Major Project Portfolio

Building a Cryptocurrency
on The ERC-20 Blockchain

Five Islands Secondary College

Multi-Media class of 2021

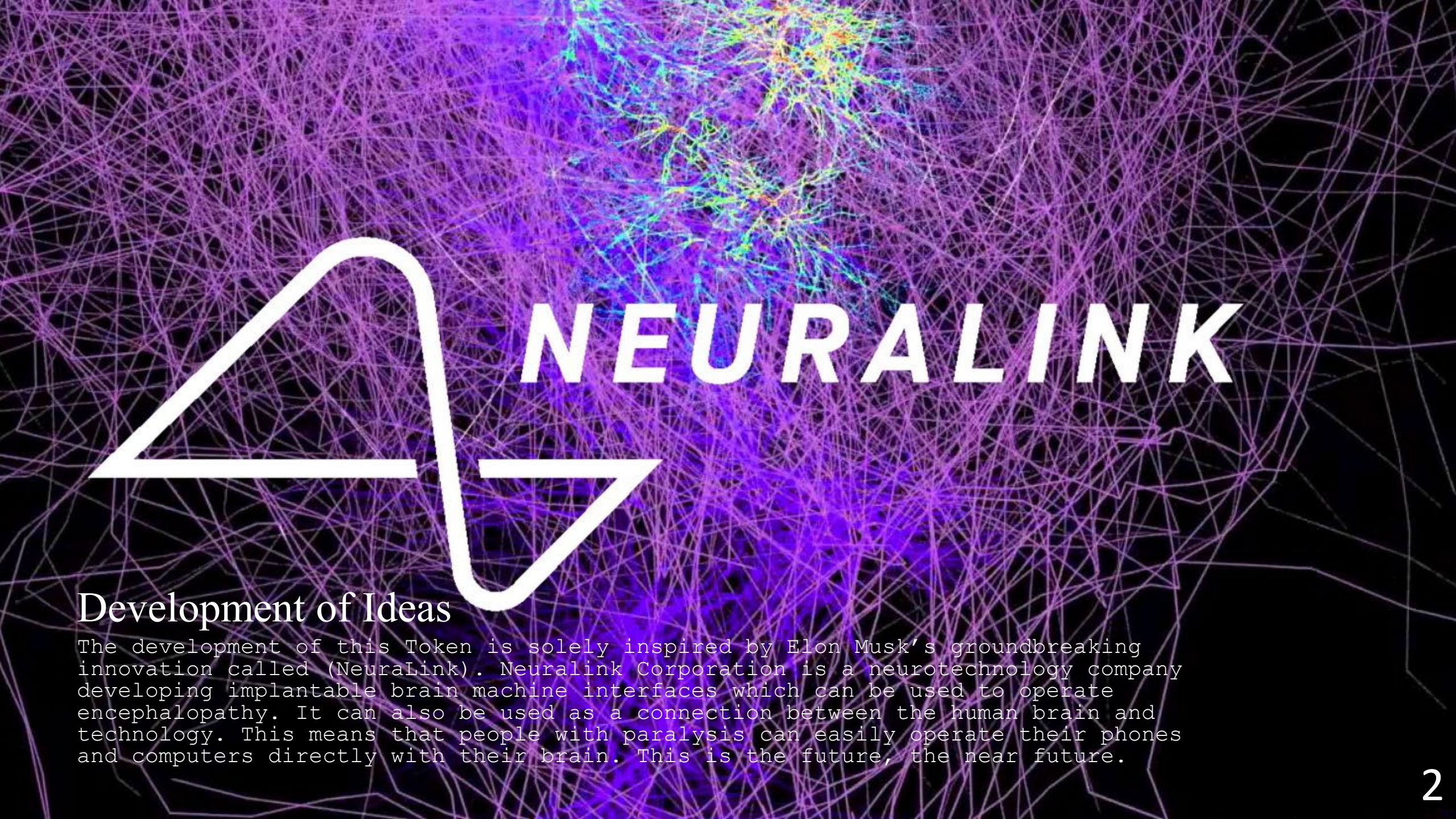
Created by Nate Mitchell Ryan

Student Number: 33873328

Glossary

Statement of Intent

Building a Cryptocurrency token on the Ethereum blockchain (ERC20). I intend to publish and deploy the currency to the online market. I'm building a cryptocurrency inspired by a ground-breaking innovation with outstanding abilities. With this coin I intend to not only promote the company of choice but further this coin to a company itself. I'm all for the benefit a company has on its society so that is my intention.



Development of Ideas

The development of this Token is solely inspired by Elon Musk's groundbreaking innovation called (NeuraLink). Neuralink Corporation is a neurotechnology company developing implantable brain machine interfaces which can be used to operate encephalopathy. It can also be used as a connection between the human brain and technology. This means that people with paralysis can easily operate their phones and computers directly with their brain. This is the future, the near future.

Development of ideas

The development inspiration is being involved in something that I thought would have a significant impact on the world. Anyone can build a company but not everyone can run a company, especially a company that benefits the world. That is why I chose NeuraLink.

Development of Ideas

Cryptocurrency, the stock market, the future. Is my passion as I also want to further my education into University and study Computer Science to become a Programmer or Artificial Intelligence Engineer. This is a major reason for the development of a Cryptocurrency.

Development of Ideas

I also scanned through Cryptocurrency exchanges. My reason for that is to gather information about their Token supply, if and what company they base their coin around, compare with them to other coins of the same level, as well as comparing success rates to each other by analyzing graph reports and the time in which the coins have been on the market.

Development of Ideas

I want to be different so that is why I have put countless days into building this currency. I want to make a difference in the society we live in and continuously improve our world by innovating and discovering the undiscoverable. I believe being that one step ahead of people is unsurpassed, and with that comes unprecedented success. Failure is an option, If you don’t fail, you’re not innovating, and When something is important enough, you do it even if the odds are not in your favor.

The Building Process

Building the contract

Essentially, a token contract is a smart contract that contains a map of account addresses and their balances. The balance represents a value that is defined by the contract creator: one token contract might use balances to represent physical objects, another monetary value, and a third the holder’s reputation. The unit of this balance is commonly called a token.

When tokens are transferred from one account to another the token contract updates the balance of the two accounts.

An ERC-20 token contract is defined by the contract’s address and the total supply of tokens available to it but has a number of optional items that are usually provided as well to provide more detail to users. These are the token’s name, its symbol, and the number of decimals.

The Building process

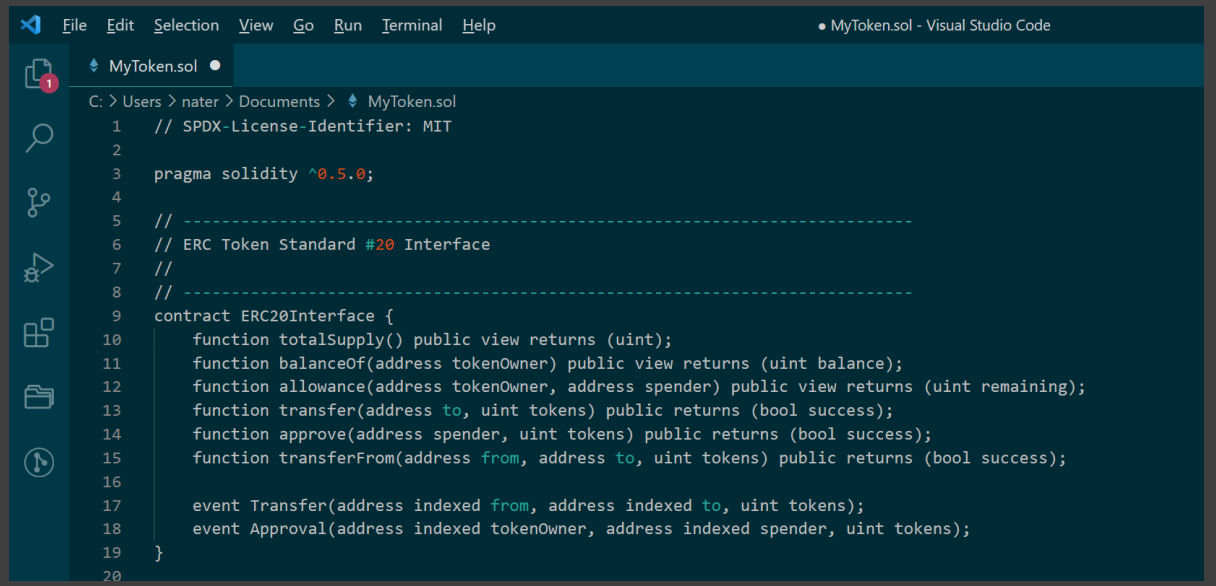
I begin coding and start by identifying my contract license which is, license-MIT. This allows me to legally build a token on the ERC20 blockchain.

Following that I Identify the language I will be using as well as the version to construct the token which is, Solidity (sol) and version 5.0

Now after I have identified the license and solidity version, I can comfortably move on with the contract and constructor functions.

Firstly, I identify the contract, Token name and The blockchain which in my case is the ERC20 blockchain.

The photo on the right is a visual example from my contract.



The Building Process

Now after I have identified the contract, I can move on to calling the functions. Functions are (self-contained) modules of code that accomplish a specific task. Functions usually take in data, process it, and return a result. Once a function is written, it can be used over and over and over again. Functions can be called from the inside of other functions.

```
function name() public view virtual override returns (string memory) {
    return _name=;
}


function symbol() public view virtual override returns (string memory) {
    return _symbol;
}


function decimals() public view virtual override returns (uint8) {
    return 18;
}


function totalSupply() public view virtual override returns (uint256) {
    return _totalSupply;
}


function balanceOf(address account) public view virtual override returns (uint256) {
    return _balances[account];
}


function transfer(address recipient, uint256 amount) public virtual override returns (bool) {
    _transfer(_msgSender(), recipient, amount);
    return true;
}


function allowance(address owner, address spender) public view virtual override returns (uint256) {
    return _allowances[owner][spender];
}

function approve(address spender, uint256 amount) public virtual override returns (bool) {
    _approve(_msgSender(), spender, amount);
    return true;
}
```

```


function transferFrom(
    address sender,
    address recipient,
    uint256 amount
) public virtual override returns (bool) {
    _transfer(sender, recipient, amount);

    uint256 currentAllowance = _allowances[sender][_msgSender()];
    require(currentAllowance >= amount, "ERC20: transfer amount exceeds allowance");
    unchecked {
        _approve(sender, _msgSender(), currentAllowance - amount);
    }

    return true;
}


function increaseAllowance(address spender, uint256 addedValue) public virtual returns (bool) {
    _approve(_msgSender(), spender, _allowances[_msgSender()][spender] + addedValue);
    return true;
}


function decreaseAllowance(address spender, uint256 subtractedValue) public virtual returns (bool) {
    uint256 currentAllowance = _allowances[_msgSender()][spender];
    require(currentAllowance >= subtractedValue, "ERC20: decreased allowance below zero");
    unchecked {
        _approve(_msgSender(), spender, currentAllowance - subtractedValue);
    }

    return true;
}
```

The Building Process

token bot called(Safe-Math) and Safe-Math is put in place for whenever someone is buying or receiving my coin and what it does is automatically sends the correct number of Tokens to the buyer, as well as the correct amount of profit to the token owner.

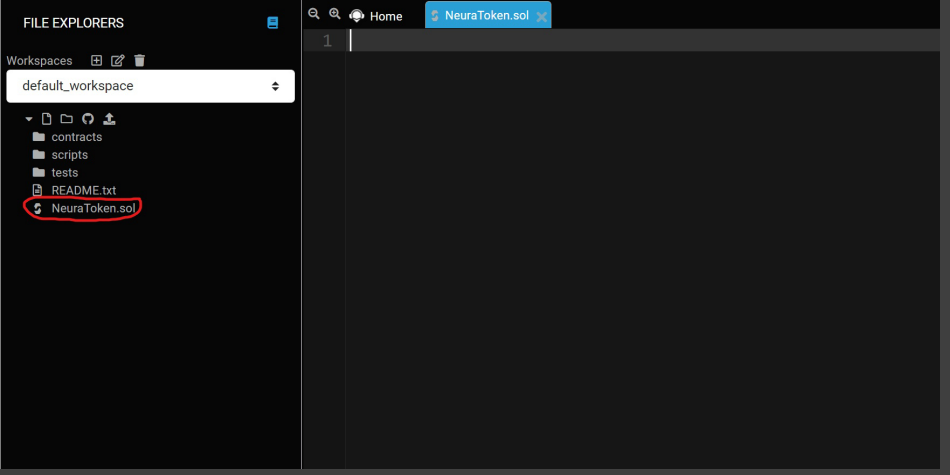
This is what it should look like.

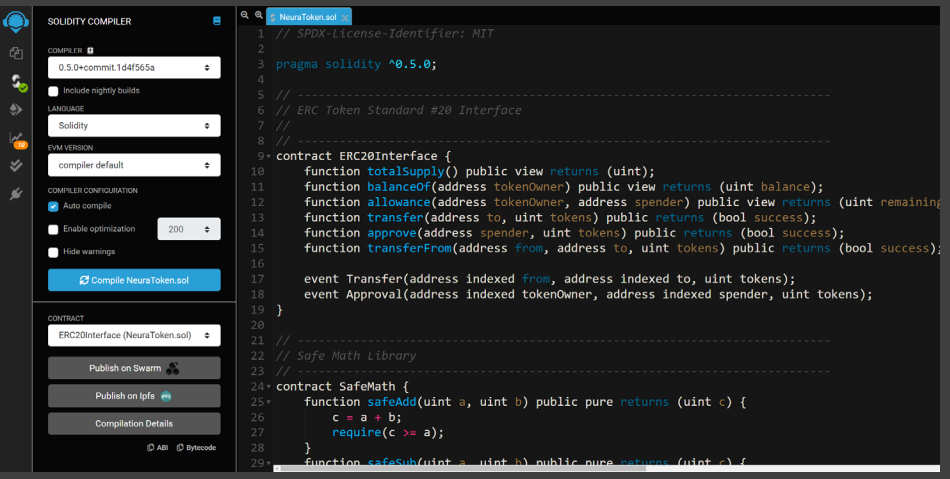
```
library SafeMath {

    function tryAdd(uint256 a, uint256 b) internal pure returns (bool, uint256) {
        unchecked {
            uint256 c = a + b;
            if (c < a) return (false, 0);
            return (true, c);
        }
    }


    function trySub(uint256 a, uint256 b) internal pure returns (bool, uint256) {
        unchecked {
            if (b > a) return (false, 0);
            return (true, a - b);
        }
    }


    function tryMul(uint256 a, uint256 b) internal pure returns (bool, uint256) {
        unchecked {
            if (a == 0) return (true, 0);
            uint256 c = a * b;
            if (c / a != b) return (false, 0);
            return (true, c);
        }
    }
```

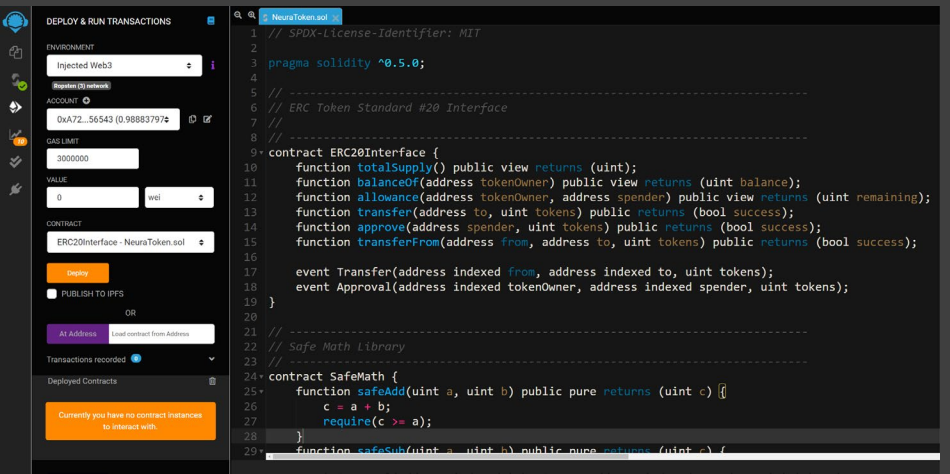
ERC20.sol | SafeMath.sol | MyToken.sol

C: > Users > nater > SafeMath.sol

```
    function add(uint256 a, uint256 b) internal pure returns (uint256) {
        return a + b;
    }

    function sub(uint256 a, uint256 b) internal pure returns (uint256) {
        return a - b;
    }

    function mul(uint256 a, uint256 b) internal pure returns (uint256) {
        return a * b;
    }

    function div(uint256 a, uint256 b) internal pure returns (uint256) {
        return a / b;
    }

    function mod(uint256 a, uint256 b) internal pure returns (uint256) {
        return a % b;
    }

    function sub(
        uint256 a,
        uint256 b,
        string memory errorMessage
    ) internal pure returns (uint256) {
        unchecked {
            require(b <= a, errorMessage);
            return a - b;
        }
    }

    function div(
        uint256 a,
        uint256 b,
        string memory errorMessage
    ) internal pure returns (uint256) {
        unchecked {
            require(b > 0, errorMessage);
            return a / b;
        }
    }

    function mod(
        uint256 a,
        uint256 b,
        string memory errorMessage
    ) internal pure returns (uint256) {
        unchecked {
            require(b > 0, errorMessage);
            return a % b;
        }
    }
}
```





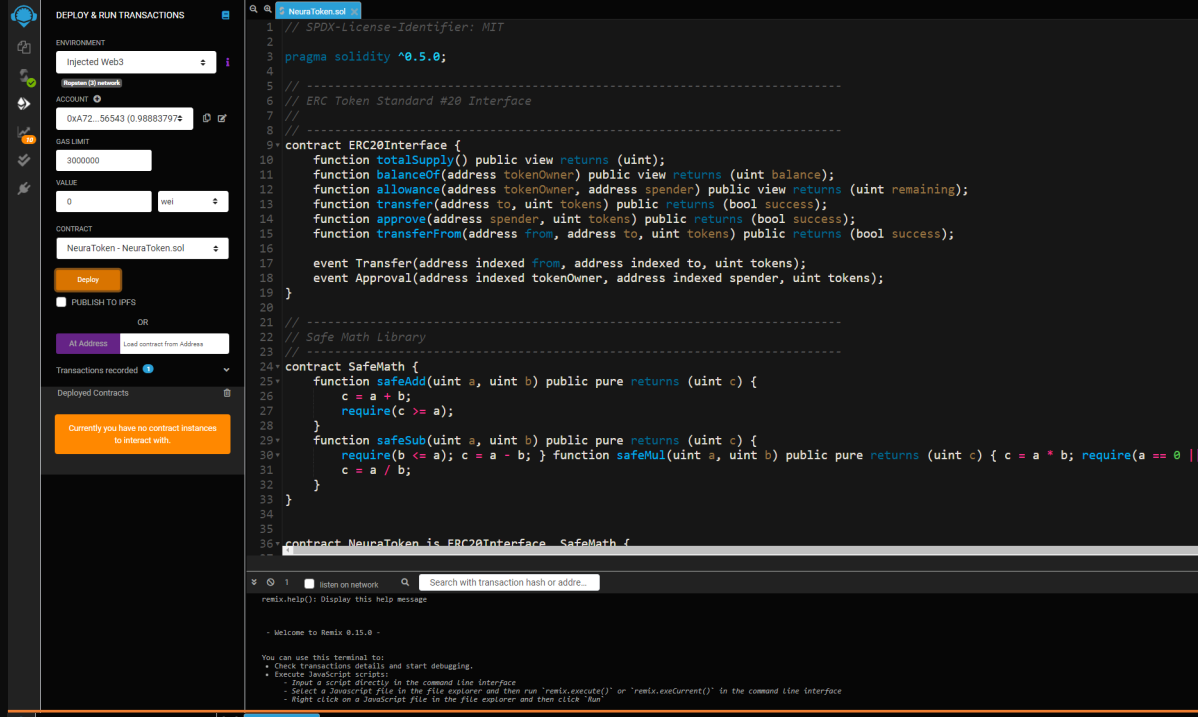


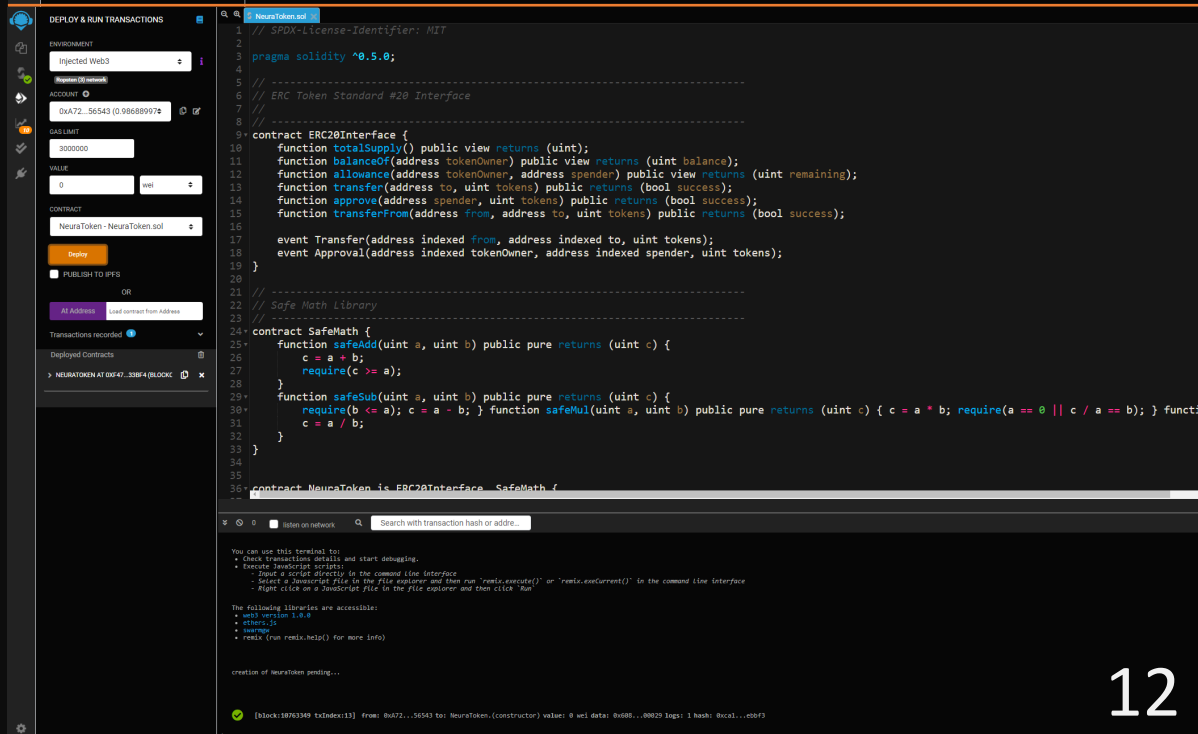
Compile Process

- The Compile Process is where take you contract and run in on an Ethereum Contract complier. Ultimately where your contract goes to get approved. It also compiles multiple files into one big contact.
- So fisrt things first I go to Remix.org and click on the file icon and type my contract name which is circled in red.
- Secondly copy and paste my contract into the remix compiler and a green tick should pop up next to the solidity icon that means I'm ready to compile.
- I press compile and click on the icon below where it brings me to the deploy page.

Deploy Process

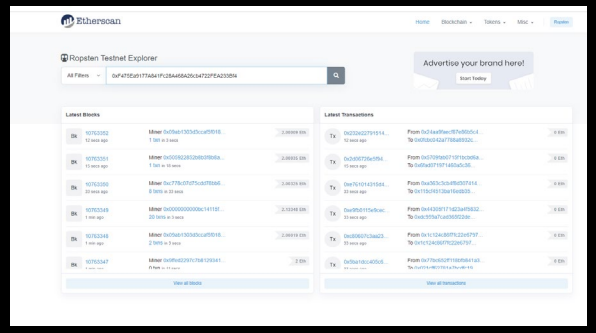
- Now I click delpy and below my contract it tells me that it is pending, and I wait until It has successfulliy deployed and how I know that it has successfully deployed is by a green tick below the pending bar.

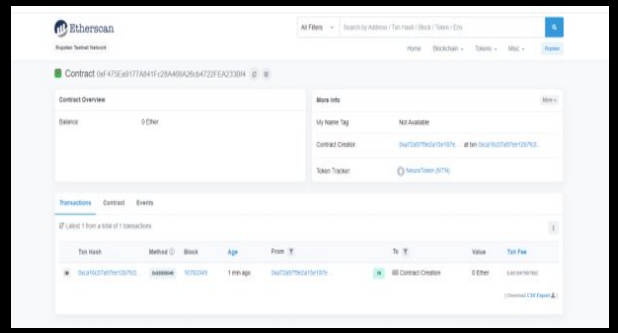


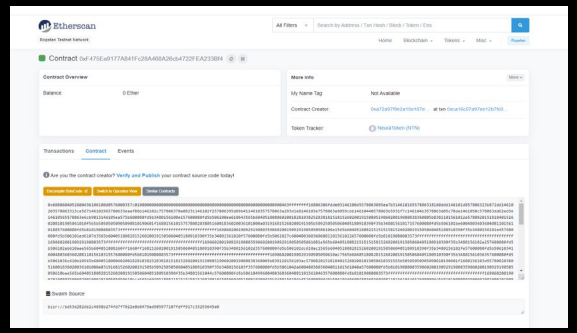


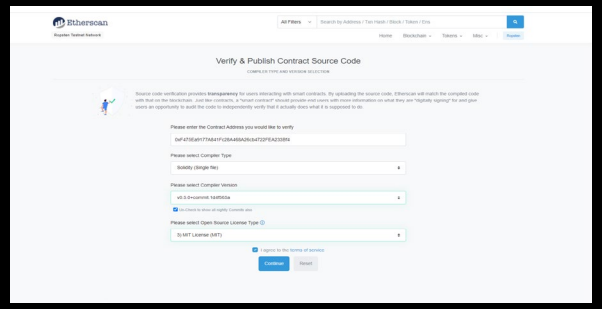
Verification Process

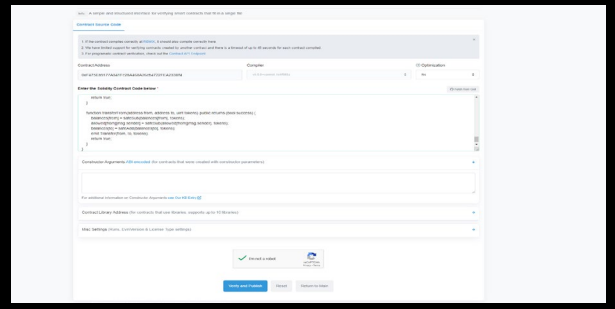
Then I copy my contract address and go to a site called etherscan and paste my address on the search bar and press search. Then I'm brang to a page where my contract is and below the address there is a tab called contract, I click on that and and brang to a page that says verify and publish. I click on that and follow the instructions. Finally, I press (Verify and Publish) let it process for a minute. Then a message saying I have successfully created a contract.

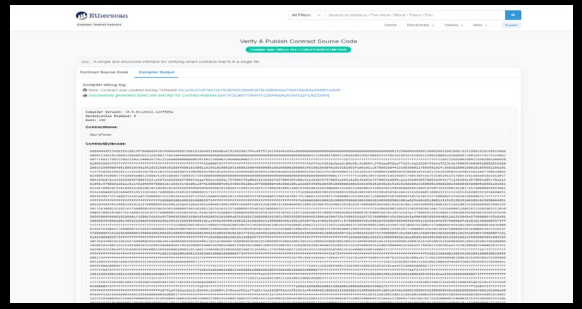


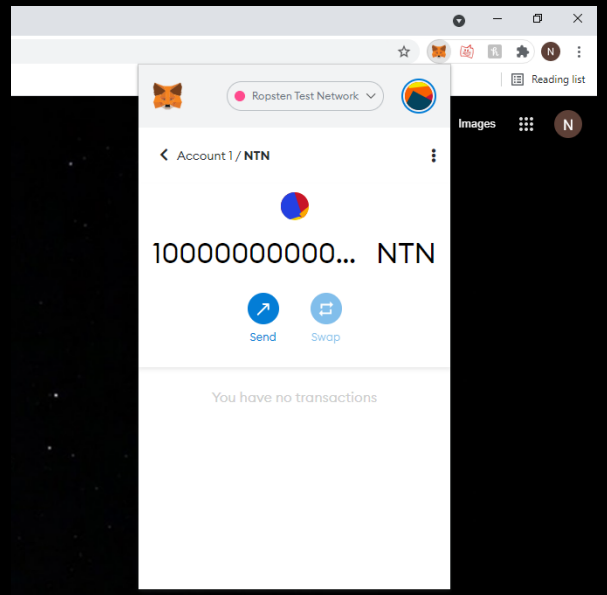


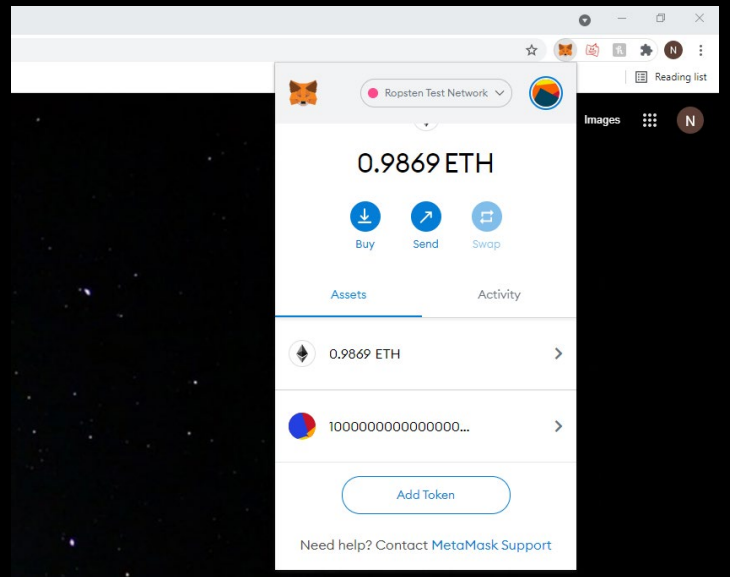


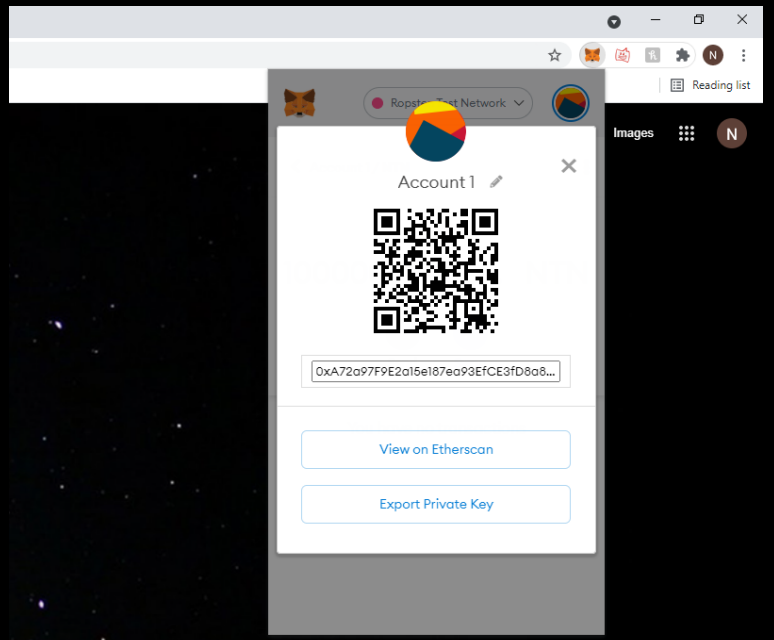










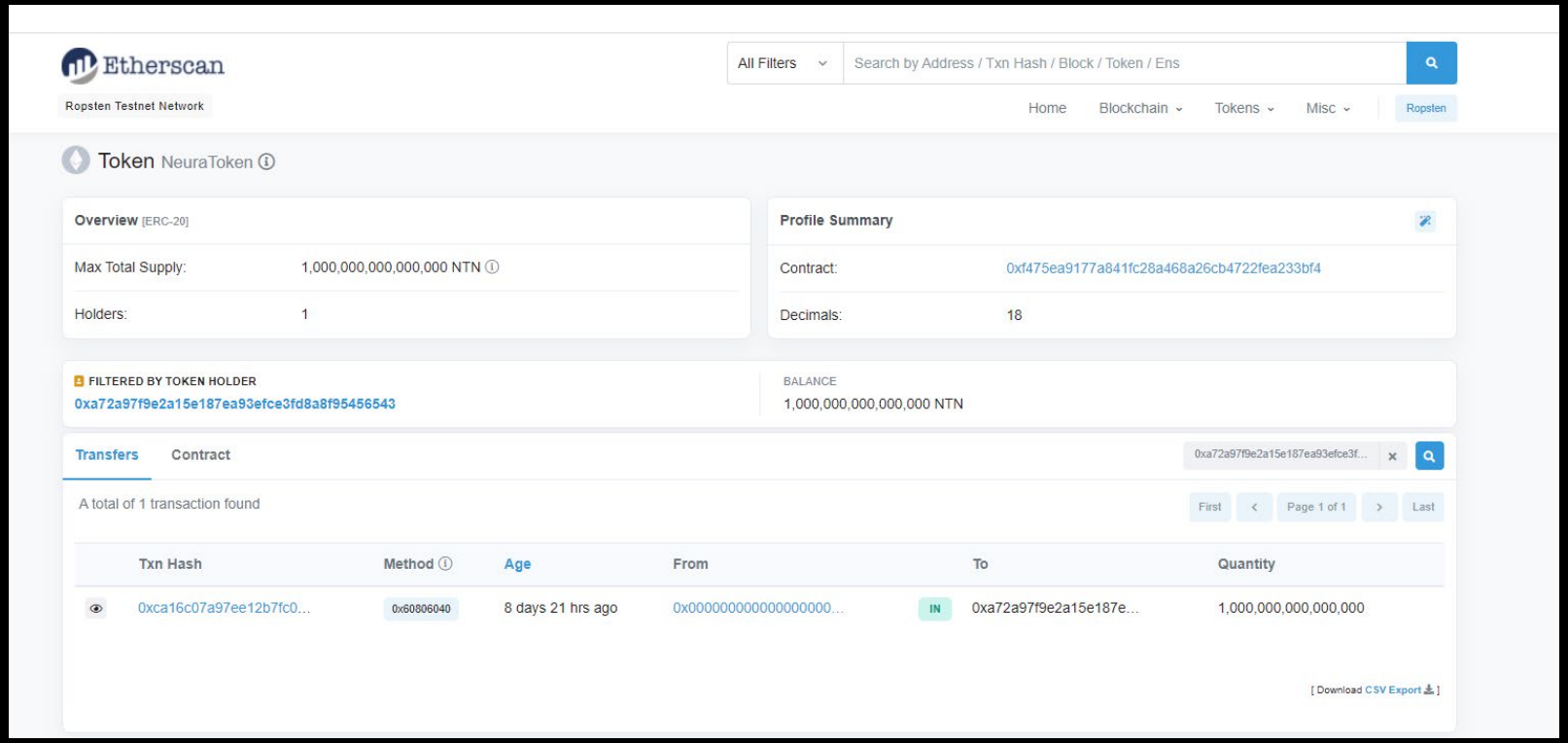


Meta Mask

- Meta Mask is a wallet for holding digital assets. It also what you would use when you deploy your token to the mainnet.

Token page

Here is my Token on the web



Programming Language

Solidity is **an object-oriented programming language for writing smart contracts**. It is used for implementing smart contracts on various blockchain platforms, most notably, Ethereum. ... The programs compiled by the Solidity are intended to be run on Ethereum Virtual Machine.

Hardware

Central Processing Unit (CPU)

Purpose:

The purpose of the CPU is to process data . The CPU is where processes such as calculating, sorting and searching take place. Whatever is done on our computers, such as checking emails, playing games and doing homework, the CPU has processed the data we use.

Graphics Processing Unit (GPU)

Purpose:

What does GPU stand for? Graphics processing unit, a specialized processor originally designed to accelerate graphics rendering. GPUs can process many pieces of data simultaneously, making them useful for machine learning, video editing, and gaming applications.

Random access Memory (RAM)

Purpose:

Computer random access memory (RAM) is one of the most important components in determining your system's performance. RAM gives applications a place to store and access data on a short-term basis. It stores the information your computer is actively using so that it can be accessed quickly.

Hard Drive Disk (HDD)

Purpose:

What Does a Hard Drive Do? A hard drive is the hardware component that stores all of your digital content. Your documents, pictures, music, videos, programs, application preferences, and operating system represent digital content stored on a hard drive. Hard drives can be external or internal.

How the Hardware benefits me: The CPU processes my work fast and efficiently allowing me to work with full ability. The GPU allows the graphics to process and pixelate fast being able to run my software comfortably. The RAM helps me store the information im constantly using for easy access. The HDD helps me store all my digital content for safe use.

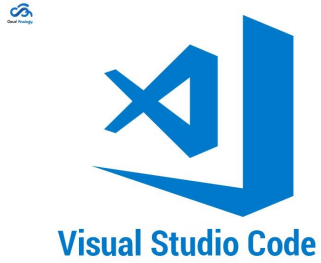
Software

Windows 10

Details:

One of the primary aims of Windows 10 is to unify the Windows experience across multiple devices, such desktop computers, tablets, and smartphones. As part of this effort, Microsoft developed Windows 10 Mobile alongside Windows 10 to replaces Windows Phone – Microsoft's previous mobile OS.

Visual Studio Code



Details:

Visual Studio Code is a streamlined code editor with support for development operations like debugging, task running, and version control. It aims to provide just the tools a developer needs for a quick code-build-debug cycle and leaves more complex workflows to fuller featured IDEs, such as Visual Studio IDE.

Microsoft word

Details:

Microsoft Word or MS Word (often called Word) is a graphical word processing program that users can type with. It is made by the computer company Microsoft. Its purpose is to allow users to type and save documents. Like other word processors, it has helpful tools to make documents.

PowerPoint

Details:

The purpose of PowerPoint is to act as a visual aid as a presenter goes along presenting their option, ideas, sales pitch, etc. Make sure to not make your slides too wordy and concentrate on adding only basic bullet points.

Overview:

Windows 10 is the life behind the other software's. Ill use Visual Studio Code as my main software, where I build my trading algorithm. Word is where I report my progress and PowerPoint is for my portfolio.

Hardware

Motherboard

Purpose:

The motherboard is the backbone that ties the computer's components together at one spot and allows them to talk to each other. Without it, none of the computer pieces, such as the CPU, GPU, or hard drive, could interact. Total motherboard functionality is necessary for a computer to work well.

Keyboard

Purpose:

A computer keyboard is an input device that allows a person to enter letters, numbers, and other symbols (these are called characters in a keyboard) into a computer. It is one of the most used input devices for computers. Using a keyboard to enter lots of data is called typing.

Mouse

Purpose:

A computer mouse is a handheld hardware input device that controls a cursor in a GUI and can move and select text, icons, files, and folders. For desktop computers, the mouse is placed on a flat surface such as a mouse pad or a desk and is placed in front of your computer.

Mousepad

Purpose:

A mousepad enhances the usability of the mouse compared to using a mouse directly on a table by providing a surface to allow it to measure movement accurately and without jitter. Some mousepads increase ergonomics by providing a padded wrist rest, although the benefits of this are debatable.

Flash Drive

Purpose:

USB flash drives are often used for storage, data back-up and transferring of computer files. Compared with floppy disks or CDs, they are smaller, faster, have significantly more capacity, and are more durable due to a lack of moving parts.

Monitor

Purpose:

The term "monitor" is often used synonymously with "computer screen" or "display." The monitor displays the computer's user interface and open programs, allowing the user to interact with the computer, typically using the keyboard and mouse.

Token Address

- This is my Token address

0xF475Ea9177A841Fc28A468A26cb4722FEA233Bf4

This is the Link to my Token VIA ropsten.etherscan.io

https://ropsten.etherscan.io/address/0xF475Ea9177A841Fc28A468A26cb4722FEA233Bf4