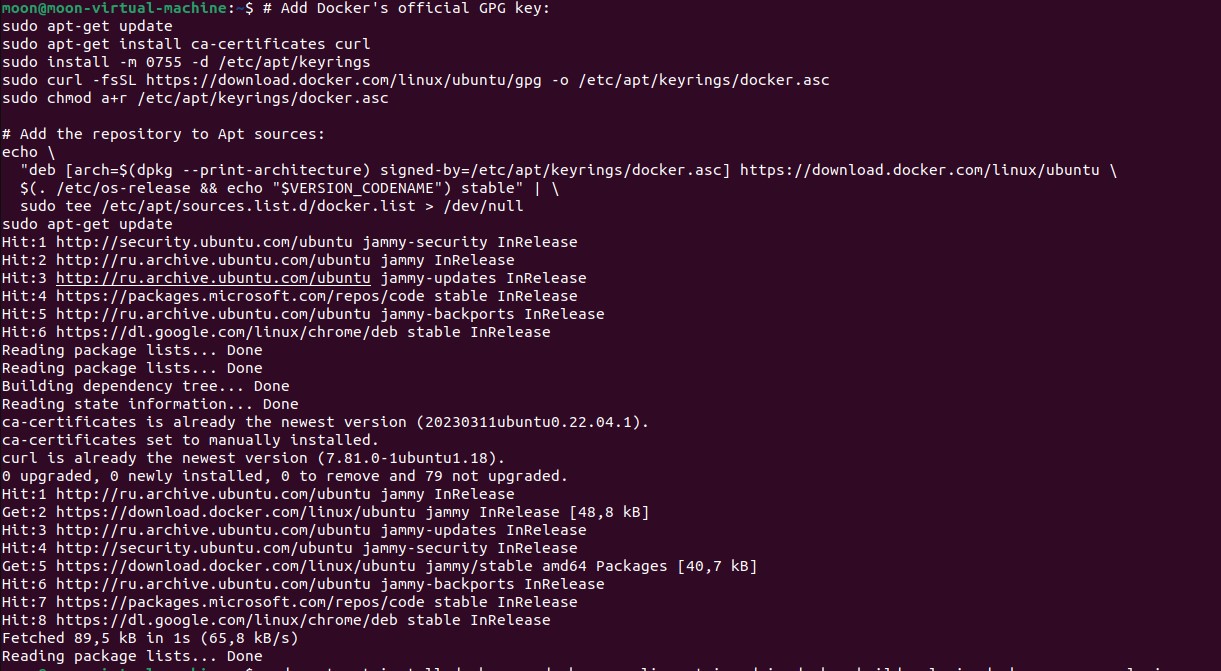
# Add the repository to Apt sources:

echo \

"deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] https://download.docker.com/linux/ubuntu \

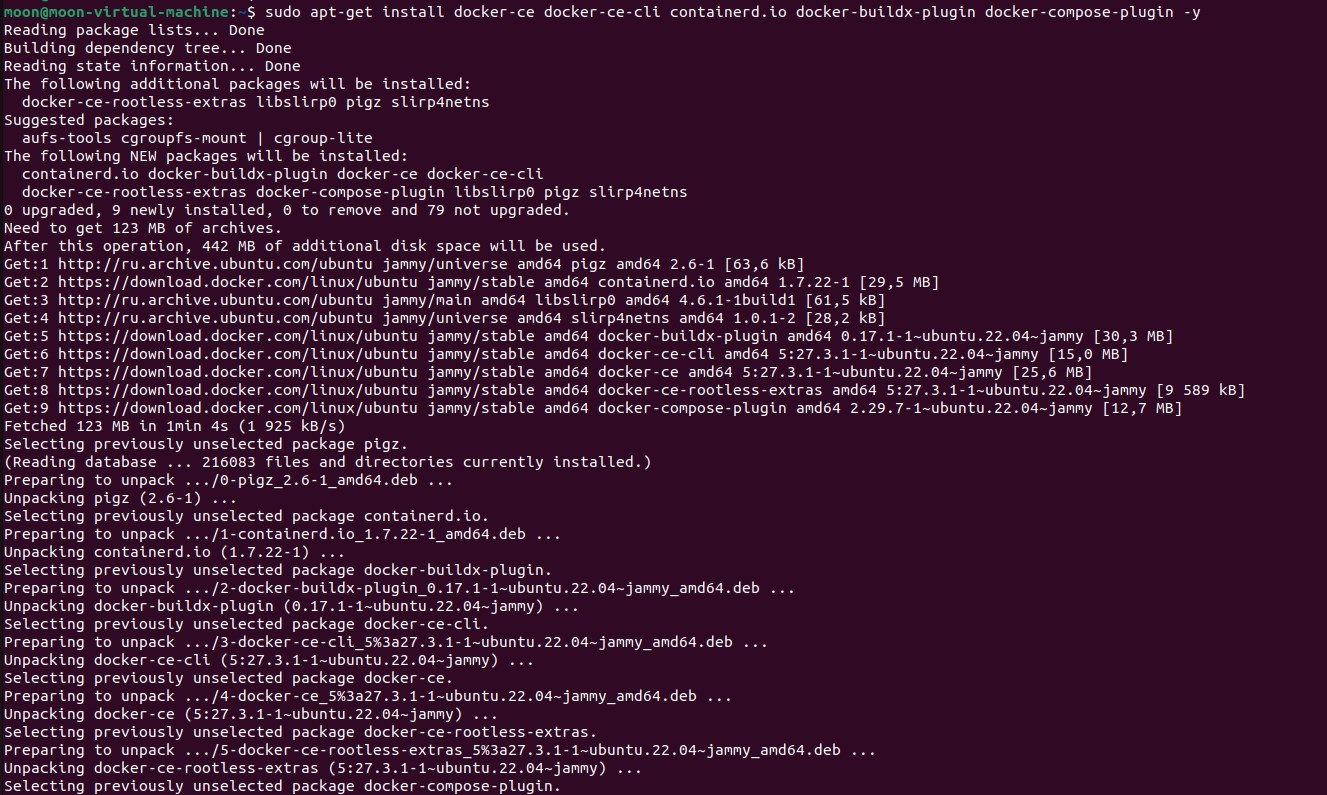
$(. /etc/os-release && echo "$VERSION\_CODENAME") stable" | \

sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

sudo apt-get update`

1. Install the Docker packages.

`sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin -y`

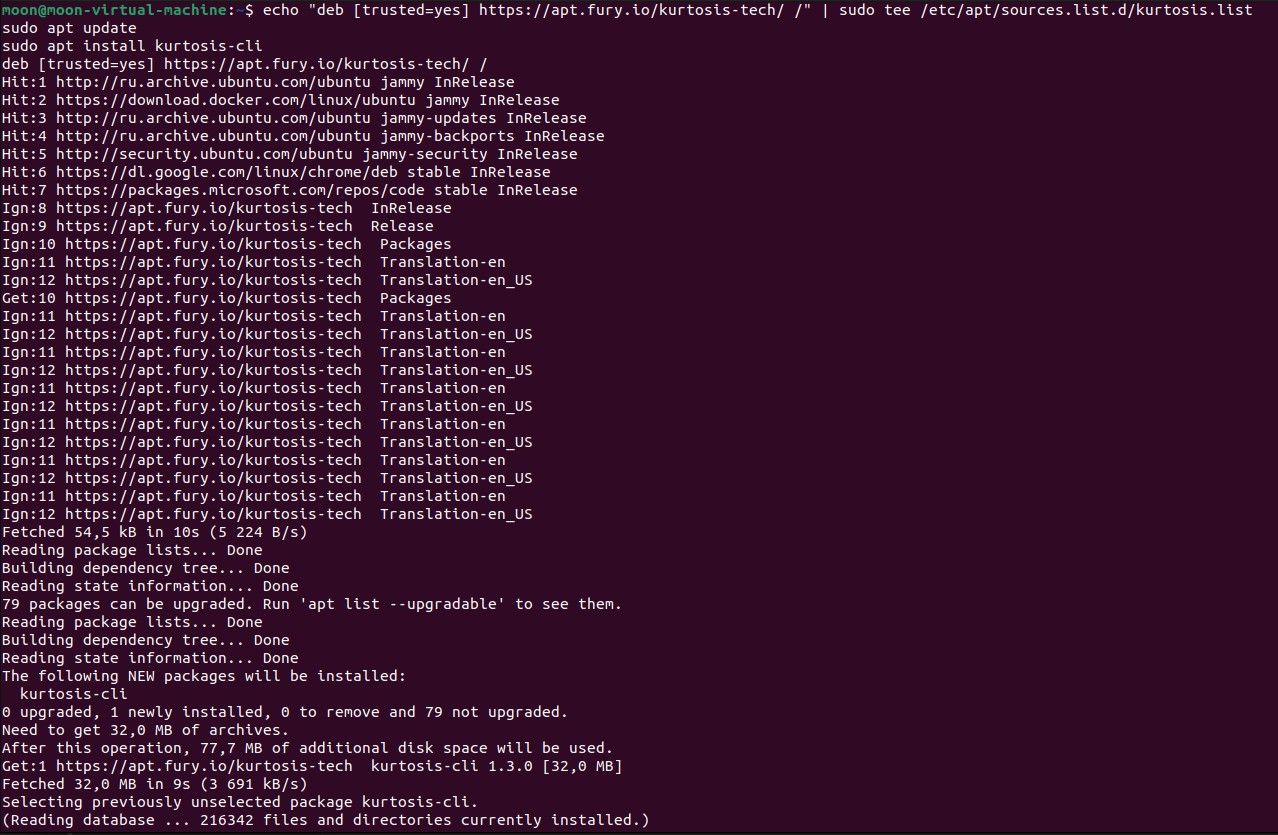
This command downloads a test image and runs it in a container. When the container runs, it prints a confirmation message and exits.

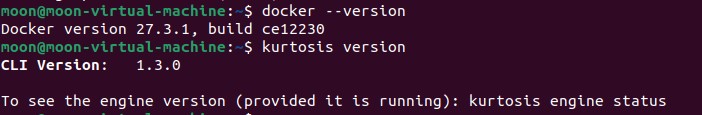
1. **Install Kurtosis-cli on Ubuntu**

`echo "deb [trusted=yes] https://apt.fury.io/kurtosis-tech/ /" | sudo tee /etc/apt/sources.list.d/kurtosis.list

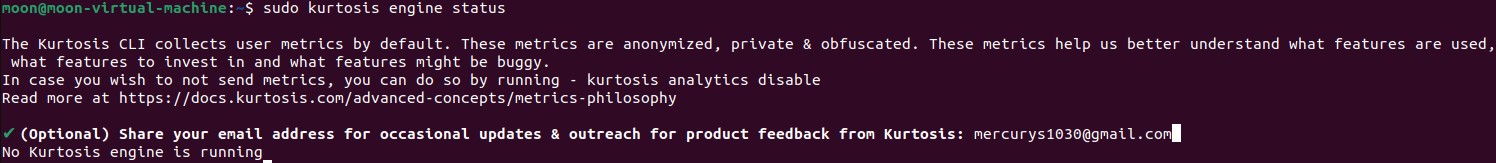
sudo apt update

sudo apt install kurtosis-cli`

This command downloads kurtosis-cli and install on ubuntu.

Check docker and kurtosis

And run:

`sudo kurtosis engine status`



Configure email

1. **Install** [**Node.js**](https://nodejs.org/en)**,** [**yarn**](https://classic.yarnpkg.com/lang/en/docs/install/#mac-stable)**, git and** [**npx**](https://www.npmjs.com/package/npx) **Ubuntu**

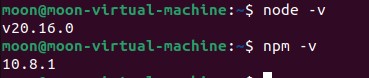
* Install Node.js
* Install Node.js and NPM ([Node Package Manager)](https://www.npmjs.com/):

`sudo apt install -y nodejs npm`

* Once the install is complete, verify your Node.js installation:

`node -v`

`npm -v`



* Install yarn
* Update Your System

` sudo apt update`

* Install Curl

` sudo apt install curl -y`

* Add the Yarn GPG Key

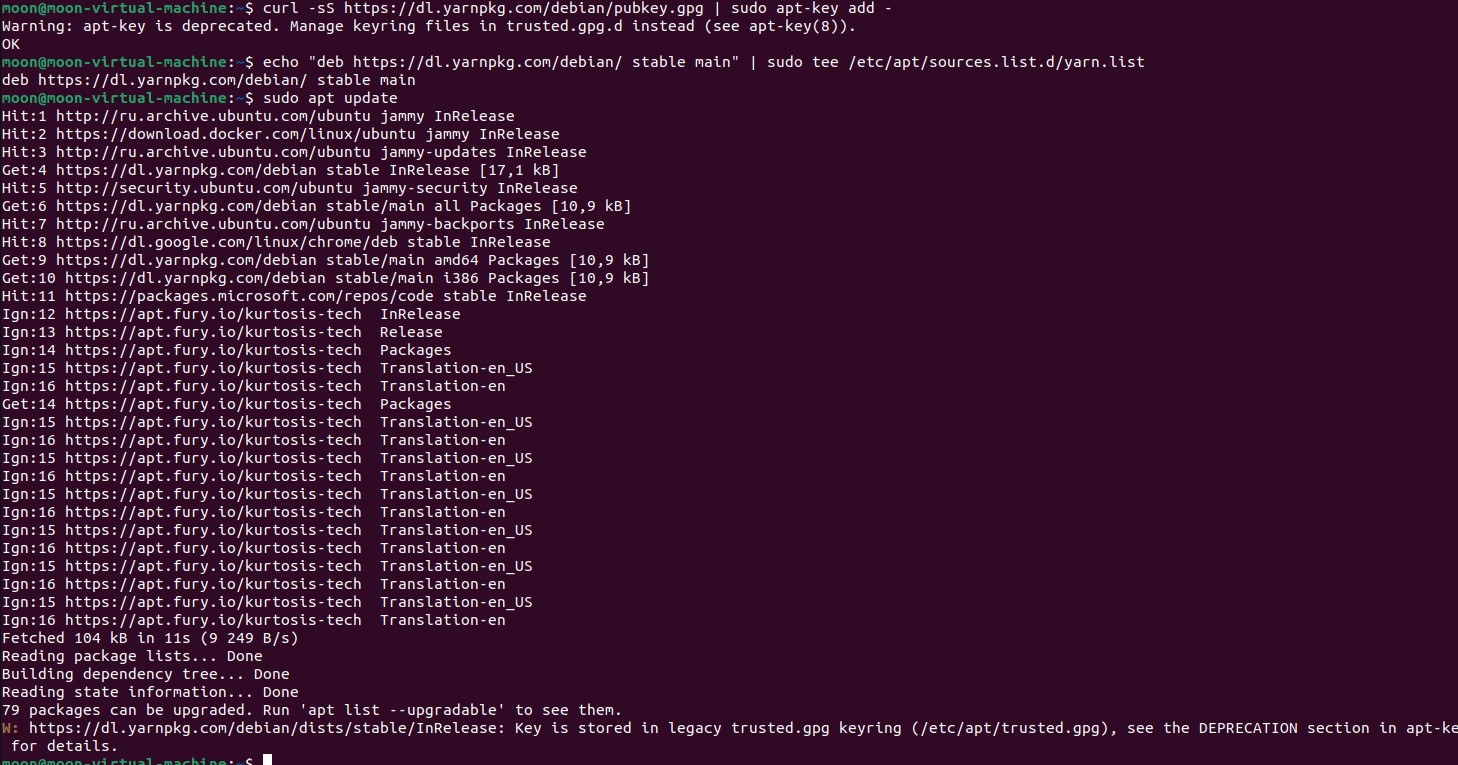
` curl -sS https://dl.yarnpkg.com/debian/pubkey.gpg | sudo apt-key add -`

* Add the Yarn Repository

`echo "deb https://dl.yarnpkg.com/debian/ stable main" | sudo tee /etc/apt/sources.list.d/yarn.list`

* Update Package List Again

` sudo apt update`



* Install Yarn

` sudo apt install yarn -y`

` sudo apt install yarn@4.5.0 -y`

* Verify Installation

` yarn --version`

* Install Git and configure name and email

`sudo apt update`

`sudo apt install git -y`

`git --version`

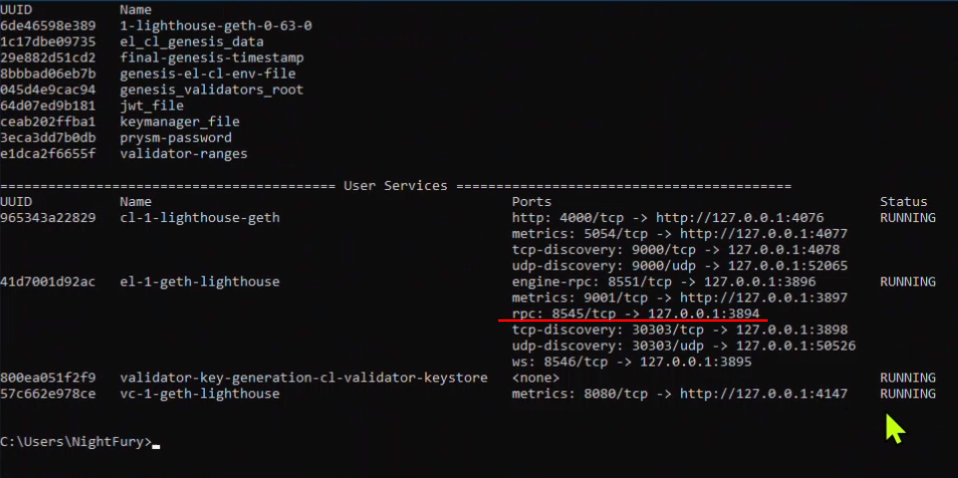
`git config –global user.name <USERNAME> `

`git config –global user.email <USEREMAIL> `

1. **Instantiating a local Ethereum testnet**

To spin up a local Ethereum testnet, run:

`sudo kurtosis --enclave local-eth-testnet run github.com/ethpandaops/ethereum-package`



1. **Connect your dApp development environment to the local Ethereum testnet**

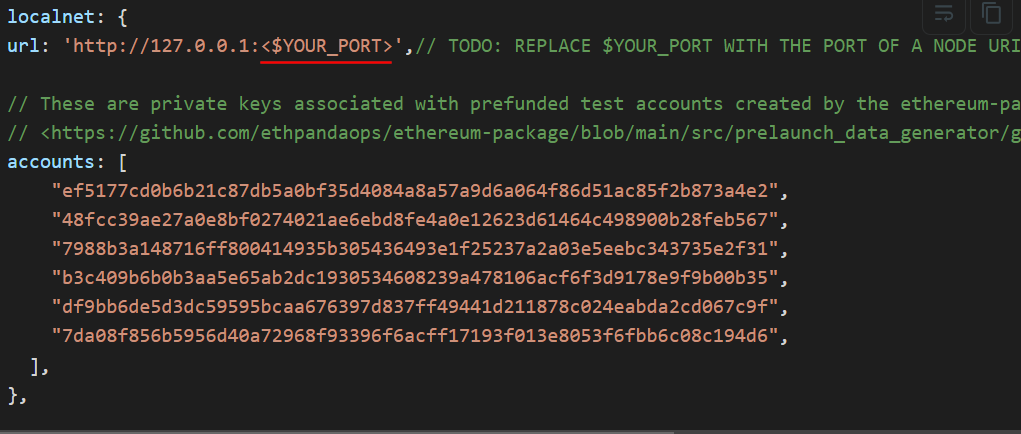
* Setup the dApp development environment

Now that you have a running local testnet, you can connect your dApp development environment to use your local testnet. The Hardhat framework will be used in this guide to deploy a blackjack dApp to your local testnet.

To set up your dApp development environment, clone the repository that contains our sample dApp and install its dependencies, run:

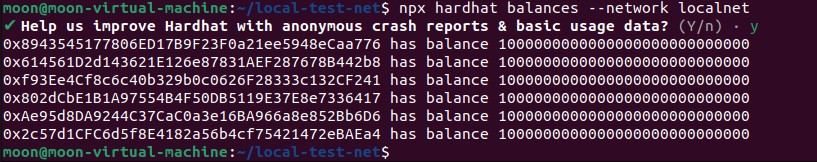
` git clone -b subgraph-automatic https://github.com/devstar1014/local-test-net.git && cd local-test-net && npm install`

* Configure Hardhat to use the local testnet



`$YOUR\_PORT` is the port number of the rpc which mentioned above `**4** Instantiating a local Ethereum testnet`

And run the command:

`sudo npx hardhat balances --network localnet`

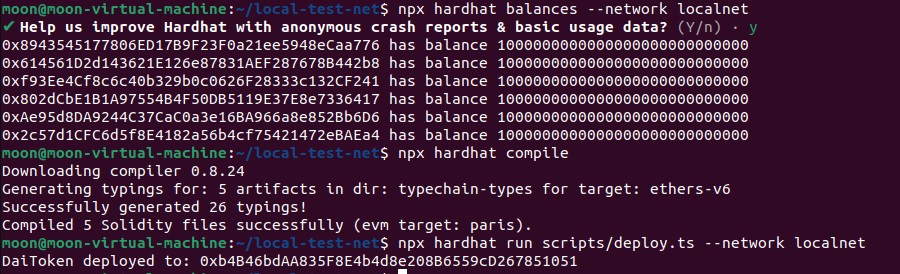
* Deploy and test your dApp locally

With the dApp development environment fully connected to the local Ethereum testnet, you can now run development and testing workflows against your dApp using the local testnet.

To compile and deploy the `DAIToken.sol` smart contract for local prototyping and development, run:

`npx hardhat compile`

`npx hardhat run scripts/deploy.ts --network localnet`



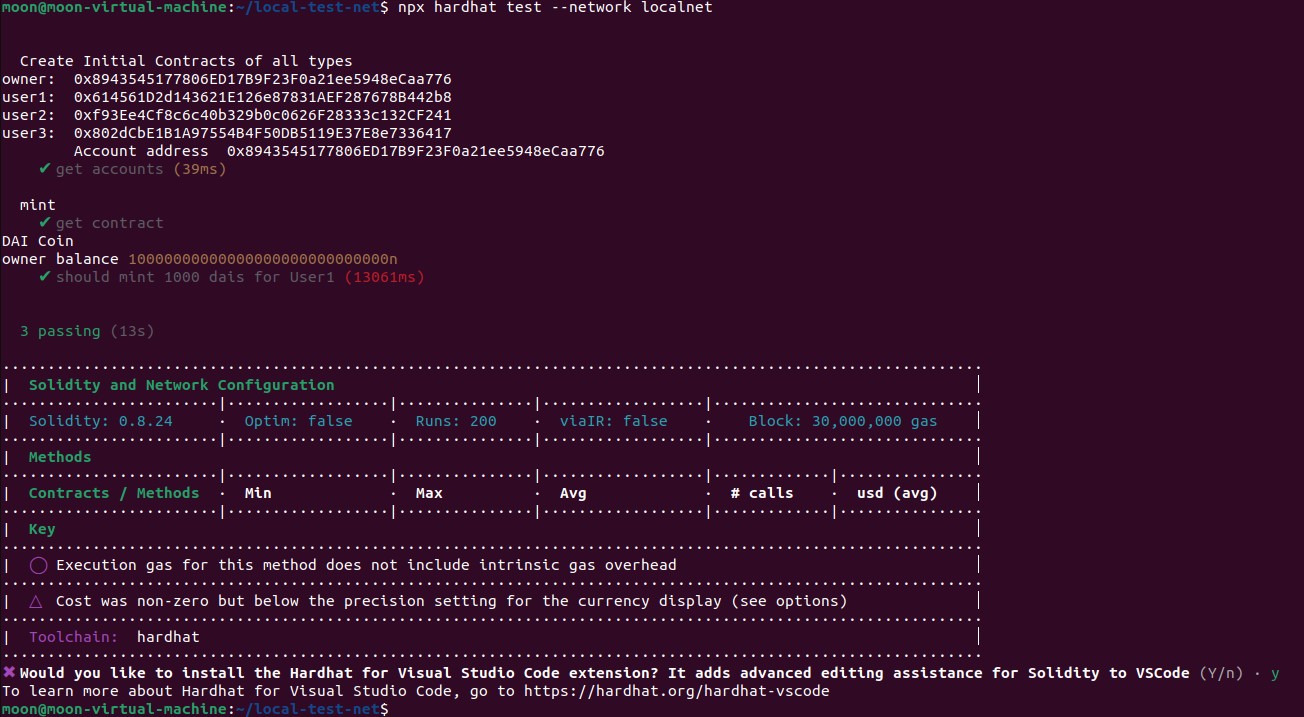


The output should look something like:



`DaiToken deployed to: 0xAb2A01BC351770D09611Ac80f1DE076D56E0487d`

The output should look something like this:

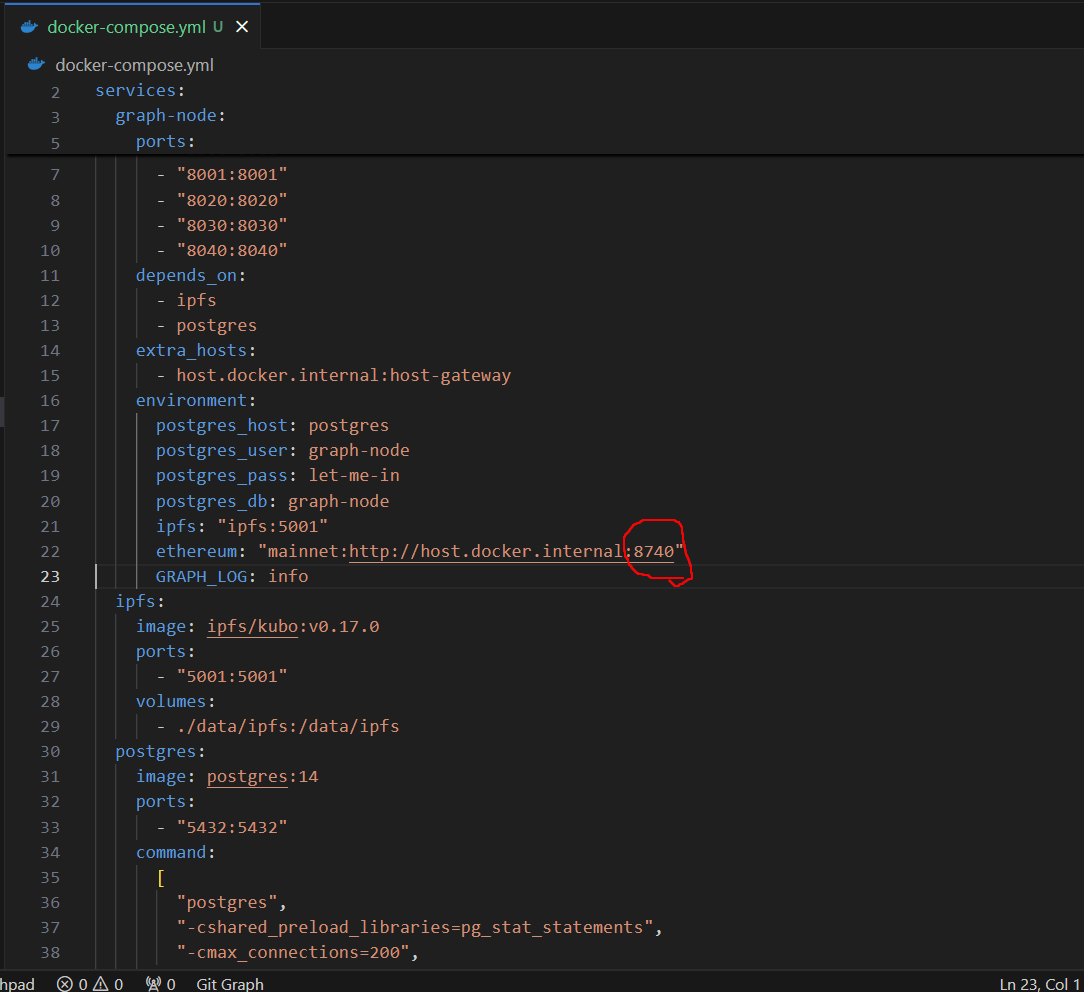
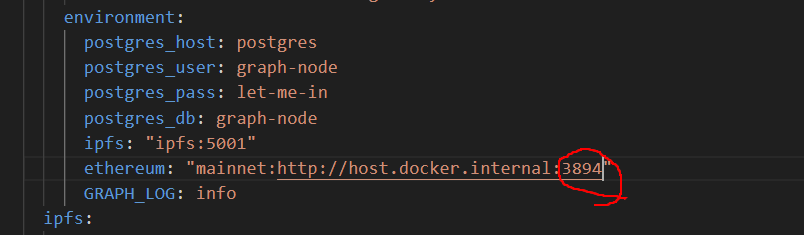
`npx hardhat test --network localnet`

Before this, you should replace the address of `DaiTokenContractAddress` with the address that deployed into localnet.

1. **Deploy `graph-node` and test subgraph**

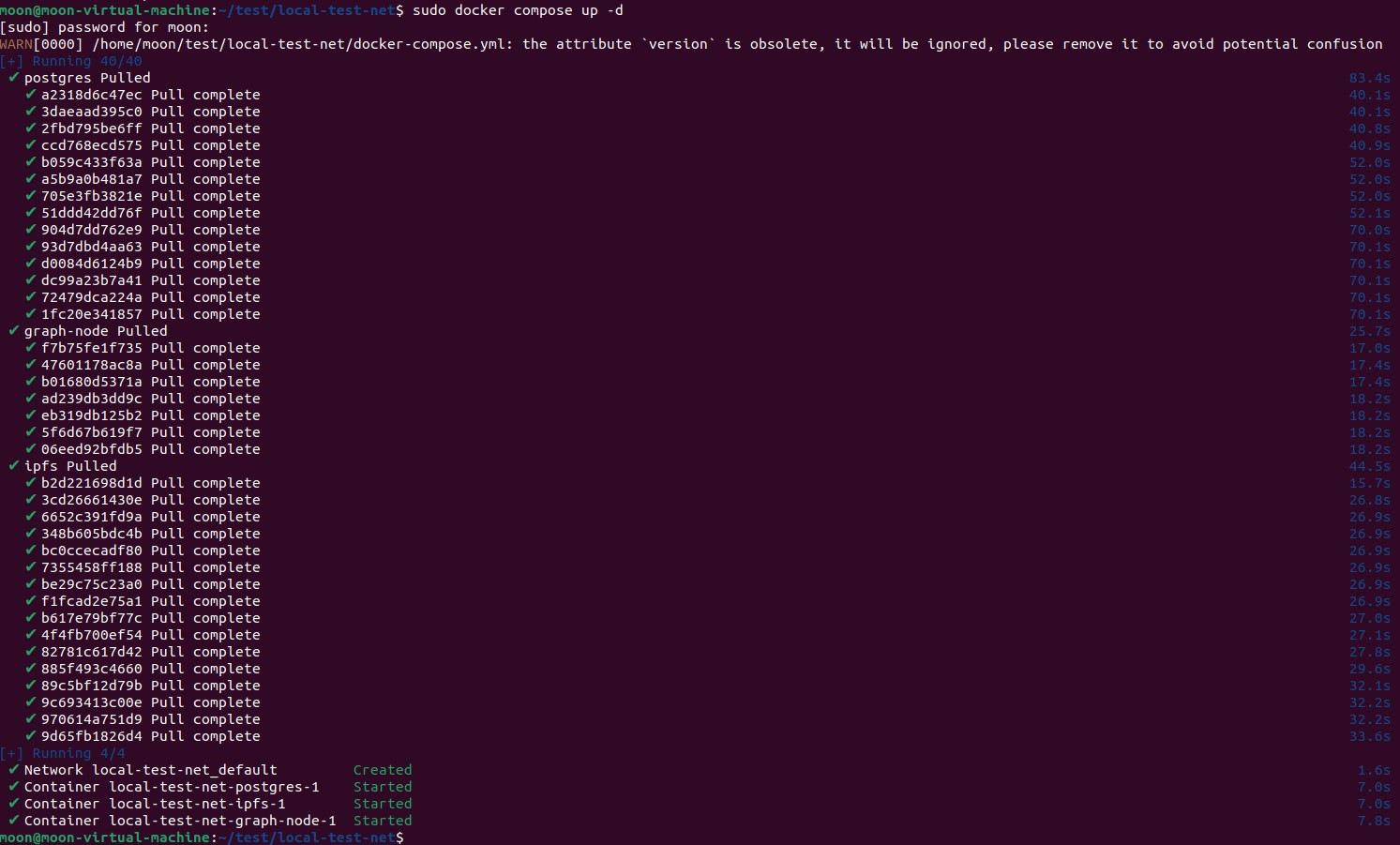
To deploy graph-node, you should install `graph-cli` and `graph-ts`.

You can see the `docker-compose.yml` in `local-test-net` directory.

You should replace the rpc port of ethereum node with the port of your local-eth-testnet which constructed in your local.

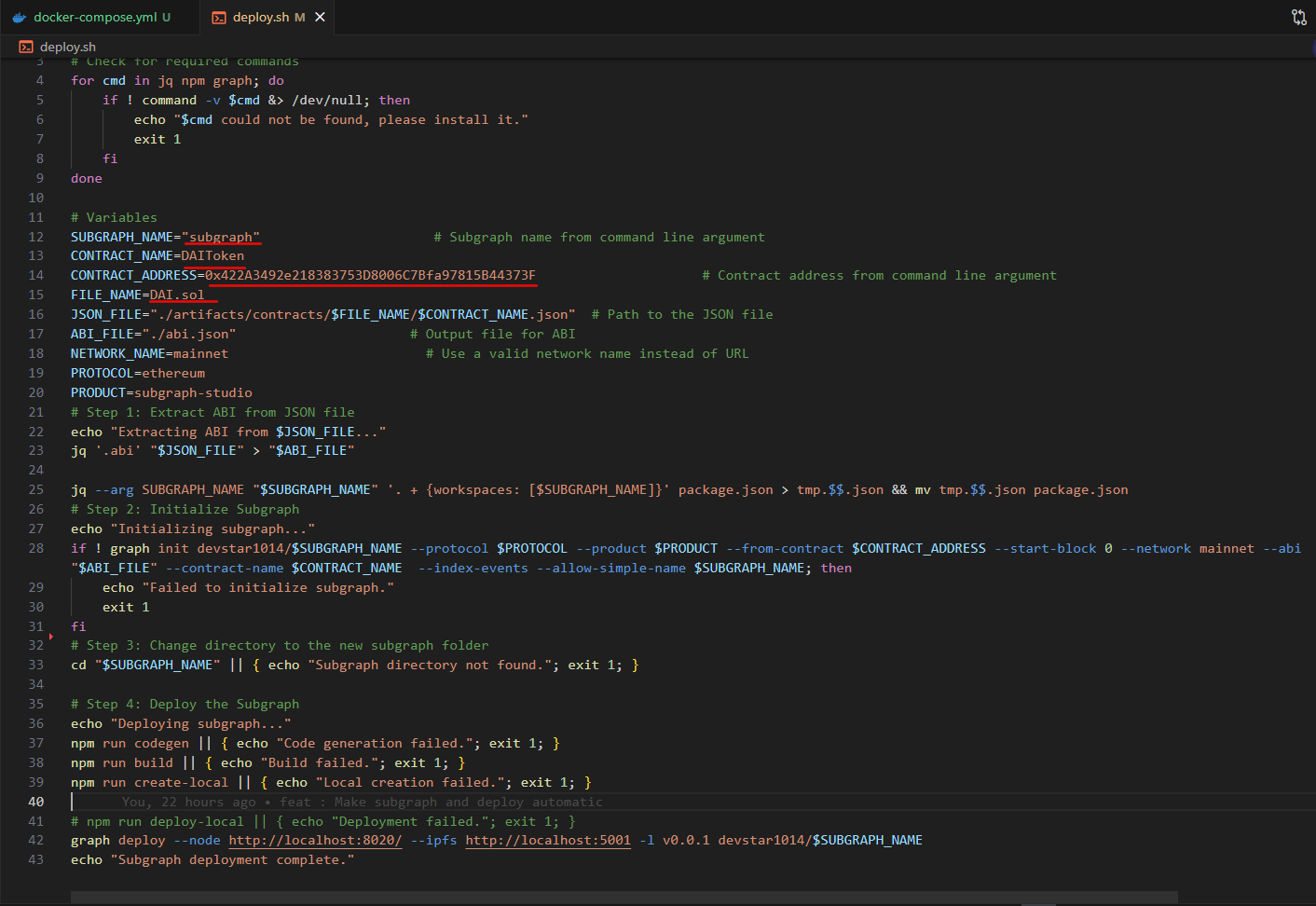
And run command:

`sudo docker-compose up -d`

Also you should install `jq`.

Run command:

`sudo apt update`

`sudo apt install jq -y`

You should change the value of the underlined fields.

FILE\_NAME is the name of the file that described contract.

CONTRACT\_ADDRESS : the address of the contact which deployed into localnet

And run `sudo chmod +x ./deploy.sh`.

Run: `./deploy.sh`