SMART CONTRACT TUTORIAL (Part 1)

This tutorial will guide you through an introduction of writing and deploying a smart contract on a blockchain. You will learn the basics of smart contracts, the Hello World equivalent of blockchain development.
The technologies and software that will be used for this tutorial are:

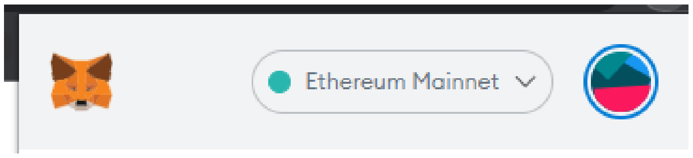
- **Alchemy (https://www.alchemy.com/)**
- **Metamask (https://metamask.io/)**
- **Visual Studio Code (https://code.visualstudio.com/)**
- **Unix System Terminal (This tutorial is performed on Ubuntu 20.04)**
- **Ropsten Test Network (https://ropsten.etherscan.io/)**
- **Node.JS (version 14+) (https://nodejs.org/en/)**
- **Hardhat (https://hardhat.org/)**

STEP 1: Metamask

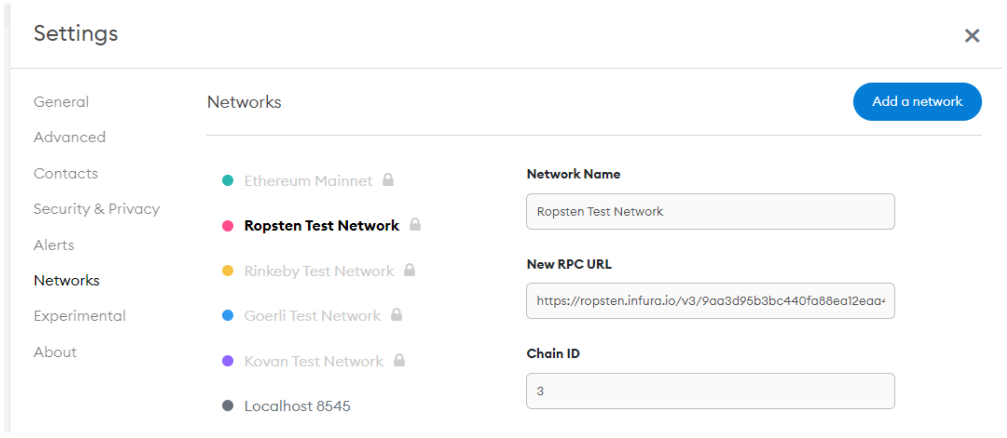
1. https://metamask.io/
2. Download the appropriate browse extension
3. Save your secret key phrase. If you lose this, you can not recover your account

STEP 2: Ropsten Test Network

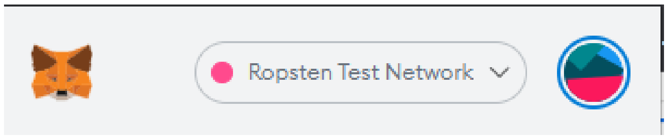
After you have your Metamask wallet set up, connect to the ***Ropsten Test Network.***
You will see your default network is connected to the *Ethereum Mainnet.*
Click on the arrow on the top next to your account circle, and click ***Show/Hide Test Network***



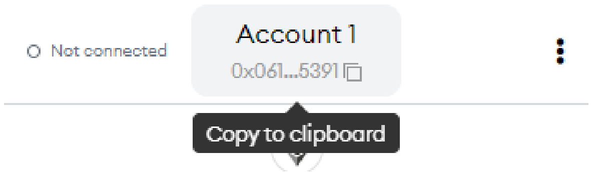
Enable Test Networks and connect to Ropsten Test Network.



Network Name: Ropsten Test Network
New RPC URL: https://ropsten.infura.io/v3/9aa3d95b3bc440fa88ea12eaa
Chain ID: 3

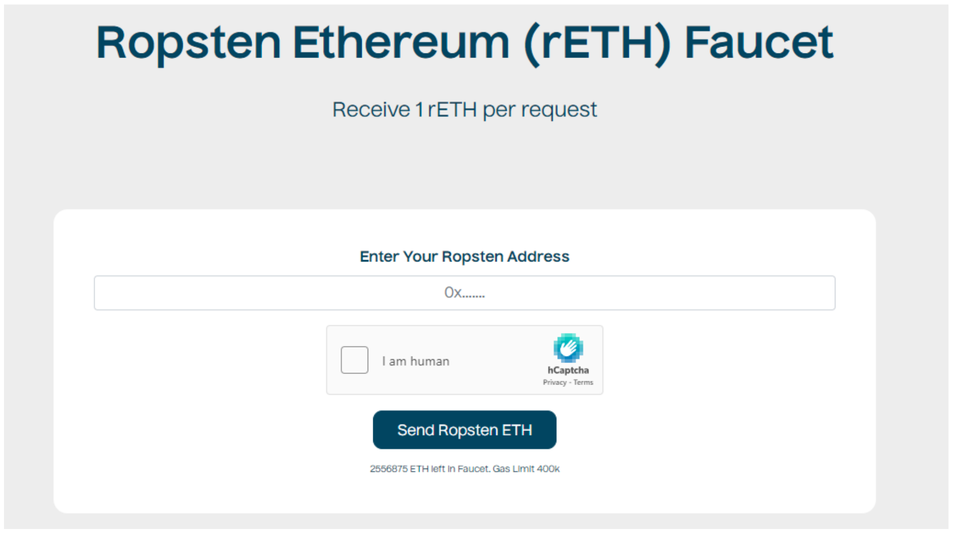


After you have connected to the Ropsten Test Network, copy your wallet address for the next step.



The Ropsten Test Network is a test network most similar to the Ethereum Mainnet.
Test nets are used solely for the purpose of testing smart contract deployment and features.
It allows for developers to examine transaction gas fee usage and make adjustments to their contracts before officially deploying to the Ethereum mainnet.
Testing contracts before Mainnet deployment is essential as the blockchain is immutable.

Go to https://faucet.dimensions.network/
Paste your wallet address to request rETH from the Ropsten faucet.


Ropsten Ethereum (rETH) Faucet
Receive 1 rETH per request
Enter Your Ropsten Address
0x.......
I am human
hCaptcha
Privacy - Terms
Send Ropsten ETH
2556875 ETH left in Faucet. Gas Limit 400k

This ETH is ***not*** real money. It is only for testing purposes.
It will take approximately 2 hours for your rETH to arrive at your wallet.

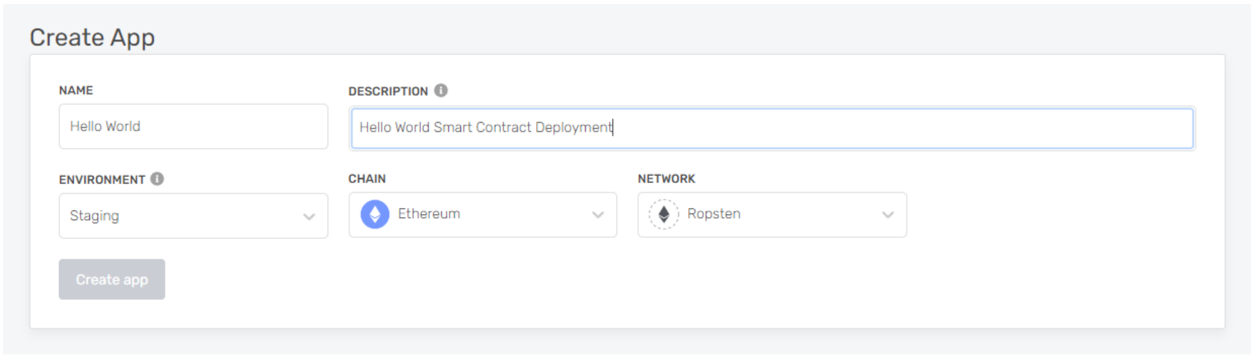
STEP 3: Alchemy

Next, you will make an account on Alchemy, a Web3 developer platform focused on making blockchain development easy.
Follow the next steps.

1. https://www.alchemy.com/
2. Make an account
3. Click Create App and set the next parameters

a. Name: Hello World
b. Description: Hello World Smart Contract Deployment
c. Environment: Staging
d. Chain: Ethereum
e. Network: Ropsten

The selections for this tutorial should look like the image below.


Create App

NAME: Hello World
DESCRIPTION: Hello World Smart Contract Deployment
ENVIRONMENT: Staging
CHAIN: Ethereum
NETWORK: Ropsten

STEP 4: Initialize Project

Now that you have your Metamask and Alchemy set up, it’s time to begin creating the project.
Go to home directory or directory of your choosing in terminal

```
mkdir hello-world
cd hello-world
```

Once you have your project folder, initialize your project with npm.

```
npm init -- yes
```

If you do not have ***npm*** installed, your terminal will direct you on how.
You will also need to have ***Node.JS*** installed. However, the default version that comes with Ubuntu (*v10*) will not suffice. You must manually update your Node.JS to a later version via the following code.

```
curl -sL https://deb.nodesource.com/setup_14.x | sudo -E bash -
```

```
sudo apt-get install -y nodejs
```

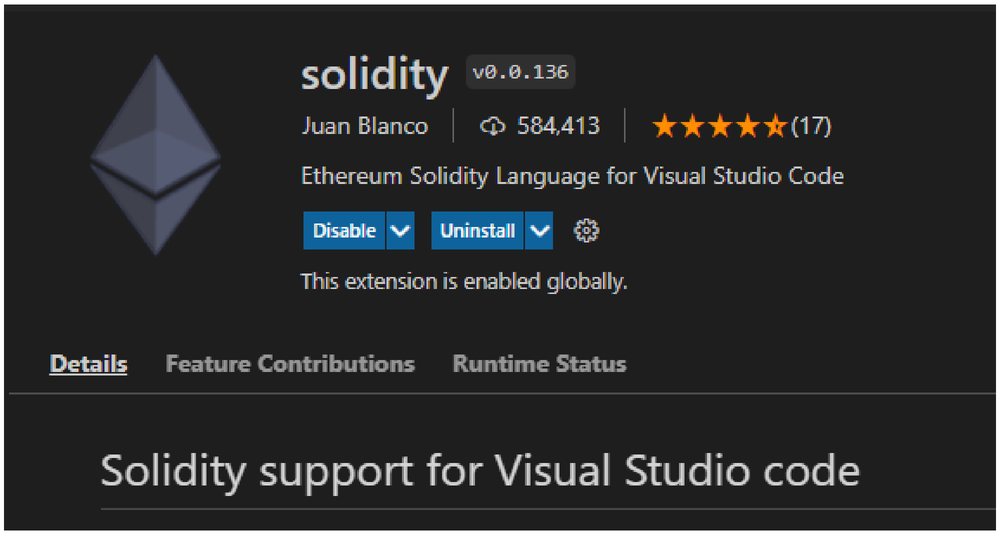
Check your version. It should be ***v14***.

```
node -version
```

```
npm -version
```

You can check out the *package.json* file that’s been created in the terminal or open it in ***VS code*** (**recommended**).

- You can add descriptions via VS code or your editor of choice
- Download the *Solidity plugin* via VS Code


solidity
v0.0.136
Juan Blanco
584,413
(17)
Ethereum Solidity Language for Visual Studio Code
Disable
Uninstall
This extension is enabled globally.
Details
Feature Contributions
Runtime Status
Solidity support for Visual Studio code

STEP 5: Download Hardhat

Hardhat is a development environment for Ethereum Virtual Machine smart contracts.
You can read more at: https://hardhat.org/
Download hardhat

```
npm install --save-dev hardhat
```

You will likely see some warning vulnerabilities but they can be ignored.

STEP 6: Create Hardhat Project

Create Hardhat project

```
npx hardhat
```

This will generate an empty ***hardhat.config.js*** in your folder. This file will be utilized later to specify the setup of the project.

Add project folders

```
mkdir contracts
```

- `contracts/` is where you'll keep your smart contract code file

```
mkdir scripts
```

- `scripts/` is where you'll keep scripts to deploy and interact with your contract

STEP 7: Write Contract

Create a ***HelloWorld.sol*** file in *contracts/*

- Copy the following code into *HelloWorld.sol.*
- You can read more about the Solidity programming language at: https://docs.soliditylang.org/en/v0.8.12/

```
// Specifies the version of Solidity, using semantic versioning.
// Learn more:
https://solidity.readthedocs.io/en/v0.5.10/layout-of-source-files.html#prag
ma
pragma solidity >=0.7.3;

// Defines a contract named `HelloWorld`.
// A contract is a collection of functions and data (its state). Once
deployed, a contract resides at a specific address on the Ethereum
blockchain. Learn more:
https://solidity.readthedocs.io/en/v0.5.10/structure-of-a-contract.html
contract HelloWorld {

   //Emitted when update function is called
   //Smart contract events are a way for your contract to communicate that
something happened on the blockchain to your app front-end, which can be
'listening' for certain events and take action when they happen.
   event UpdatedMessages(string oldStr, string newStr);

   // Declares a state variable `message` of type `string`.
   // State variables are variables whose values are permanently stored in
contract storage. The keyword `public` makes variables accessible from
outside a contract and creates a function that other contracts or clients
can call to access the value.
   string public message;

   // Similar to many class-based object-oriented languages, a constructor
is a special function that is only executed upon contract creation.
   // Constructors are used to initialize the contract's data. Learn
more:https://solidity.readthedocs.io/en/v0.5.10/contracts.html#constructors
   constructor(string memory initMessage) {

      // Accepts a string argument `initMessage` and sets the value into
the contract's `message` storage variable).
      message = initMessage;
   }

   // A public function that accepts a string argument and updates the
`message` storage variable.
   function update(string memory newMessage) public {
      string memory oldMsg = message;
      message = newMessage;
      emit UpdatedMessages(oldMsg, newMessage);
```

```
    }
}
```

STEP 8: Connect Metamask & Alchemy

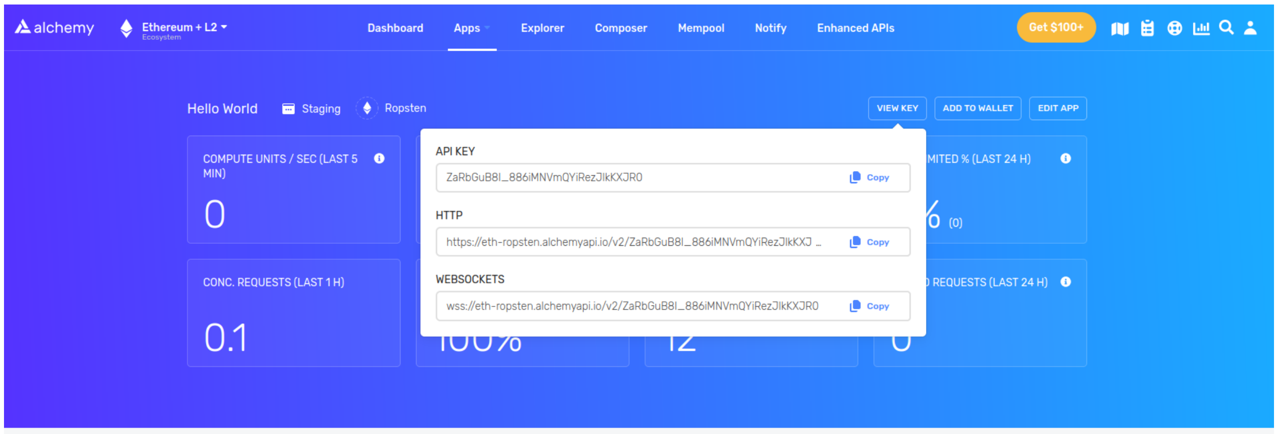
```
npm install dotenv --save
```

Make a **.env** file

- *(dot)env* in your root folder \hello-world

Add ***HTTP Alchemy API URL***

- Available on Alchemy dashboard
- Copy your HTTP Alchemy API URL
- Paste to (dot)env



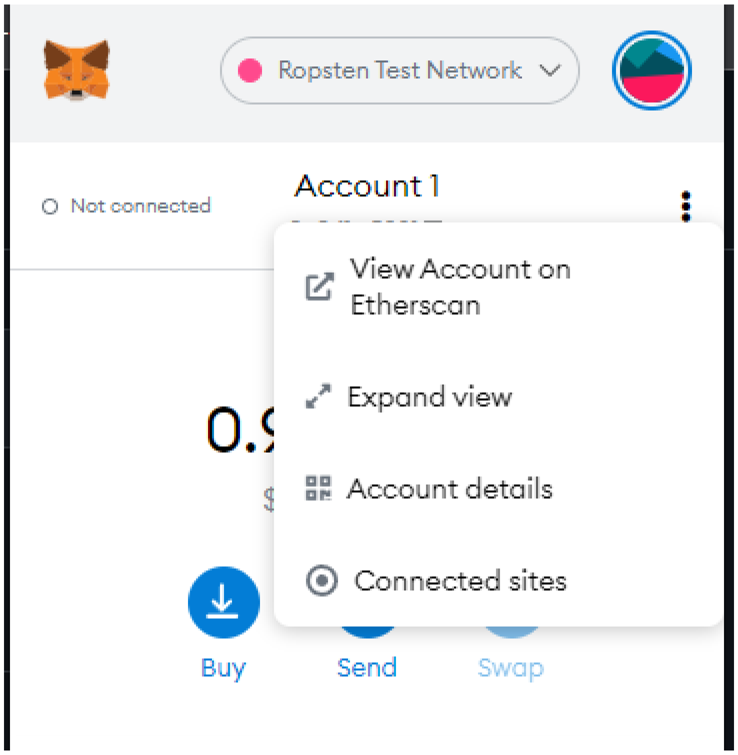
API KEY
ZaRbGuB8I_886iMNVmQYiRezJlkKXJR0

HTTP
https://eth-ropsten.alchemyapi.io/v2/ZaRbGuB8I_886iMNVmQYiRezJlkKXJ ...

WEBSOCKETS
wss://eth-ropsten.alchemyapi.io/v2/ZaRbGuB8I_886iMNVmQYiRezJlkKXJR0

Add Metamask private key

- Click on the 3 vertical dots on the top right under your account circleiota
- Go to account details
- Export private key
- Paste to (dot)env



(dot)env should contain these 2 lines.

```
API_URL = "https://eth-ropsten.alchemyapi.io/v2/your-api-key"
PRIVATE_KEY = "your-metamask-private-key"
```

Make a ***.gitignore*** in root directory

```
touch .gitignore
```

This hides the .env file with your secret keys from the internet

STEP 9: Install *Ethers.js*

Ethers.js is a library that makes it easier to interact and make requests to Ethereum

```
npm install --save-dev @nomiclabs/hardhat-ethers "ethers@^5.0.0"
```

Update your ***hardhat.config.js*** to:

```
/**
* @type import('hardhat/config').HardhatUserConfig
*/

require('dotenv').config();
require("@nomiclabs/hardhat-ethers");

const { API_URL, PRIVATE_KEY } = process.env;
```

```
module.exports = {
   solidity: "0.7.3",
   defaultNetwork: "ropsten",
   networks: {
      hardhat: {},
      ropsten: {
      url: API_URL,
      accounts: [`0x${PRIVATE_KEY}`]
      }
   },
}
```

STEP 10: Compile contract

```
npx hardhat compile
```

`SPDX license identifier not provided in source file` warning can be ignored.

STEP 11: Write deploy script

Go to ***scripts/ folde***r and create ***deploy.js*** and add the below code.

```
async function main() {
   const HelloWorld = await ethers.getContractFactory("HelloWorld");

   // Start deployment, returning a promise that resolves to a contract
object
   const hello_world = await HelloWorld.deploy("Hello World!");
   console.log("Contract deployed to address:", hello_world.address);
}

main()
  .then(() => process.exit(0))
  .catch(error => {
     console.error(error);
     process.exit(1);
  });
```

STEP 12: Deploy contract

```
npx hardhat run scripts/deploy.js --network ropsten
```

You will see your contract deployed on a unique blockchain address:

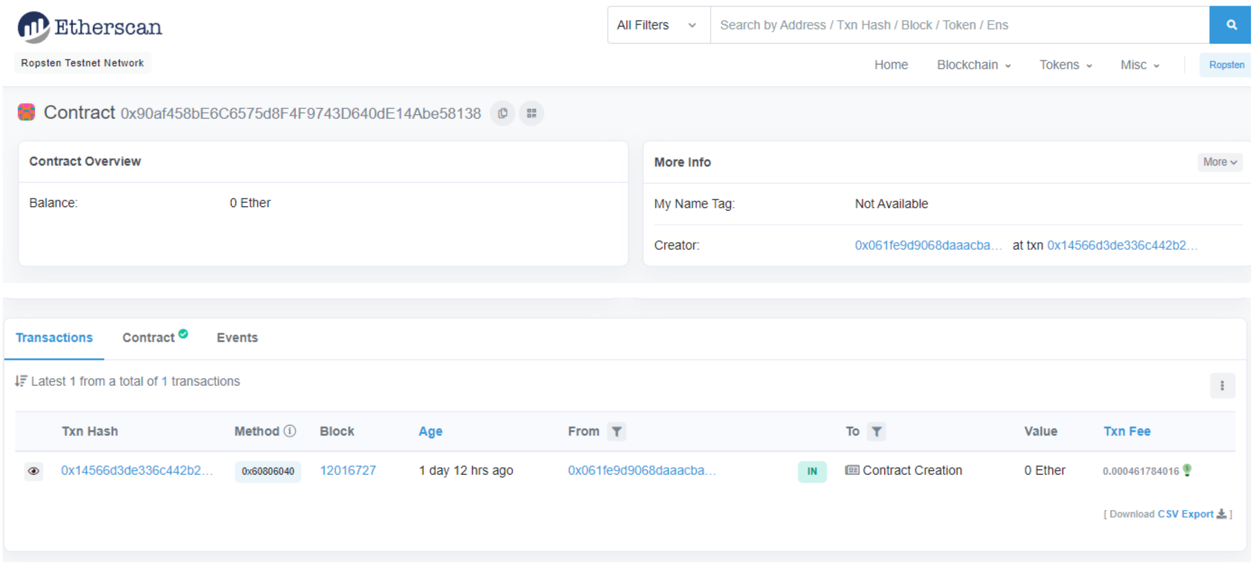
```
Contract deployed to address: 0x6cd7d44516a20882cEa2DE9f205bF401c0d23570
```

Copy and save the address.

STEP 13: View deployed contract

You can view your deployed contract on the Ropsten testnet

- https://ropsten.etherscan.io/
- Copy and paste your deployed address to the search bar



Contract 0x90af458bE6C6575d8F4F9743D640dE14Abe58138

Contract Overview

Balance: 0 Ether

More Info

My Name Tag: Not Available

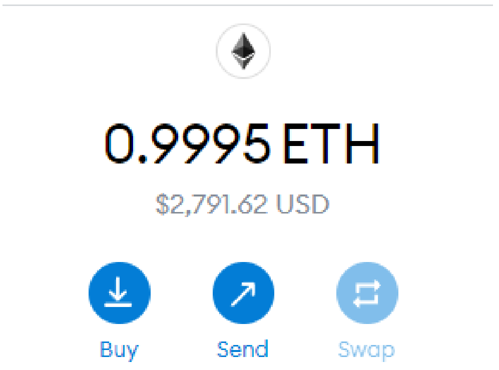
Creator: 0x061fe9d9068daaacba... at txn 0x14566d3de336c442b2...

Latest 1 from a total of 1 transactions

	Txn Hash	Method	Block	Age	From		To	Value	Txn Fee
	0x14566d3de336c442b2...	0x60806040	12016727	1 day 12 hrs ago	0x061fe9d9068daaacba...	IN	Contract Creation	0 Ether	0.000461784016

[Download CSV Export]

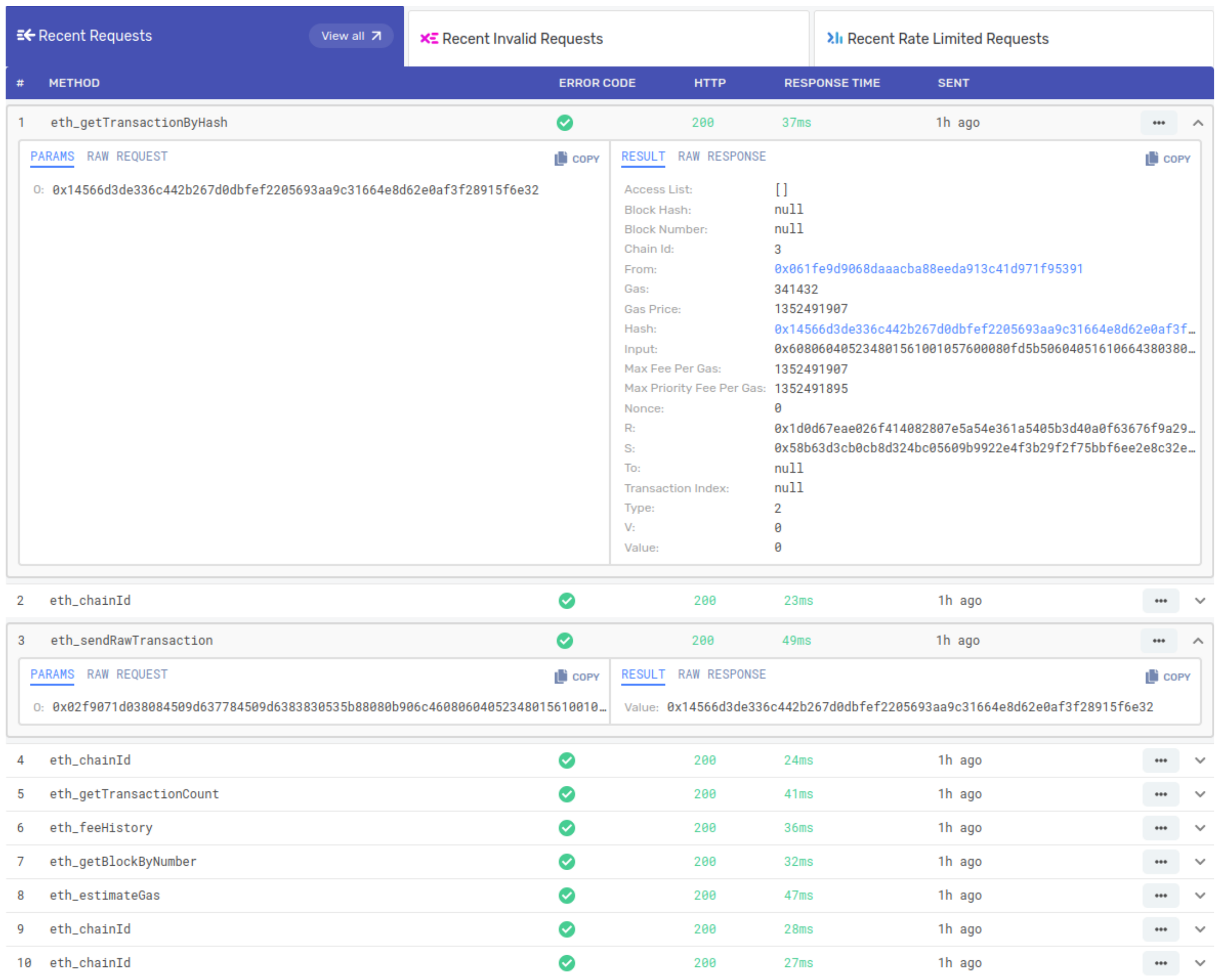
- Check your metamask Eth balance. You will see a small deduction that was used as a transaction fee to deploy your contract.


0.9995 ETH

$2,791.62 USD

- For this deployment, it cost me 0.005 eth, approx $14 at the current price of 1 ethereum.

You can also go to your *Alchemy* dashboard to see what's going on under the hood of your contract deployment.



#	METHOD	ERROR CODE	HTTP	RESPONSE TIME	SENT
1	eth_getTransactionByHash	✓	200	37ms	1h ago
2	eth_chainId	✓	200	23ms	1h ago
3	eth_sendRawTransaction	✓	200	49ms	1h ago
4	eth_chainId	✓	200	24ms	1h ago
5	eth_getTransactionCount	✓	200	41ms	1h ago
6	eth_feeHistory	✓	200	36ms	1h ago
7	eth_getBlockByNumber	✓	200	32ms	1h ago
8	eth_estimateGas	✓	200	47ms	1h ago
9	eth_chainId	✓	200	28ms	1h ago
10	eth_chainId	✓	200	27ms	1h ago

Request 1 — PARAMS / RAW REQUEST

```
0: 0x14566d3de336c442b267d0dbfef2205693aa9c31664e8d62e0af3f28915f6e32
```

Request 1 — RESULT / RAW RESPONSE

Access List:	[]
Block Hash:	null
Block Number:	null
Chain Id:	3
From:	0x061fe9d9068daaacba88eeda913c41d971f95391
Gas:	341432
Gas Price:	1352491907
Hash:	0x14566d3de336c442b267d0dbfef2205693aa9c31664e8d62e0af3f…
Input:	0x6080604052348015610010576000080fd5b50604051610664380380…
Max Fee Per Gas:	1352491907
Max Priority Fee Per Gas:	1352491895
Nonce:	0
R:	0x1d0d67eae026f414082807e5a54e361a5405b3d40a0f63676f9a29…
S:	0x58b63d3cb0cb8d324bc05609b9922e4f3b29f2f75bbf6ee2e8c32e…
To:	null
Transaction Index:	null
Type:	2
V:	0
Value:	0

Request 3 — PARAMS / RAW REQUEST

```
0: 0x02f9071d038084509d637784509d6383830535b88080b906c46080604052348015610010…
```

Request 3 — RESULT / RAW RESPONSE

```
Value: 0x14566d3de336c442b267d0dbfef2205693aa9c31664e8d62e0af3f28915f6e32
```

See the *JSON-RPC* calls that Hardhat/Ethers made when you called the `.deploy()` function from *deploy.js.* For example:

```
eth_sendRawTransaction
```

Requests to write contract on the Ropsten testnet

```
eth_getTransactionByHash
```

Requests to read info about your transaction given the hash

Congratulations!
You have successfully deployed a *Hello World* smart contract onto the Ropsten test network.

If you'd like to deploy on other testnets, please refer to the Part 2 document.