Testing Blockchain Transactions

FinTech

Lesson 19.3

Class Objectives

By the end of this lesson, you will be able to:

- Use a blockchain explorer to visualize each part of a transaction.
- Develop code by using Python and Web3.py to connect to a local Ganache blockchain.
- Use Web3.py in conjunction with Ganache to test your transaction
- Formulate code to sign and send a transaction by using Web3.py.
- Use Streamlit and Web3.py together to build an application that communicates with the blockchain.
- Test a blockchain web application by using Ganache.


WELCOME

Let’s begin by recapping the previous unit and lessons.

What is the difference between a hot wallet and a cold wallet?

Hot Wallet

- Software or hardware device
- Often or always connected to the internet
- Easier to access account information
- Less secure

Cold Wallet

- Hardware device
- Rarely connected to the internet
- More difficult to access account information
- Extremely secure

What are public and private keys?

Public and Private Keys

Public key

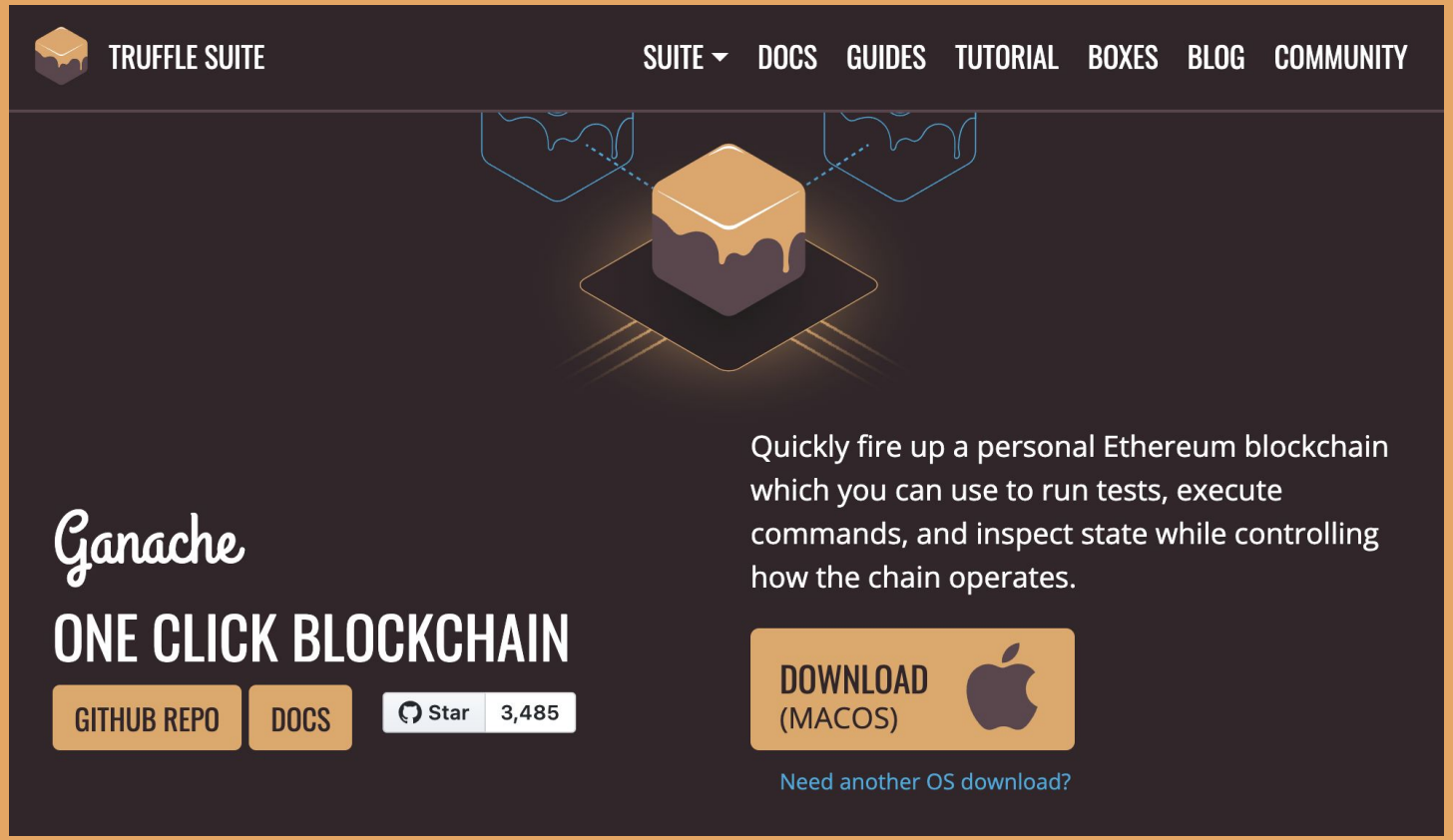
- Is used to encrypt plaintext to convert it to cipher text.
- Known by all users of a blockchain.
- Can be used by all members of the blockchain community to send transactions to a particular user.

Private key

- Is used by the receiver to decrypt the cipher text in order to read messages or send ether.
- Known only by the account owner.
- Is needed to send ether to other accounts.

Introduction to Ganache

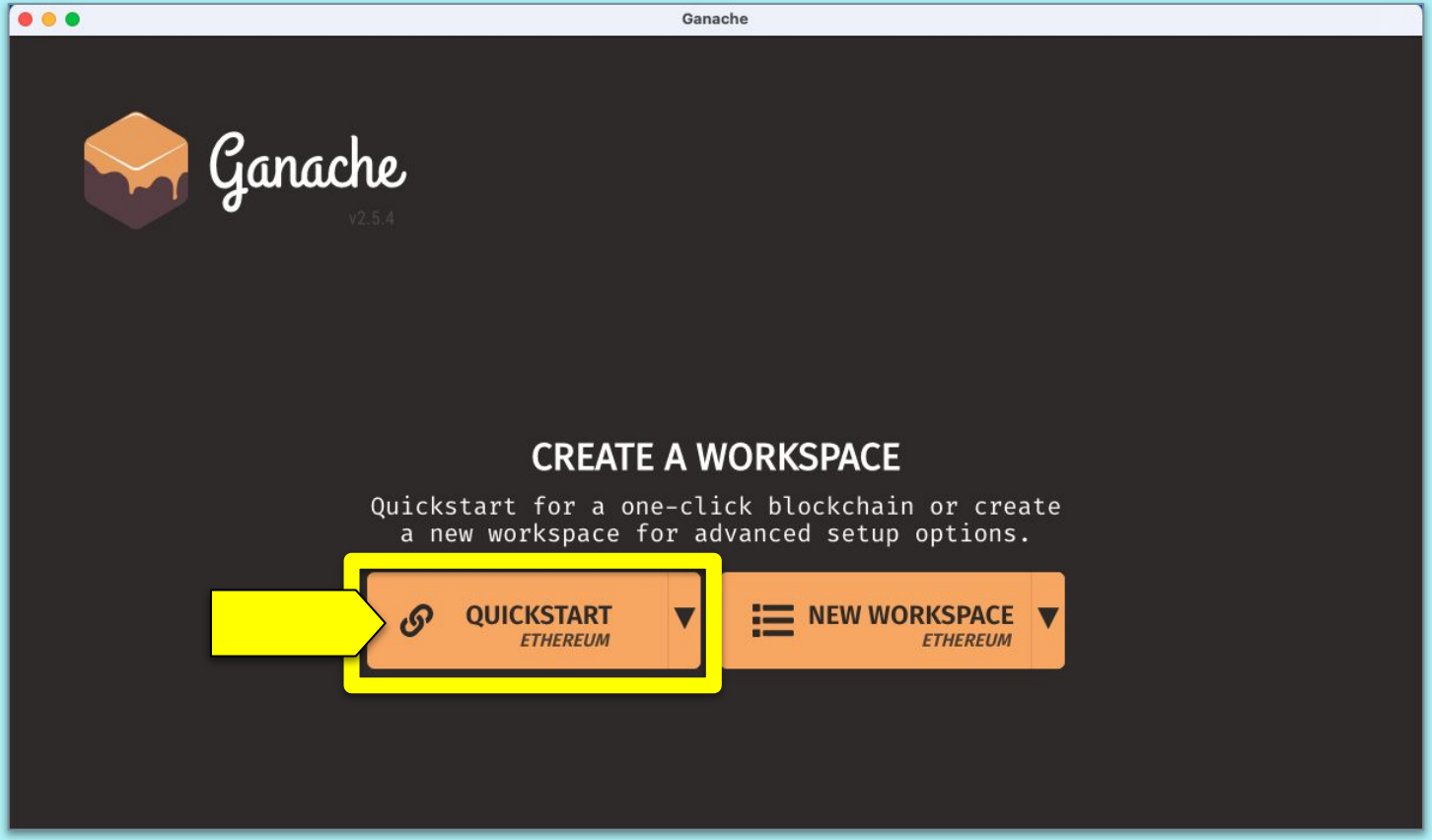
Confirm that you have downloaded and installed Ganache.



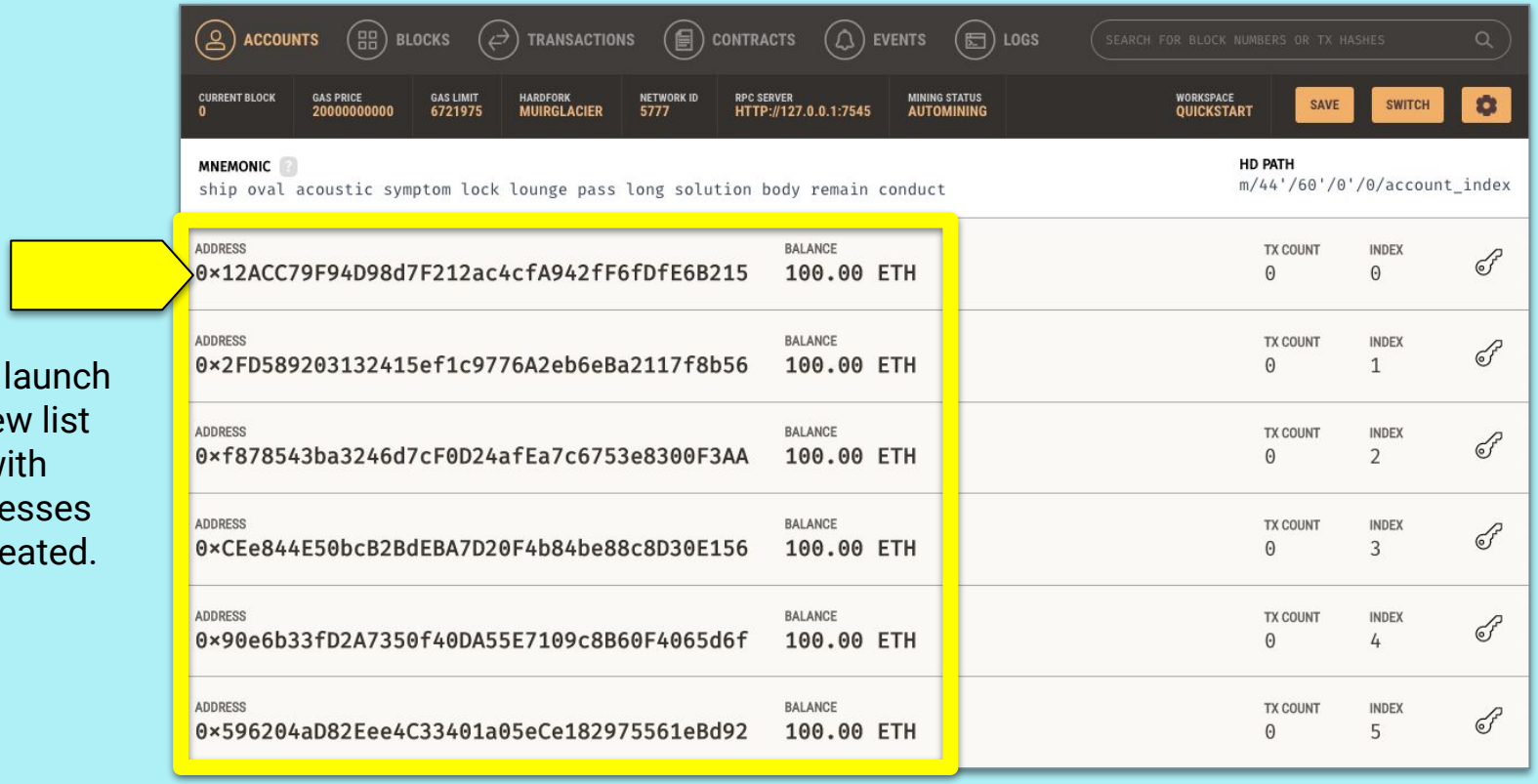
Ganache

- Ganache is a personal local blockchain with accounts that are preloaded with ether.
- The ether in these accounts has no real value on the Ethereum blockchain, as this blockchain is local only.
- Ganache is a useful tool for testing transactions and smart contracts.

Launch Ganache and select **Quickstart Ethereum**.

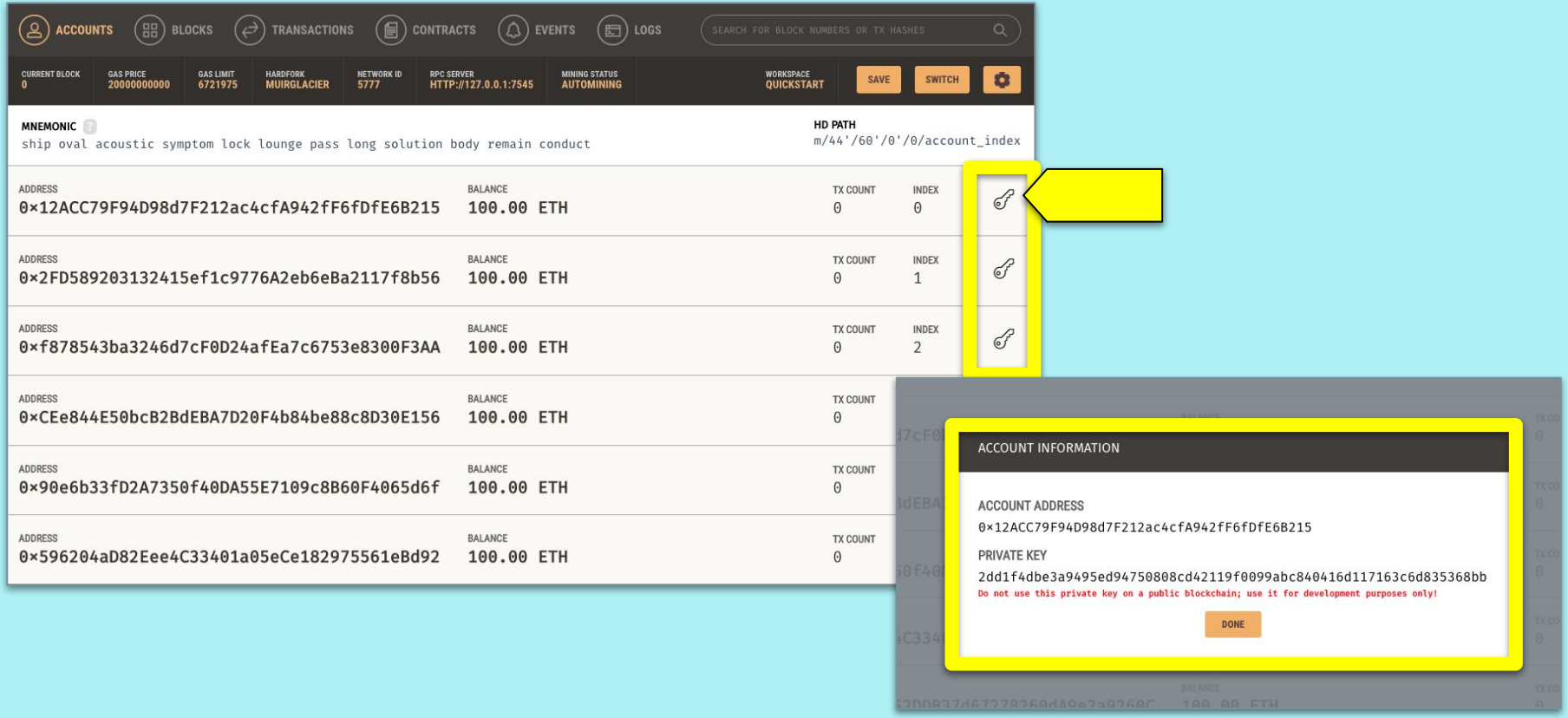


The result is a list of 10 different account addresses with a balance of 100 ETH in each.

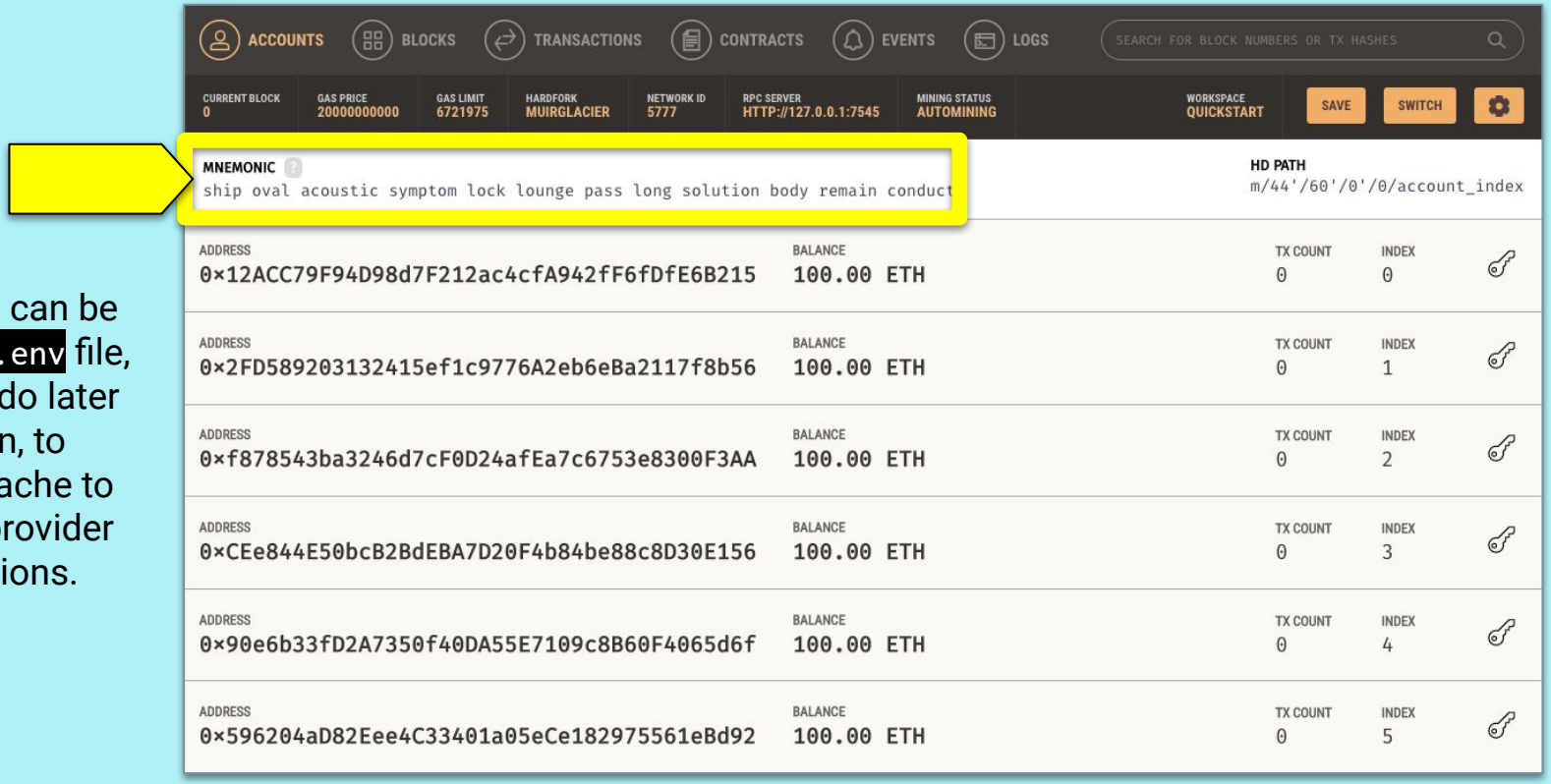


Each time we launch Ganache, a new list of accounts with different addresses and keys is created.

Clicking an account's key symbol opens a window that reveals the test account's private key as well as the address.



The mnemonic seed phrase is in the top left part of the interface.



This phrase can be added to a `.env` file, which we'll do later in the lesson, to enable Ganache to run as the provider for transactions.

Time to Code

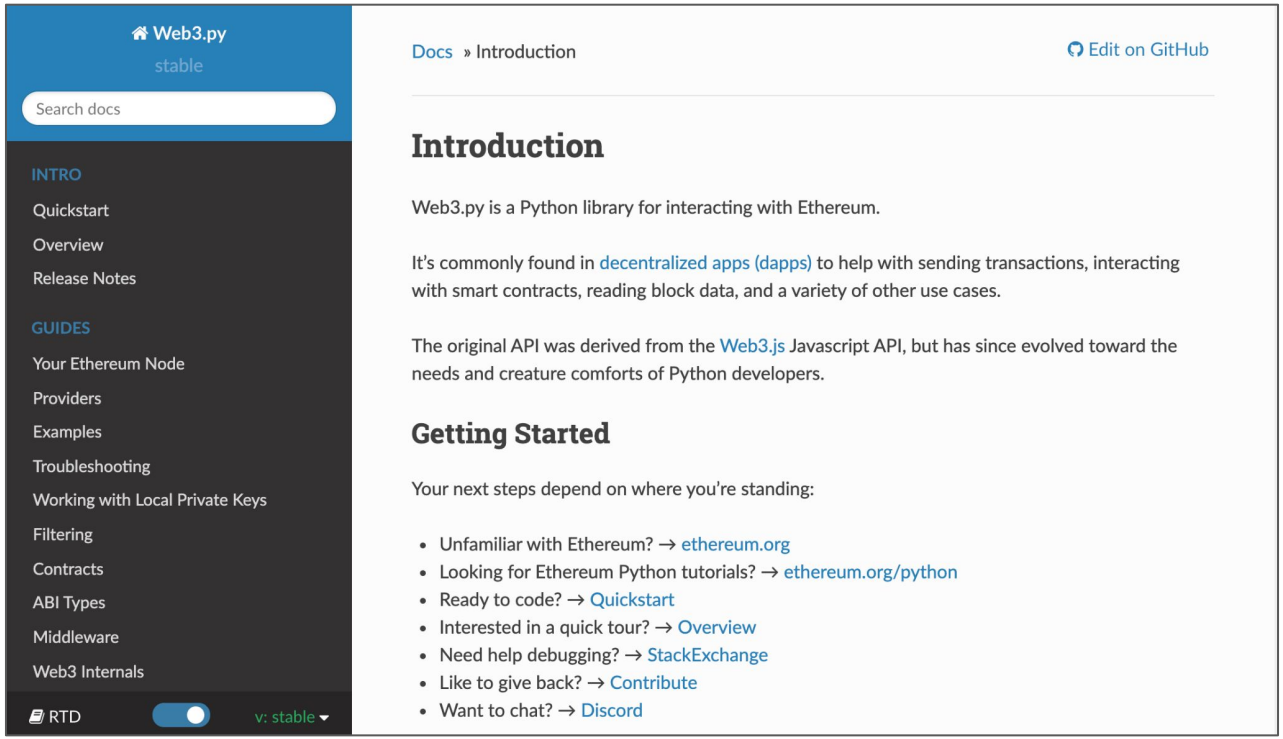
Using Streamlit with Web3.py

Suggested Time:

5 minutes

Using Streamlit with Web3.py

Web3.py will handle the transactions in the application you are building today.



Using Streamlit with Web3.py

In most decentralized applications that are built on the blockchain, Web3 is used to interact with smart contracts and read the block data. You will use it to send transactions.

The `web3. eth. Contract` objects can help developers interact with smart contracts on the Ethereum blockchain.

When you create a new contract object, you give it the JSON interface of the respective smart contract, and web3 will auto-convert all calls into low-level ABI calls over the remote procedure call (RPC).

Today we will connect web3 to Streamlit to enable our applications to interact with local blockchains.

Questions?

Time to Code

Ethereum and Streamlit

Suggested Time:

30 minutes

Ethereum and Streamlit

The focus of this activity is twofold:

01 To automate the Ethereum account functionality that we have learned up to now with the use of Python functions.

02 To integrate those Python functions with a Streamlit web application

Because Streamlit is involved, the activity will involve Python files rather than Jupyter notebooks.

Visual Studio Code will be the IDE used for this and the following activities.

Ethereum and Streamlit

In the `ethereum.py` file, you will create a Python function that does the following:

01 Accesses the `MNEMONIC` variable from the `.env` file.

02 Uses the `mnemonic` variable to create an HD `wallet`.

03 Uses the `wallet` to generate a public/private key pair.

04 Uses the `private` key to create an Ethereum `account`.

05 Returns the `account` from the function.

Activity: Automating Ethereum

In this activity, you will create a Streamlit web application that's similar to the one that you'll be asked to create in the homework assignment.

Suggested Time:

35 minutes

Time's Up! Let's Review.

Questions?


Break

Activity: Cats Mini-Project

In this activity, you will add functions to automate the process of accessing the balance from the Ganache blockchain, as well as sending a signed transaction.

You will then incorporate these functions into the Streamlit web application.

Suggested Time:

45 minutes

Time's Up! Let's Review.

Questions?

Structured Review

Structured Review

01

Are any activities that you want to review?

02

Let’s revisit key activities that are relevant for the homework assignment.

03

You can start the homework assignment, which is a combination of sections 1 and 3.


The End