PROJECT 15
MAKING AN ETHEREUM CONTRACT WITH TRUFFLE

SPRING TRIMESTER 2022

By

NIKHIL PATEL

OVERVIEW : In this lab, We will learn how to install truffle prerequisites such as node, nvm, and npm in this lab. We will develop an Ethereum smart contract in this part. Ethereum's capacity to do these sorts of applications separates it from Bitcoin.

- Truffle is used to run smart contracts written in Solidity, and it may also be tested without utilizing actual ether. It is simple to operate a truffle via truffle init, truffle develop, and truffle start. Smart contracts are simple to install on a network.
- The truffle migrate command is in charge of migrating javascript, which allows you to deploy contracts on the network. In truffle, we can construct a smart contract (create contract proofexistence1), which will automatically produce the files needed for the smart contract.

PROCEDURE :

1) **Installing Node.js and npm on Ubuntu 16.04**

   Go to terminal and execute the following :

   sudo apt install -y curl

   When you get this error Could not get lock /var/lib/dpkg/lock-frontend - open
   To remove the file/folder execute:

   sudo rm /var/lib/dpkg/lock-frontend

   sudo apt install -y curl

   In the same terminal execute this command:

   curl -fsSL https://deb.nodesource.com/setup_16.x | sudo -E bash -

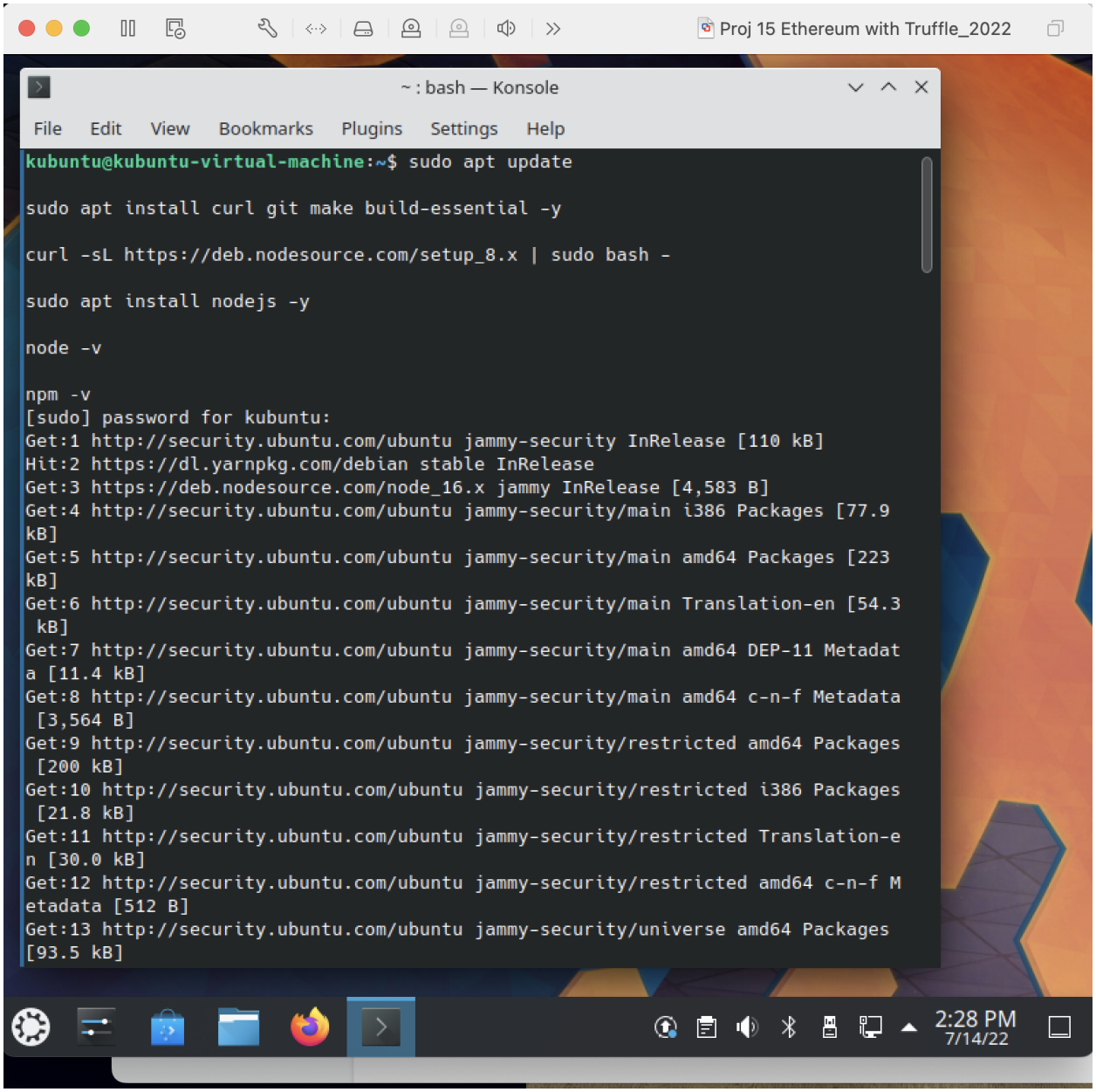
   sudo apt-get install -y nodejs

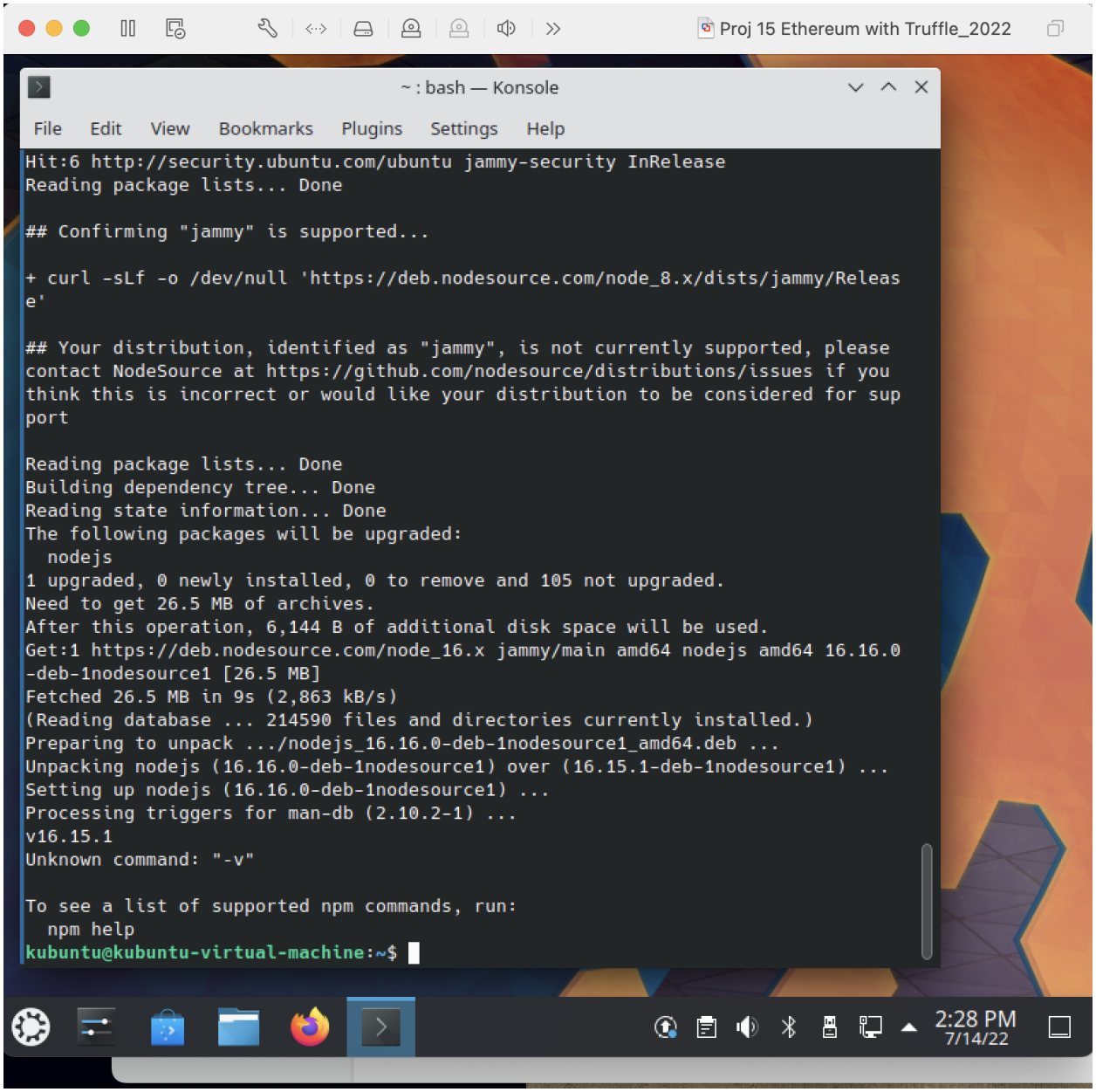
   Open a terminal, execute :

   node -v

   npm -v

   sudo npm install npm@latest -g

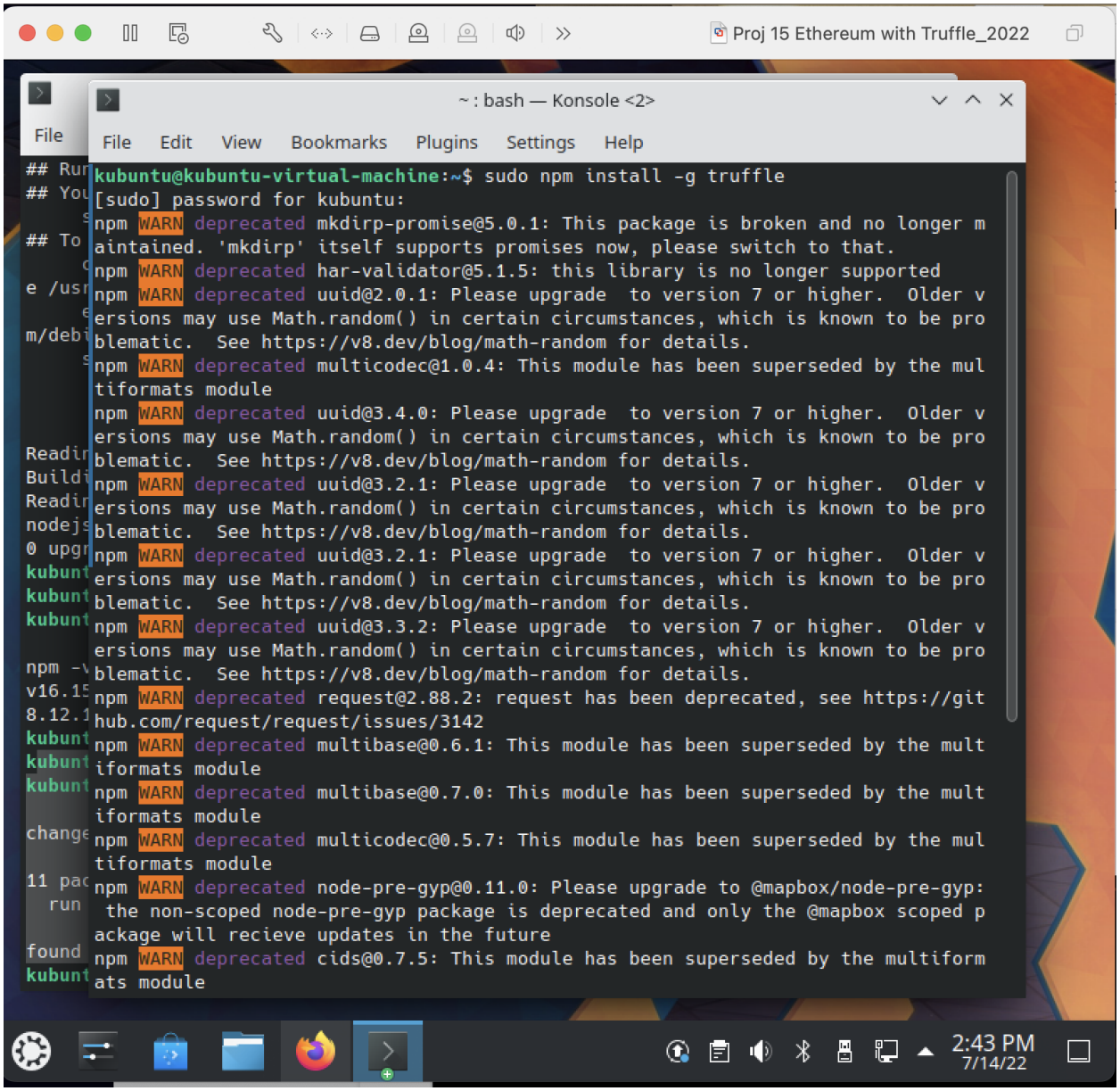

Proj 15 Ethereum with Truffle_2022
~ : bash — Konsole
File Edit View Bookmarks Plugins Settings Help
kubuntu@kubuntu-virtual-machine:~$ sudo apt update
sudo apt install curl git make build-essential -y
curl -sL https://deb.nodesource.com/setup_8.x | sudo bash -
sudo apt install nodejs -y
node -v
npm -v
[sudo] password for kubuntu:
Get:1 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Hit:2 https://dl.yarnpkg.com/debian stable InRelease
Get:3 https://deb.nodesource.com/node_16.x jammy InRelease [4,583 B]
Get:4 http://security.ubuntu.com/ubuntu jammy-security/main i386 Packages [77.9 kB]
Get:5 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [223 kB]
Get:6 http://security.ubuntu.com/ubuntu jammy-security/main Translation-en [54.3 kB]
Get:7 http://security.ubuntu.com/ubuntu jammy-security/main amd64 DEP-11 Metadata [11.4 kB]
Get:8 http://security.ubuntu.com/ubuntu jammy-security/main amd64 c-n-f Metadata [3,564 B]
Get:9 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 Packages [200 kB]
Get:10 http://security.ubuntu.com/ubuntu jammy-security/restricted i386 Packages [21.8 kB]
Get:11 http://security.ubuntu.com/ubuntu jammy-security/restricted Translation-en [30.0 kB]
Get:12 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 c-n-f Metadata [512 B]
Get:13 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 Packages [93.5 kB]
2:28 PM
7/14/22


Proj 15 Ethereum with Truffle_2022
~ : bash — Konsole
File Edit View Bookmarks Plugins Settings Help
Hit:6 http://security.ubuntu.com/ubuntu jammy-security InRelease
Reading package lists... Done
Confirming "jammy" is supported...
+ curl -sLf -o /dev/null 'https://deb.nodesource.com/node_8.x/dists/jammy/Releas
e'
Your distribution, identified as "jammy", is not currently supported, please
contact NodeSource at https://github.com/nodesource/distributions/issues if you
think this is incorrect or would like your distribution to be considered for sup
port
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages will be upgraded:
nodejs
1 upgraded, 0 newly installed, 0 to remove and 105 not upgraded.
Need to get 26.5 MB of archives.
After this operation, 6,144 B of additional disk space will be used.
Get:1 https://deb.nodesource.com/node_16.x jammy/main amd64 nodejs amd64 16.16.0
-deb-1nodesource1 [26.5 MB]
Fetched 26.5 MB in 9s (2,863 kB/s)
(Reading database ... 214590 files and directories currently installed.)
Preparing to unpack .../nodejs_16.16.0-deb-1nodesource1_amd64.deb ...
Unpacking nodejs (16.16.0-deb-1nodesource1) over (16.15.1-deb-1nodesource1) ...
Setting up nodejs (16.16.0-deb-1nodesource1) ...
Processing triggers for man-db (2.10.2-1) ...
v16.15.1
Unknown command: "-v"
To see a list of supported npm commands, run:
npm help
kubuntu@kubuntu-virtual-machine:~$
2:28 PM
7/14/22

2) Installing Truffle

Open a New Terminal, execute :

sudo npm install -g truffle



3) Set Up a Project

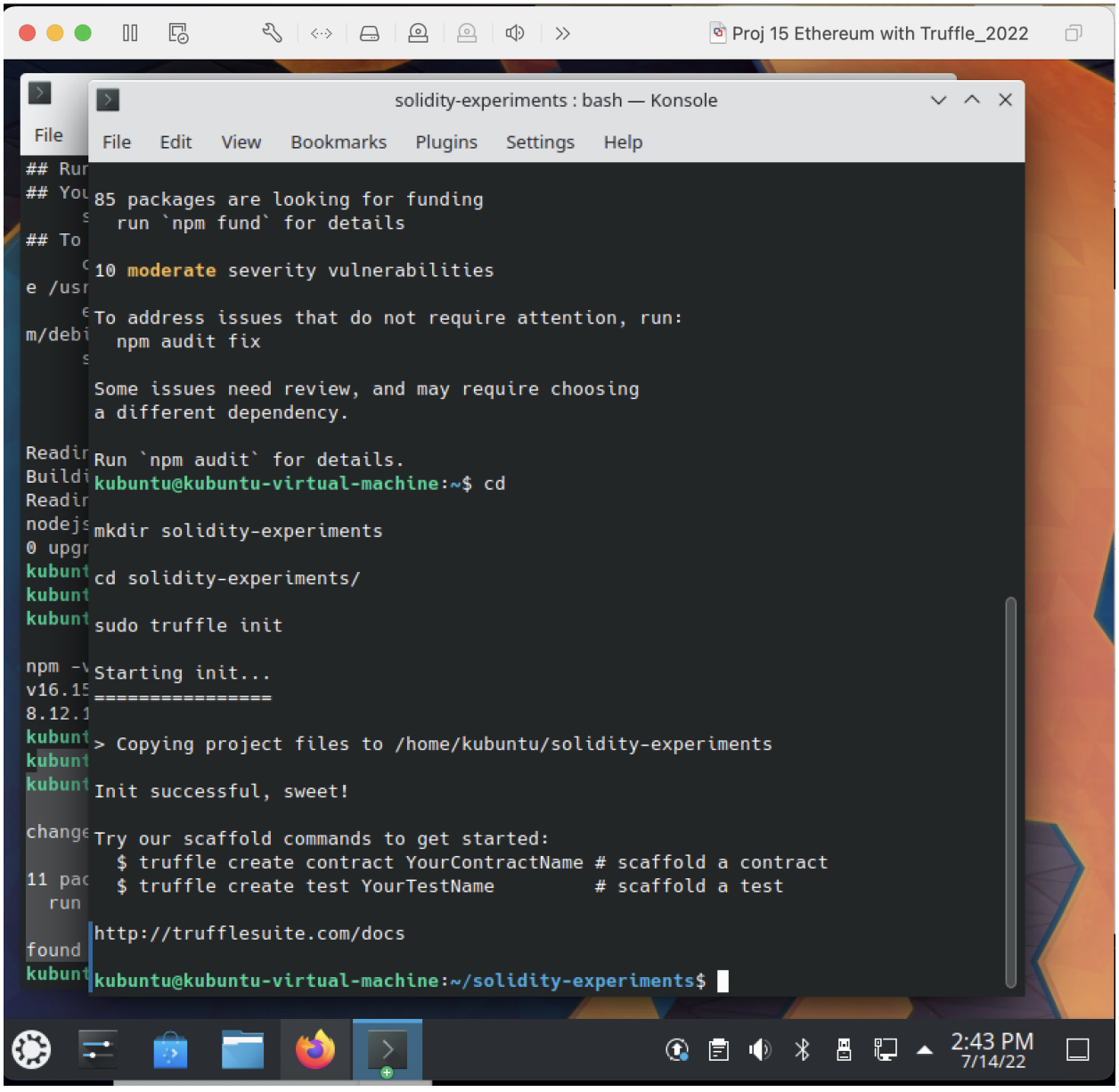
In the terminal execute the following command:

cd

mkdir solidity-experiments

cd solidity-experiments/

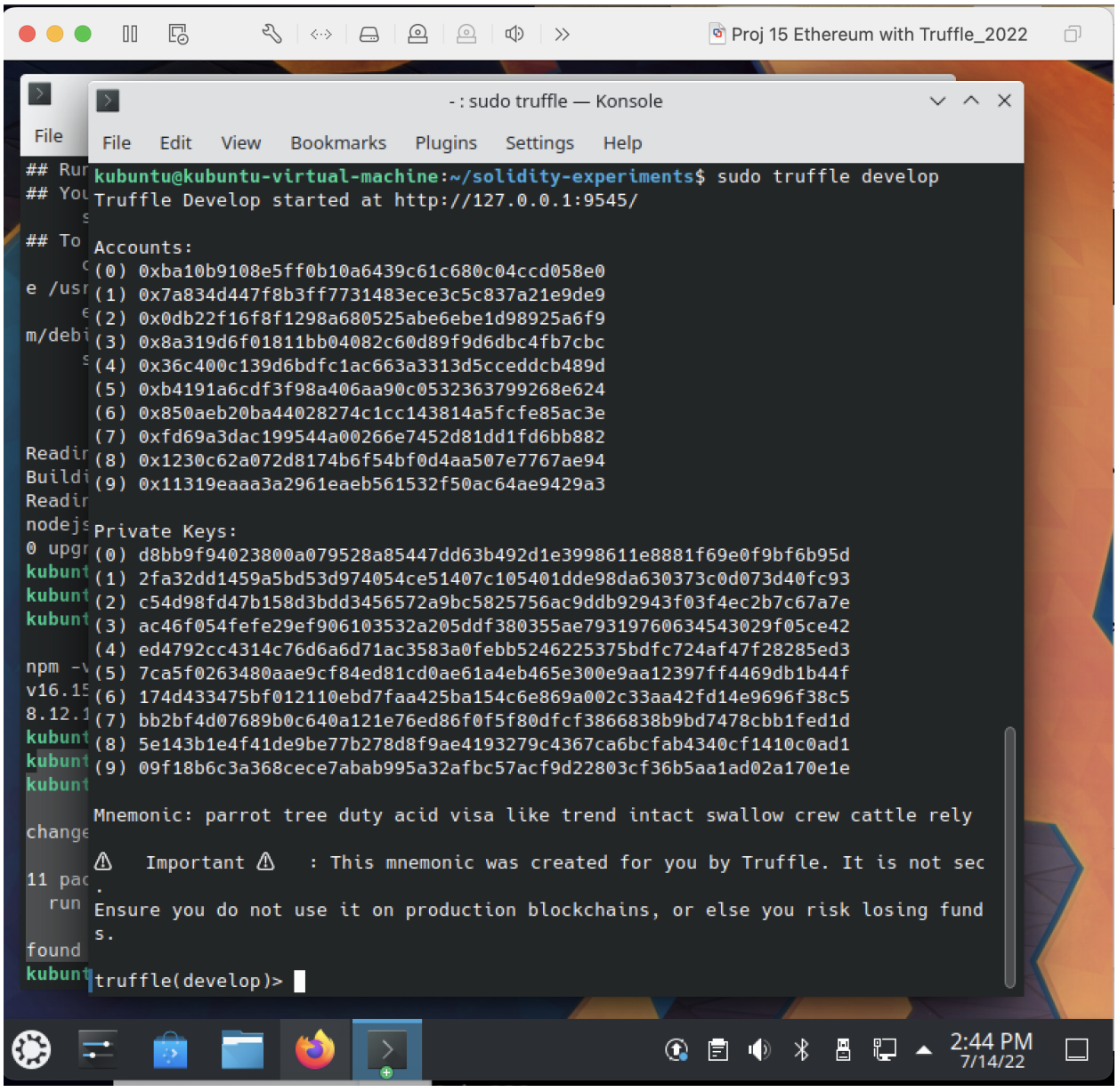
sudo truffle init



4) Starting the Development Testchain

In the terminal execute the following command:

sudo truffle develop

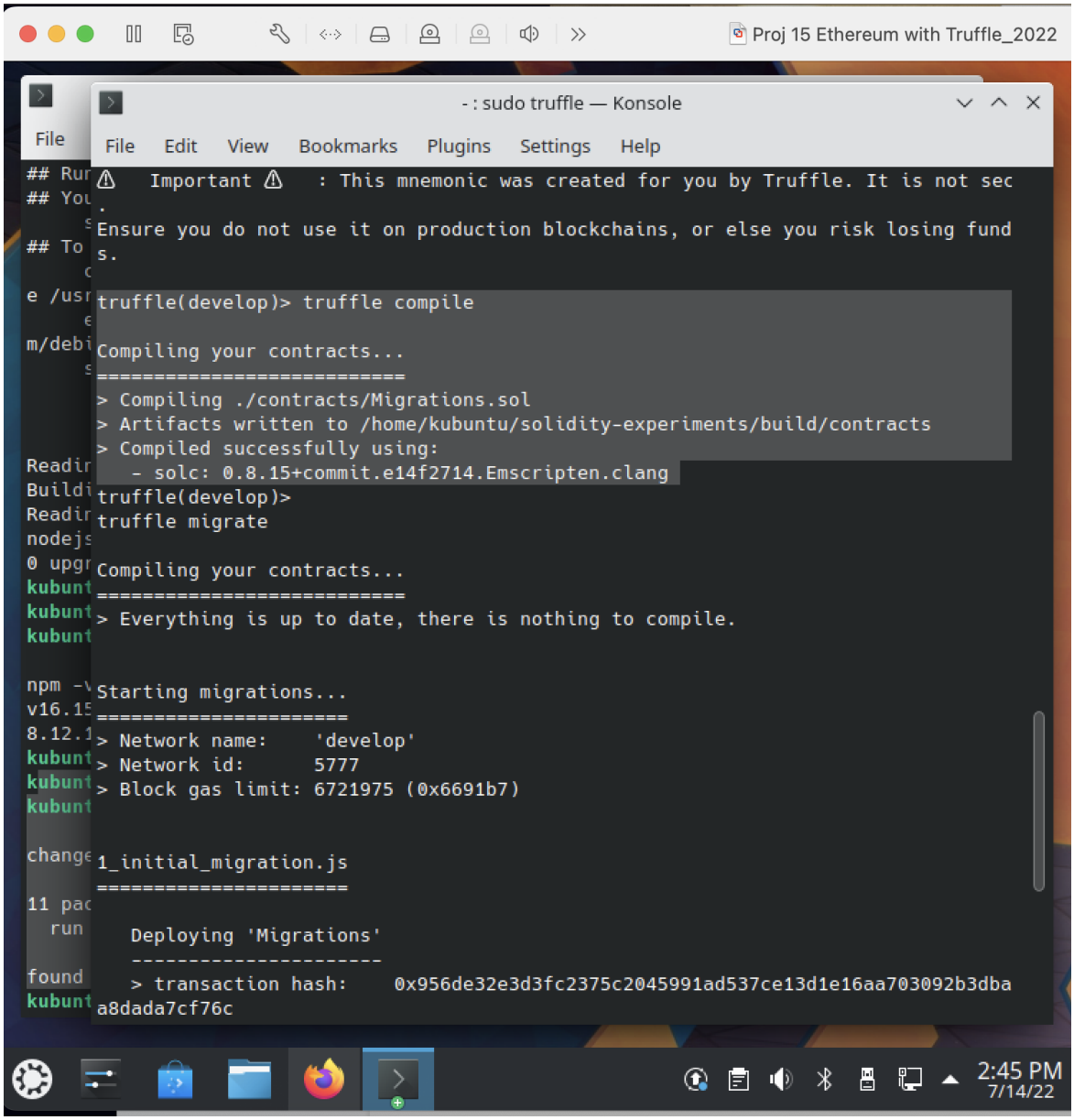


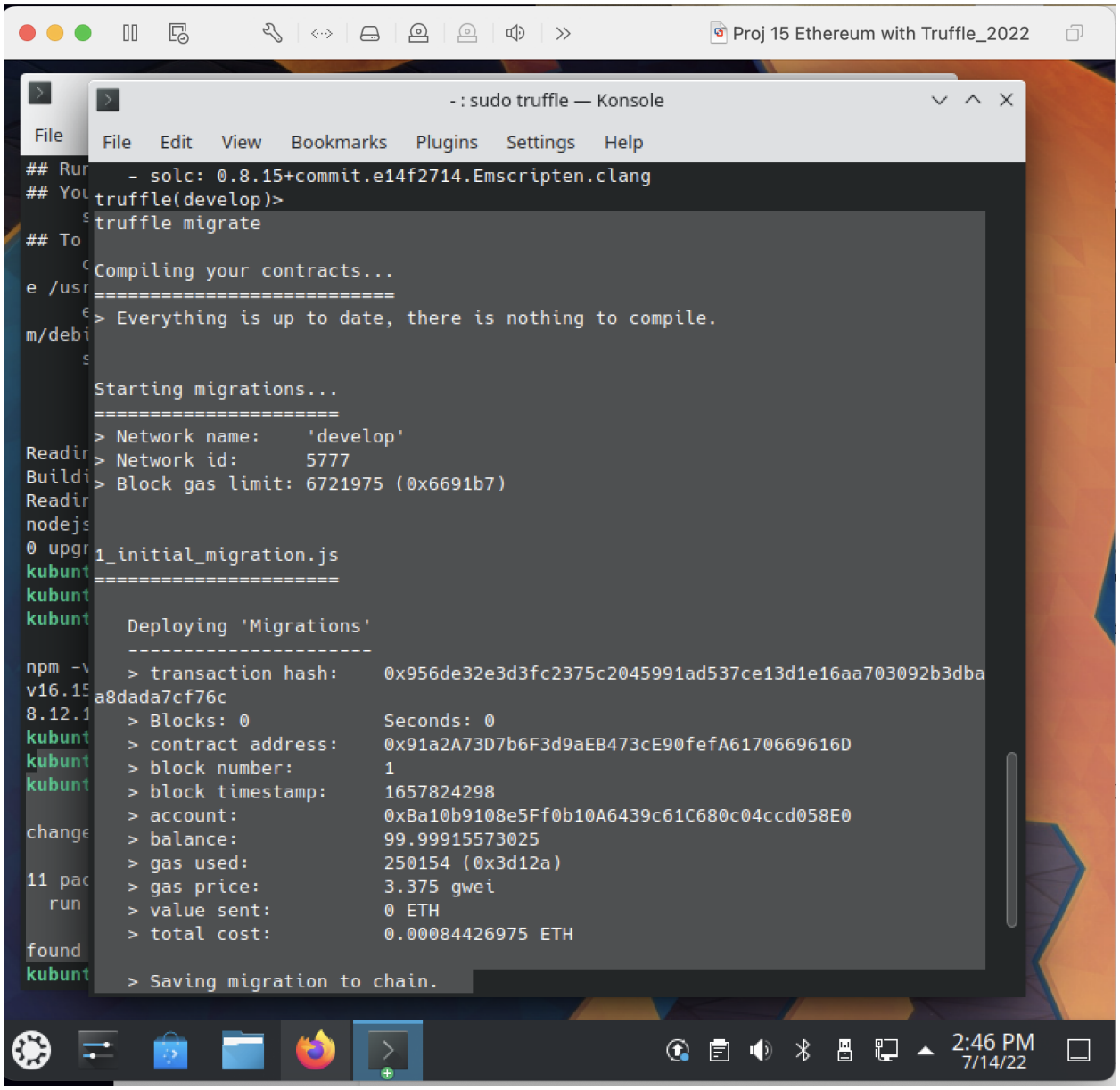
5) Compiling and Deploying Sample Contracts

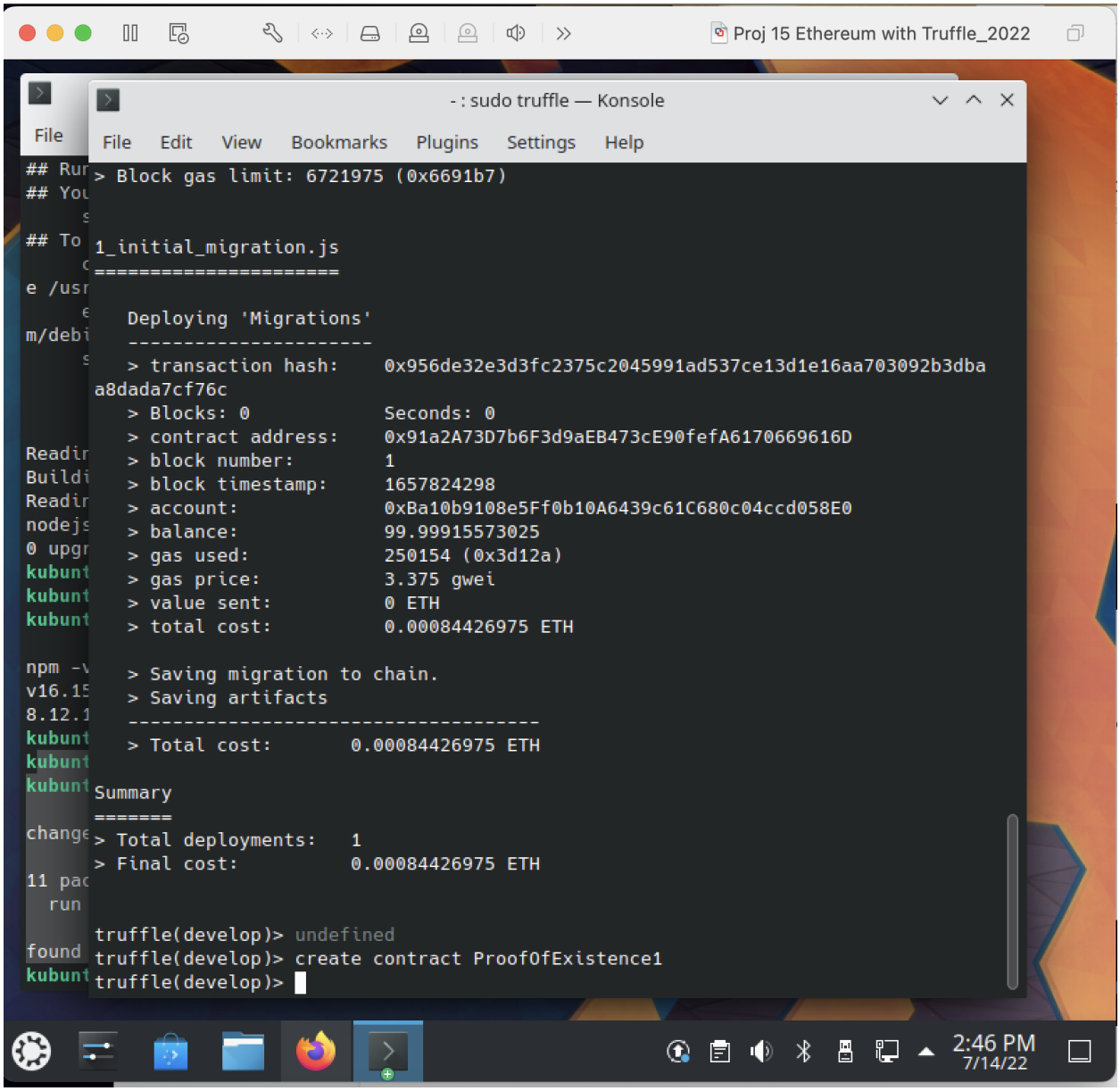
In the terminal, at the **truffle(develop)>**prompt execute :

truffle compile

truffle migrate




Proj 15 Ethereum with Truffle_2022
- : sudo truffle — Konsole
File Edit View Bookmarks Plugins Settings Help
- solc: 0.8.15+commit.e14f2714.Emscripten.clang
truffle(develop)>
truffle migrate
Compiling your contracts...
> Everything is up to date, there is nothing to compile.
Starting migrations...
> Network name: 'develop'
> Network id: 5777
> Block gas limit: 6721975 (0x6691b7)
1_initial_migration.js
Deploying 'Migrations'
> transaction hash: 0x956de32e3d3fc2375c2045991ad537ce13d1e16aa703092b3dba
a8dada7cf76c
> Blocks: 0 Seconds: 0
> contract address: 0x91a2A73D7b6F3d9aEB473cE90fefA6170669616D
> block number: 1
> block timestamp: 1657824298
> account: 0xBa10b9108e5Ff0b10A6439c61C680c04ccd058E0
> balance: 99.99915573025
> gas used: 250154 (0x3d12a)
> gas price: 3.375 gwei
> value sent: 0 ETH
> total cost: 0.00084426975 ETH
> Saving migration to chain.
2:46 PM
7/14/22



6) Creating a Smart Contract

In the terminal, at the **truffle(develop)>**prompt execute :

create contract ProofOfExistence1

7) Adding Solidity Code

Open a New Terminal, execute :

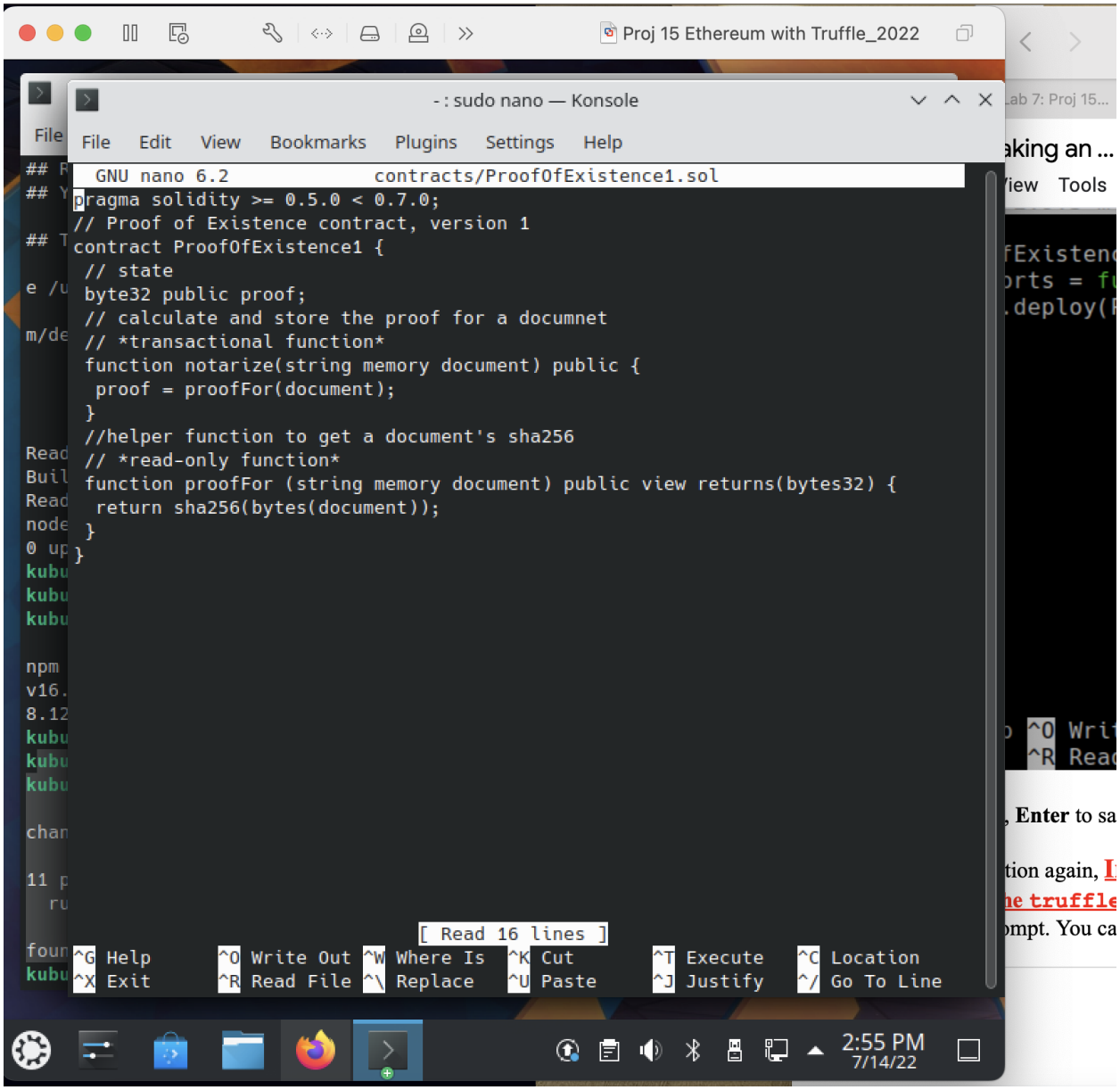
cd

cd solidity-experiments

sudo nano contracts/ProofOfExistence1.sol

Remove the code you see in the above step and replace it with the code below and make you type this code don't copy paste if you run this code with error then you need to start from the beginning:

```
pragma solidity >= 0.5.0 < 0.7.0;
// Proof of Existence contract, version 1
contract ProofOfExistence1 {
  // state
  bytes32 public proof;
  // calculate and store the proof for a document
  // *transactional function*
  function notarize(string memory document) public {
    proof = proofFor(document);
  }
  // helper function to get a document's sha256
  // *read-only function*
  function proofFor(string memory document) public view returns (bytes32)
{
    return sha256(bytes(document));
  }
}
```



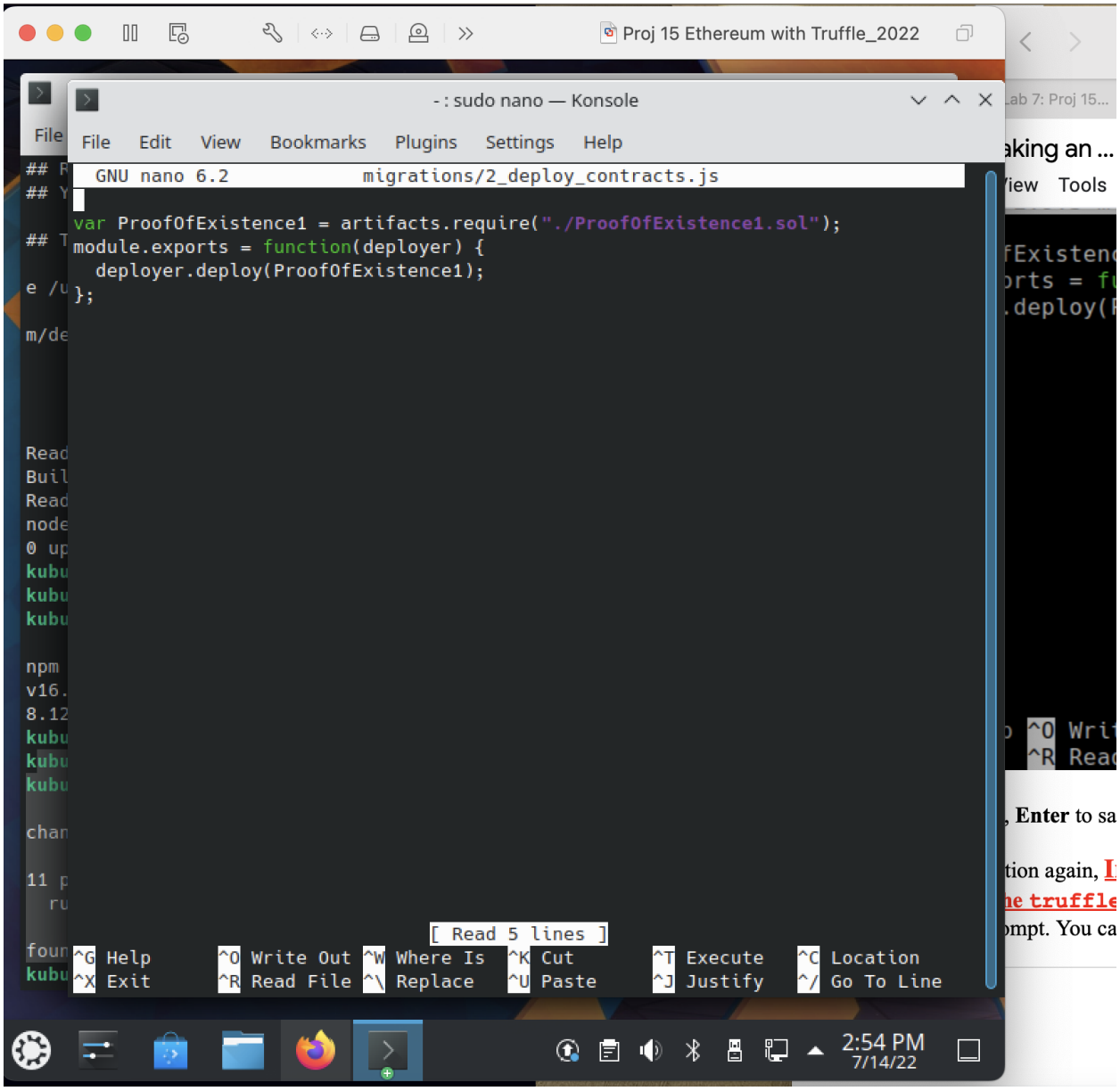
8) Deploying the Contract

In the terminal execute the following command:

sudo nano migrations/2_deploy_contracts.js

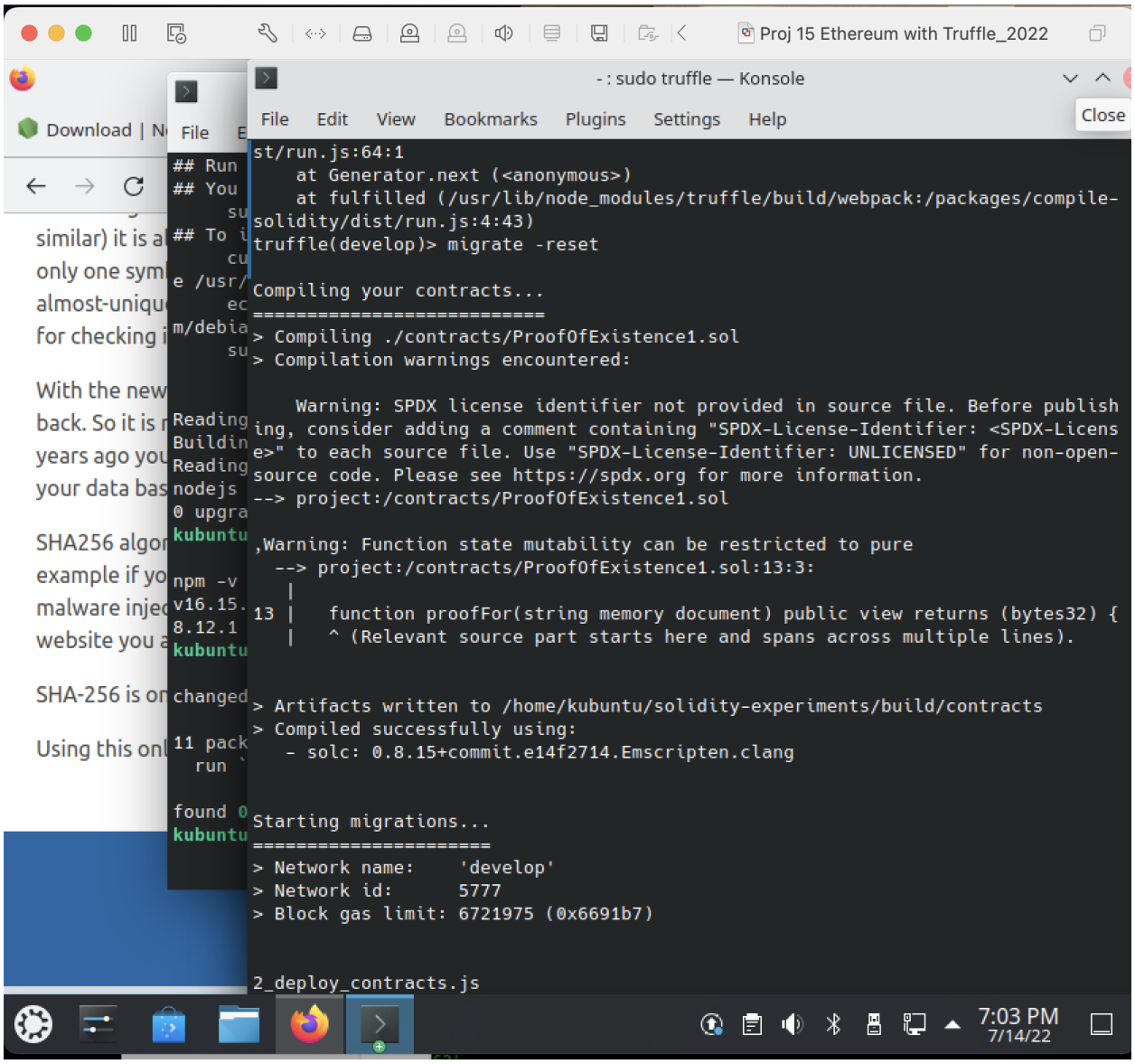
Paste the below code in the file :

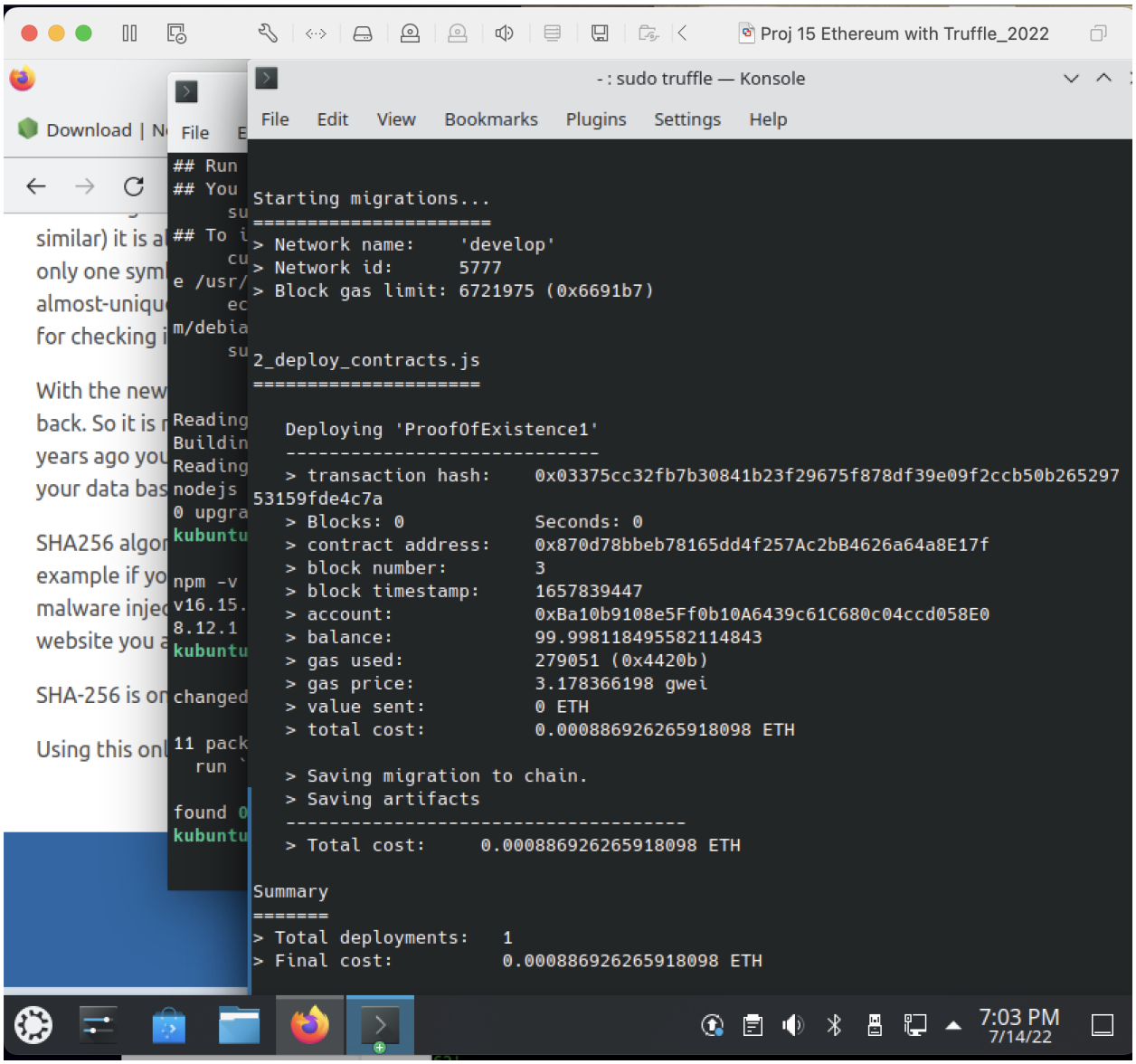
```
var ProofOfExistence1 = artifacts.require("./ProofOfExistence1.sol");
module.exports = function(deployer) {
 deployer.deploy(ProofOfExistence1);
};
```



Go to second terminal, with prompt ***truffle develop*** >, execute :

migrate –reset



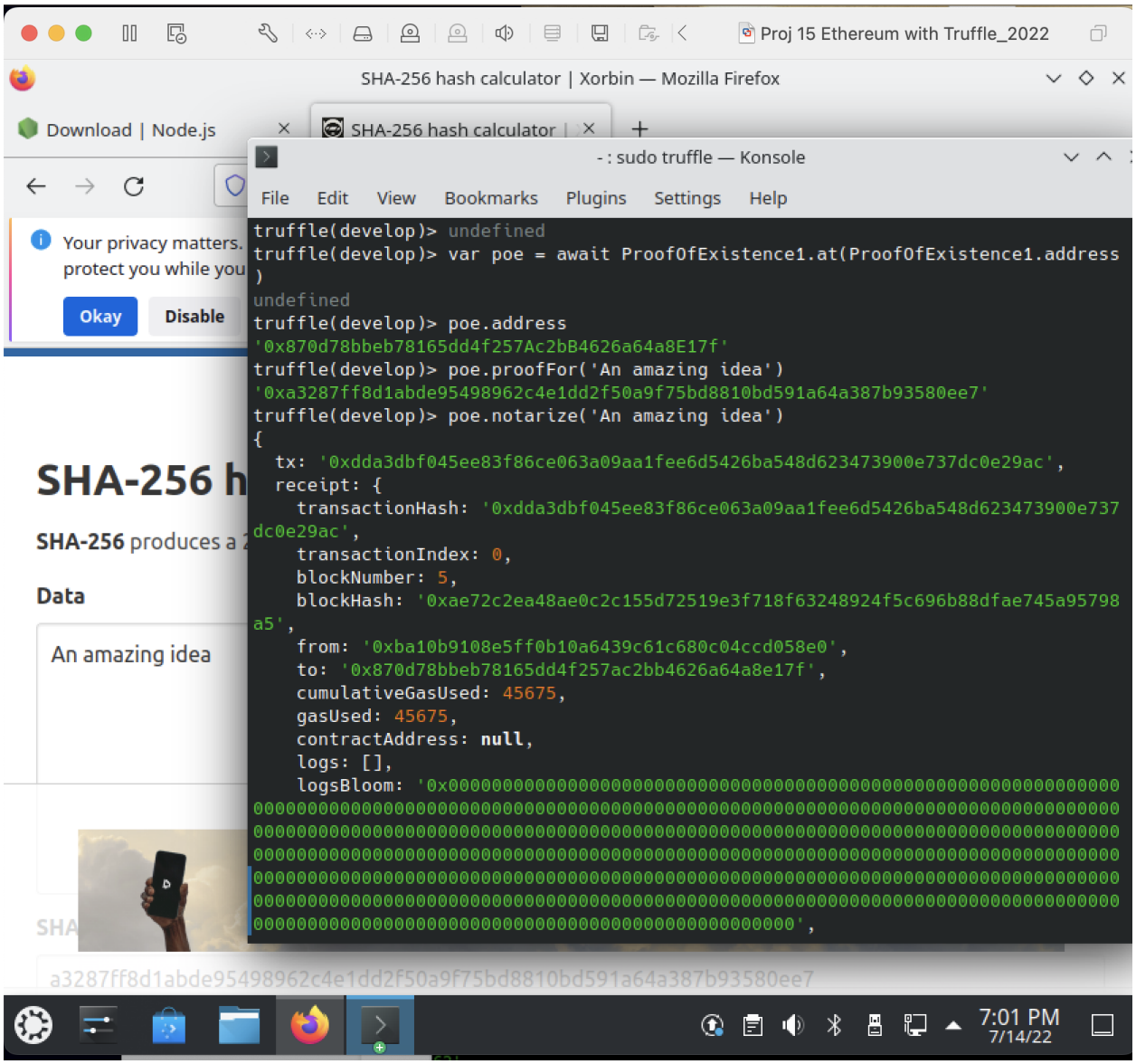


9) Finding the Smart Contract's Address

Same in the second terminal at ***truffle develop*** **>** prompt, execute :

var poe = await ProofOfExistence1.at(ProofOfExistence1.address)

poe.address



10) Calculating a Hash

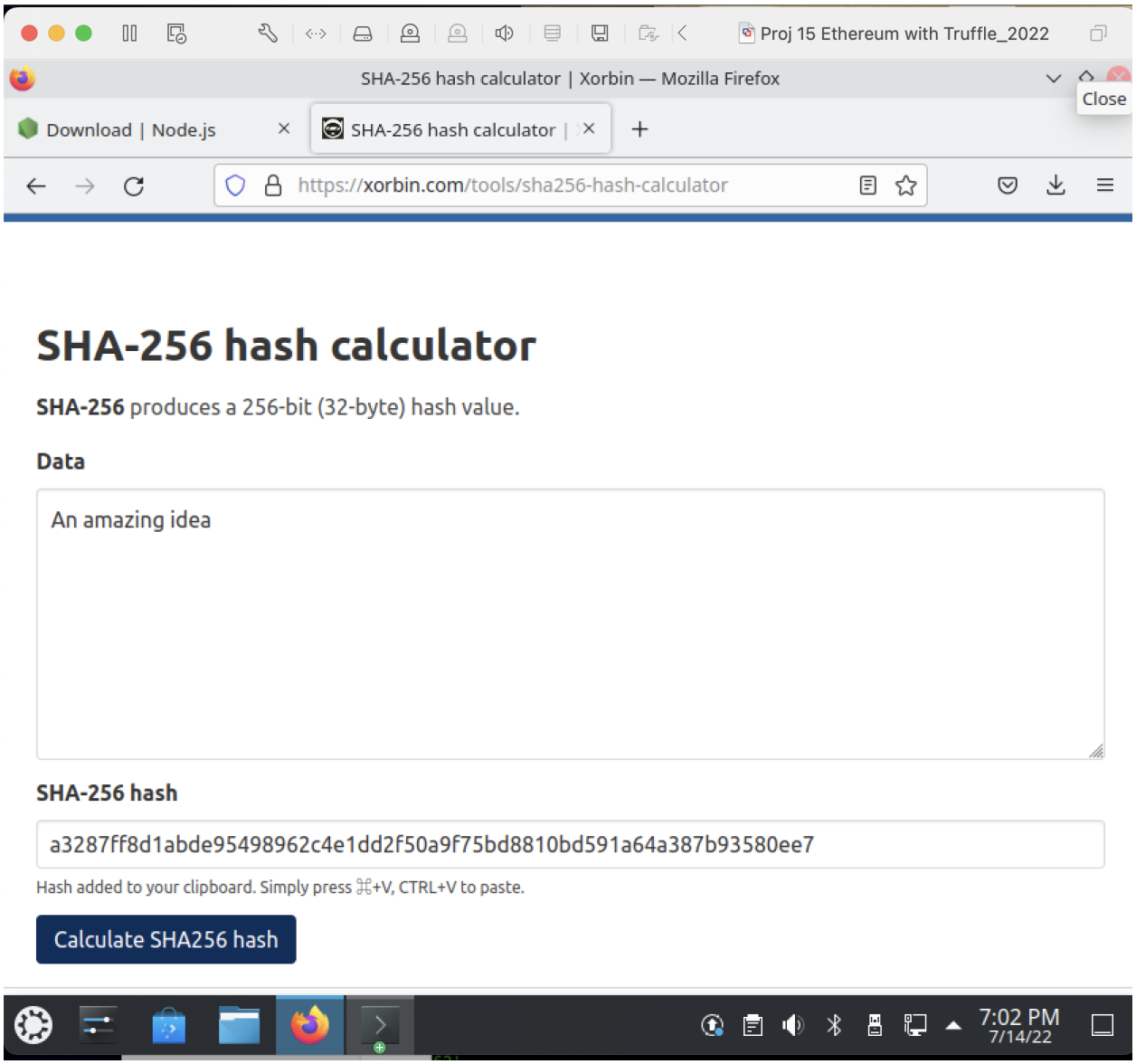
Go to the web browser within your vm and go to this link :

https://xorbin.com/tools/sha256-hash-calculator

The SHA256 hash of this message is shown below.

In the original terminal, at the ***truffle develop>*** prompt, execute :

poe.proofFor('An amazing idea')

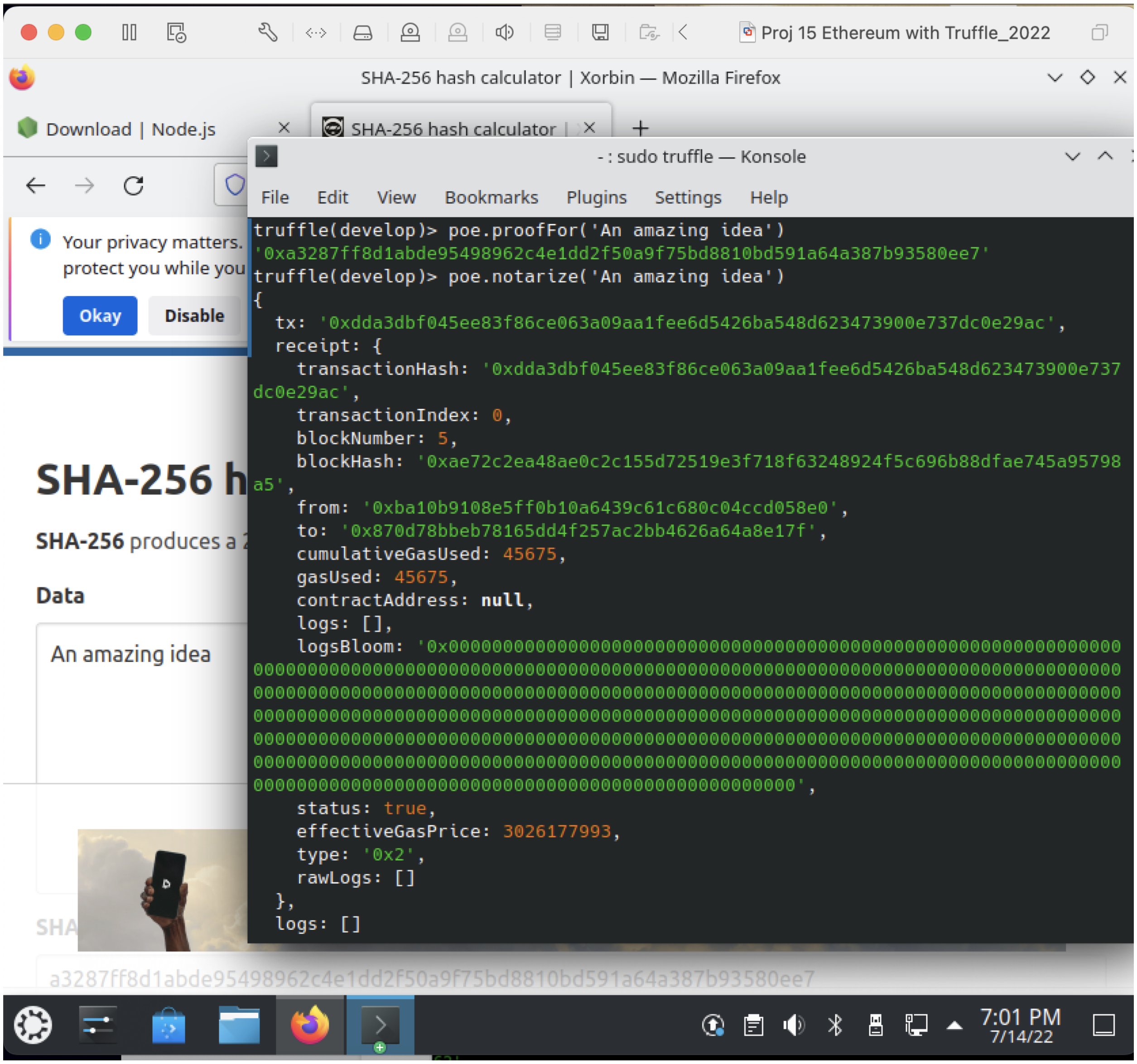


11) Registering a Document

In the original terminal, at the ***truffle develop>*** prompt, execute :

poe.notarize('An amazing idea')

poe.proof()

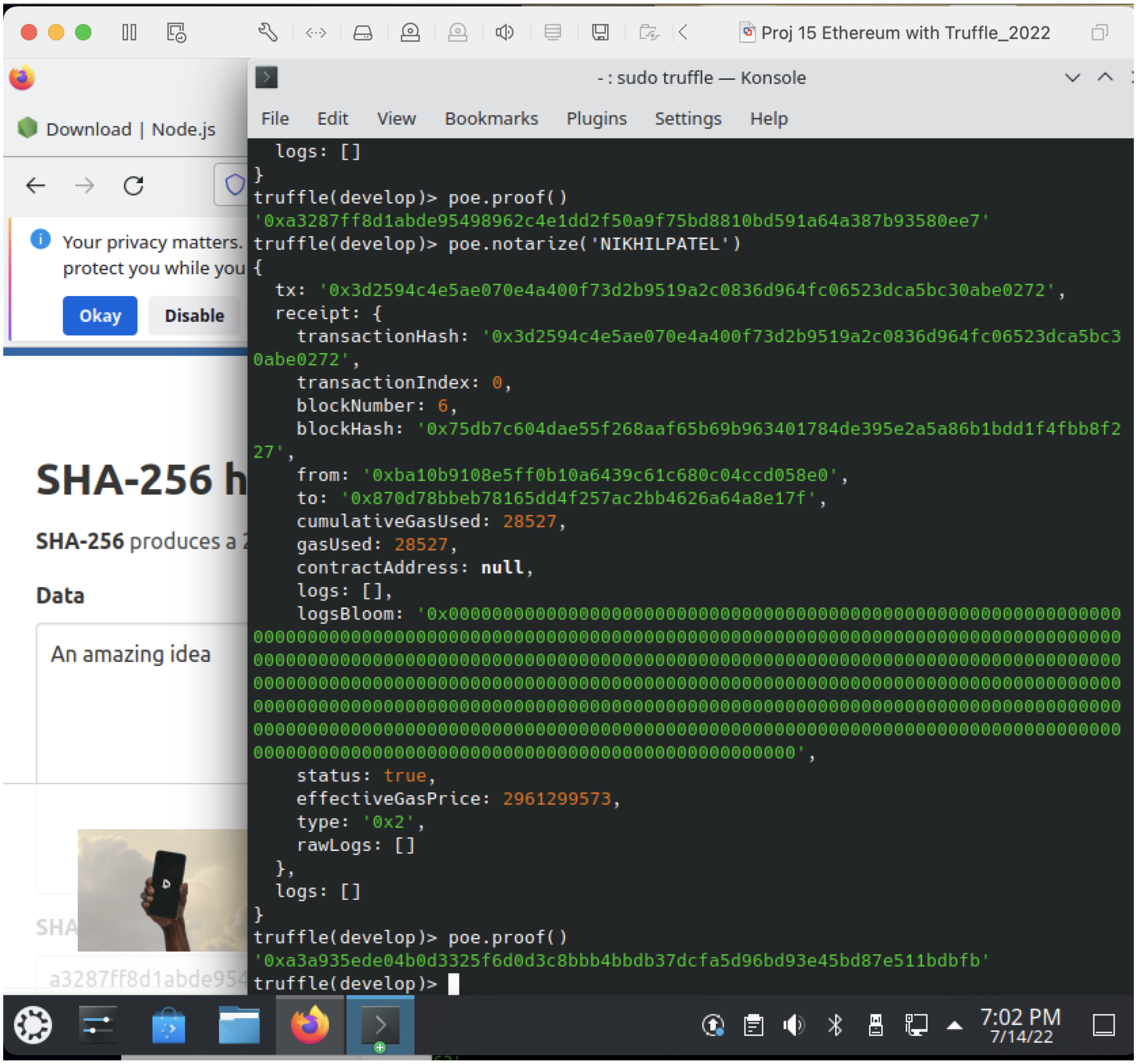


12) Registering Your Name

In the original terminal, at the ***truffle develop>*** prompt, execute{replace YOUNAME with your name}:

poe.notarize('YOURNAME')

poe.proof()



<u>CONCLUSION</u> : We can conclude that first we install and set up truffle and installed dependencies then we initialize truffle and we develop it by truffle develop command then we compile it also we migrate it and we created Proofofexistence1 smart contract in which we add notarized transaction code. Then we reset it and deploy again. In the end we calculate a hash value of any string then we verify it on an online site and hashes completely match with it so we can say that we successfully made a smart contract in truffle.