

A.2 Write a contract to execute basic addition, subtraction, multiplication and division functions and Simulate on metamask on Ropsten network

Smart contract

```
pragma solidity ^0.5.17;

contract operations {
    uint n1;
    uint n2;

    function firstNumber(uint a) public {
        n1 = a;
    }

    function secondNumber(uint b) public {
        n2 = b;
    }

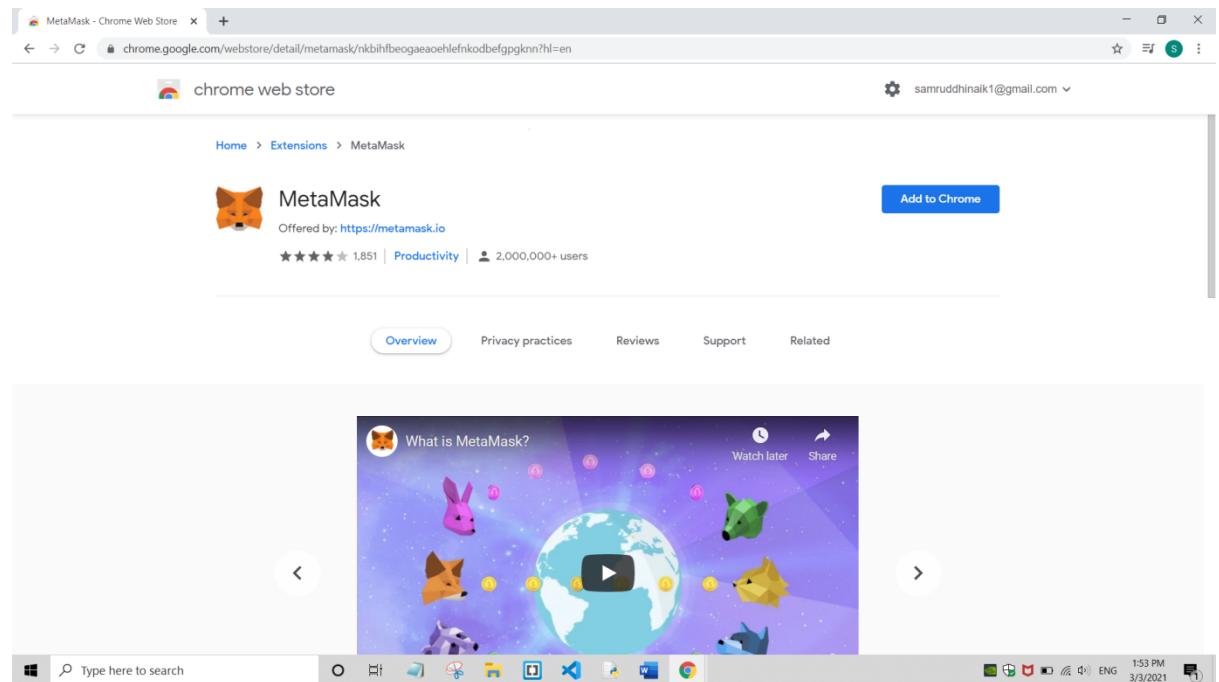
    function add() view public returns (uint) {
        return n1 + n2;
    }

    function subtract() view public returns (uint) {
        return n1 - n2;
    }

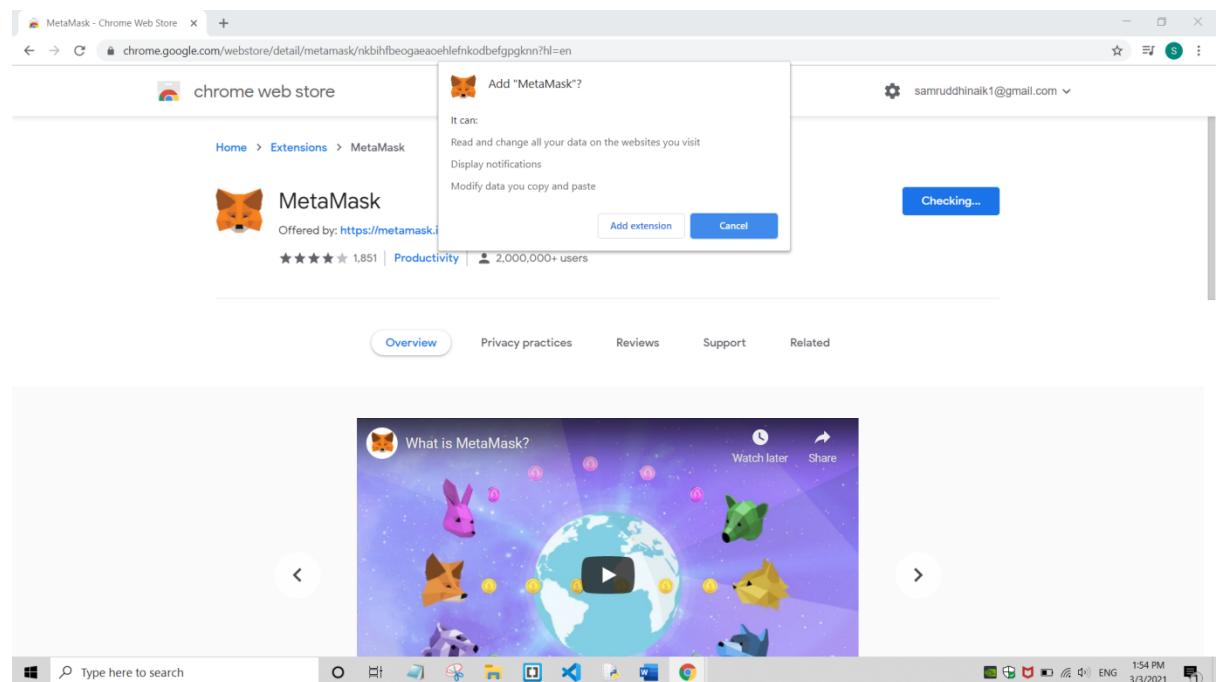
    function multiply() view public returns (uint) {
        return n1 * n2;
    }

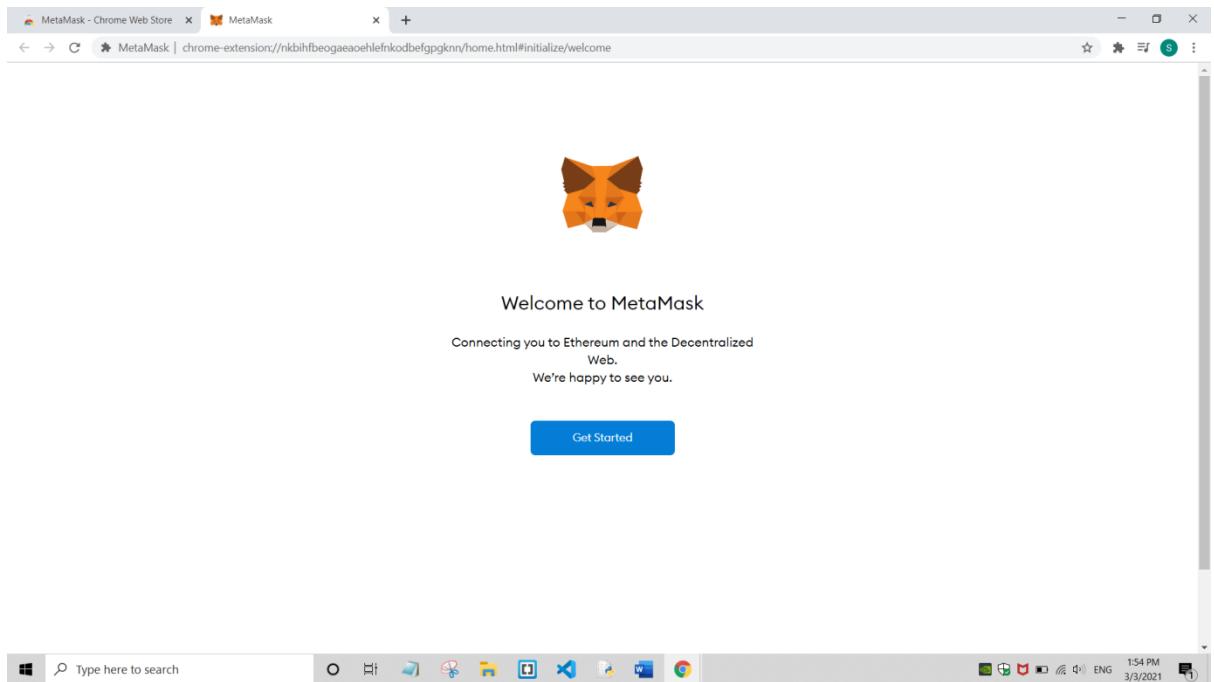
    function divide() view public returns (uint) {
        return n1 / n2;
    }
}
```

Metamask extension

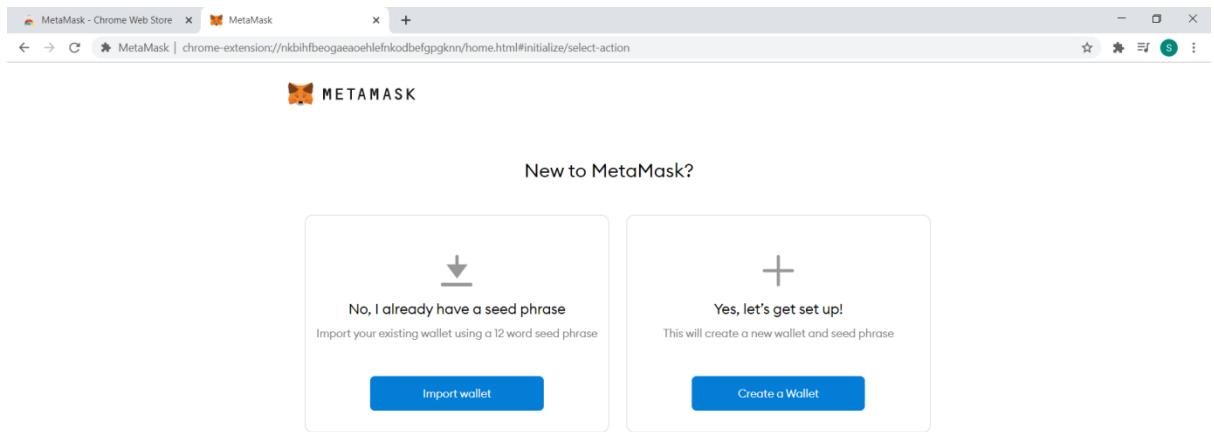


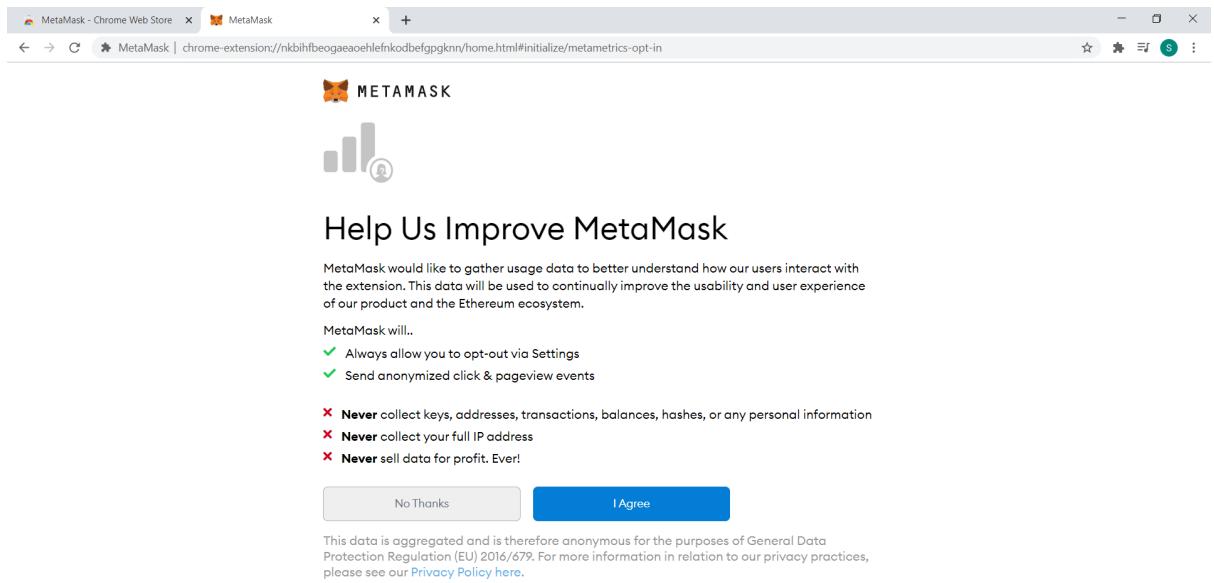
Add extension





Create a Wallet

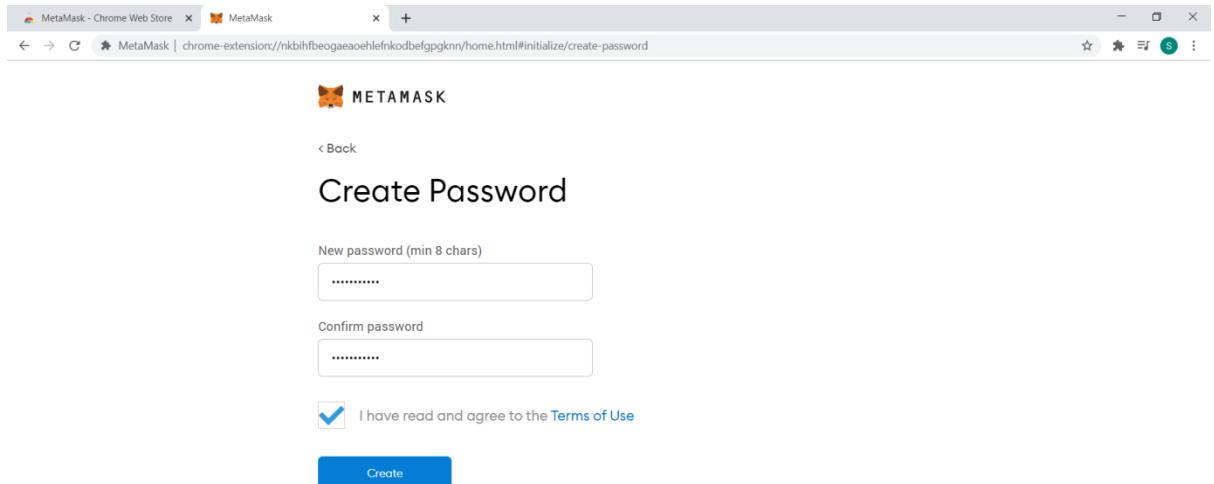




The screenshot shows a web browser window with two tabs: "MetaMask - Chrome Web Store" and "MetaMask". The active tab displays the MetaMask logo and a message asking for user consent to gather usage data. The message states: "MetaMask would like to gather usage data to better understand how our users interact with the extension. This data will be used to continually improve the usability and user experience of our product and the Ethereum ecosystem." It then lists what MetaMask will and will not do:

- ✓ Always allow you to opt-out via Settings
- ✓ Send anonymized click & pageview events
- ✗ Never collect keys, addresses, transactions, balances, hashes, or any personal information
- ✗ Never collect your full IP address
- ✗ Never sell data for profit. Ever!

At the bottom are two buttons: "No Thanks" and "I Agree". Below the buttons, a note states: "This data is aggregated and is therefore anonymous for the purposes of General Data Protection Regulation (EU) 2016/679. For more information in relation to our privacy practices, please see our [Privacy Policy](#) here."



The screenshot shows a web browser window with two tabs: "MetaMask - Chrome Web Store" and "MetaMask". The active tab displays the MetaMask logo and a "Create Password" form. The form includes fields for "New password (min 8 chars)" and "Confirm password", both containing placeholder text ".....". Below the fields is a checkbox labeled "I have read and agree to the [Terms of Use](#)". At the bottom is a blue "Create" button.



The screenshot shows the 'Secret Backup Phrase' step of the MetaMask setup process. At the top, there's a 'Tips:' section with two items: 'Store this phrase in a password manager like 1Password.' and 'Write this phrase on a piece of paper and store in a secure location. If you want even more security, write it down on multiple pieces of paper and store each in 2 - 3 different locations.' Below these tips is a large button labeled 'CLICK HERE TO REVEAL SECRET WORDS' with a lock icon, which is highlighted with a red box. At the bottom are two buttons: 'Remind me later' and 'Next'.



Get the Secret Backup Phrase

Use the Secret Backup Phrase to access Account

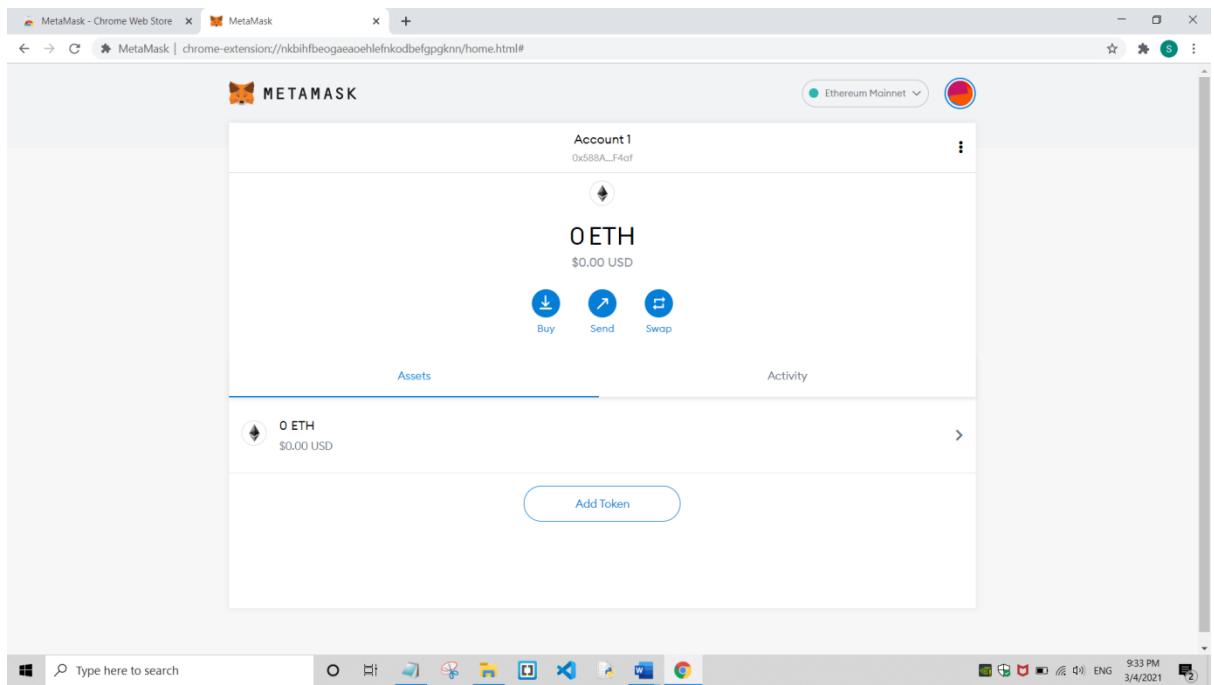
The screenshot shows the 'Congratulations' step of the MetaMask setup process. It features a small party hat icon above the word 'Congratulations'. Below the title, a message reads 'You passed the test - keep your seedphrase safe, it's your responsibility!'. Underneath this is a 'Tips on storing it safely' section with the following bullet points:

- Save a backup in multiple places.
- Never share the phrase with anyone.
- Be careful of phishing! MetaMask will never spontaneously ask for your seed phrase.
- If you need to back up your seed phrase again, you can find it in Settings > Security.
- If you ever have questions or see something fishy, email support@metamask.io.

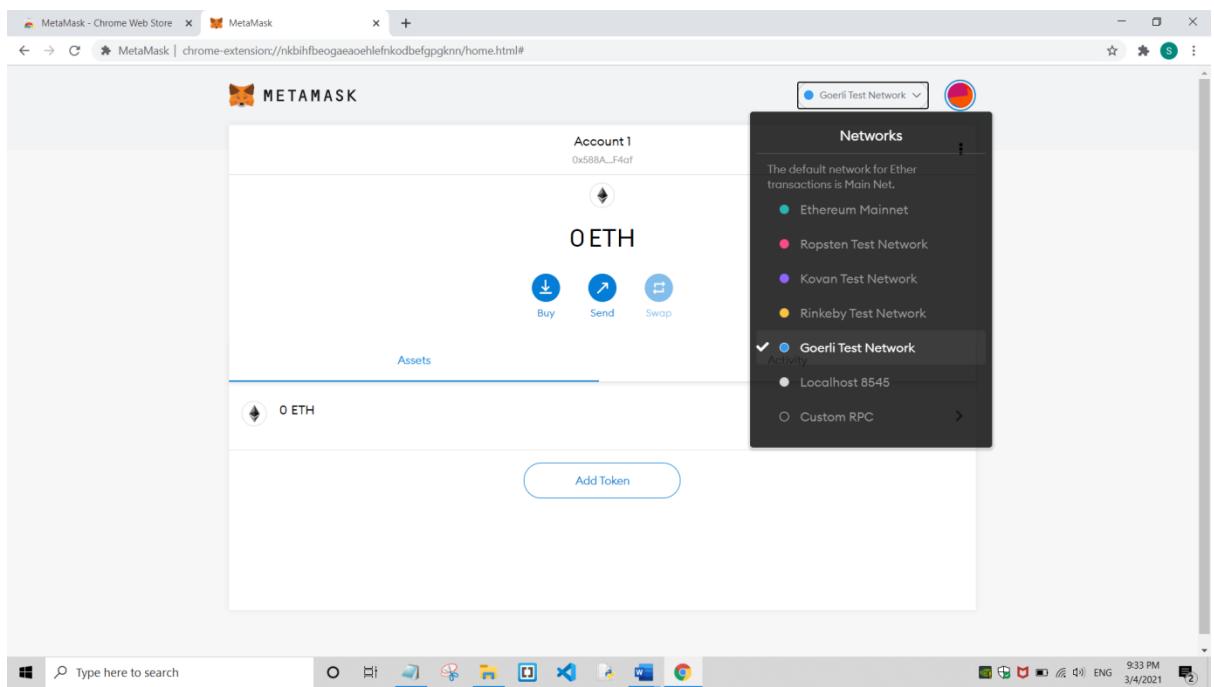
*MetaMask cannot recover your seedphrase. [Learn more.](#)

At the bottom is a blue 'All Done' button.

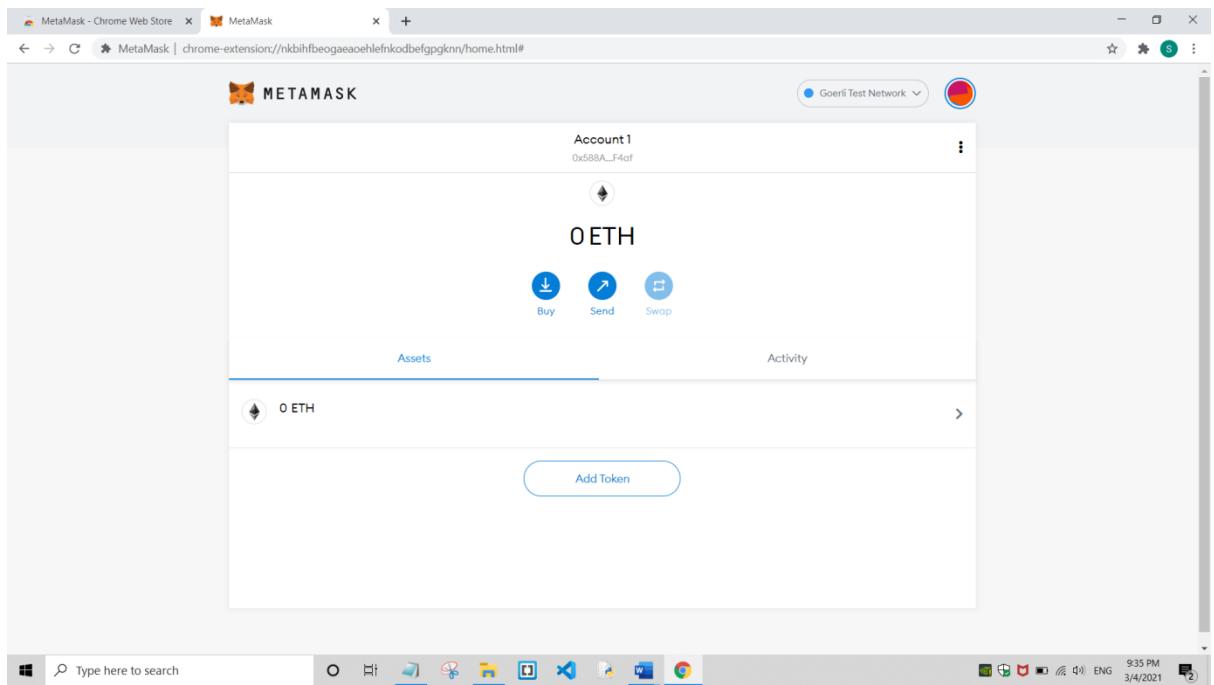




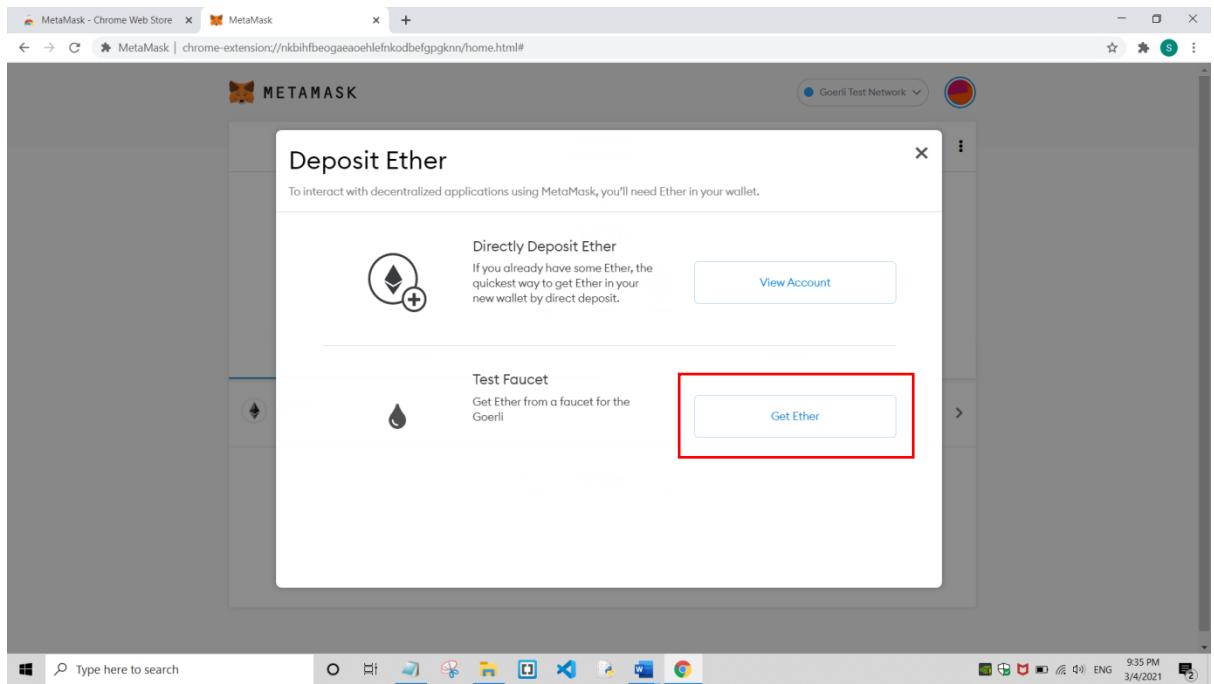
Select Goerli Test Network

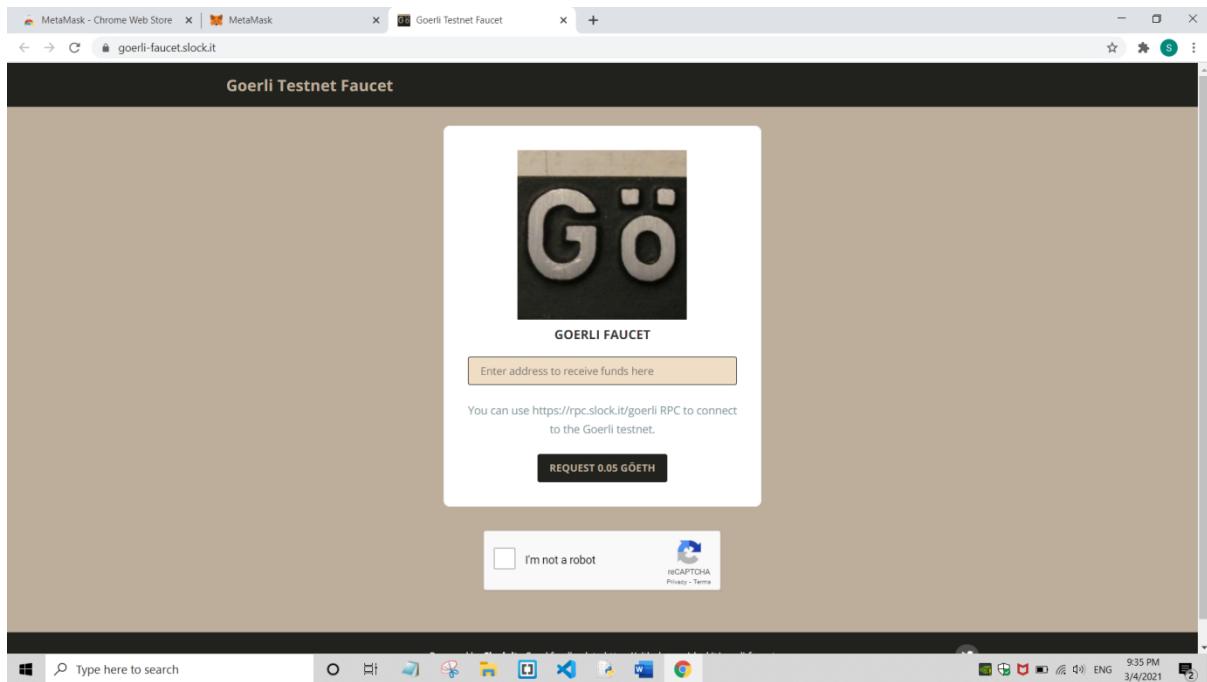


Buy Ether

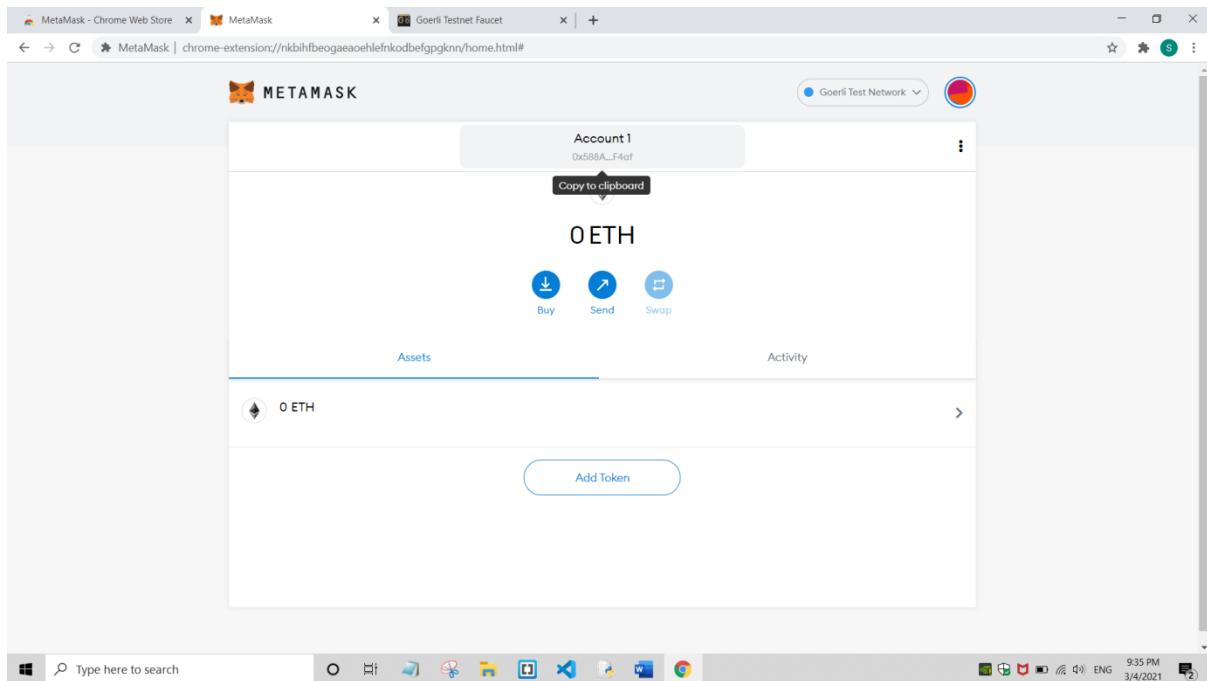


Get Ether

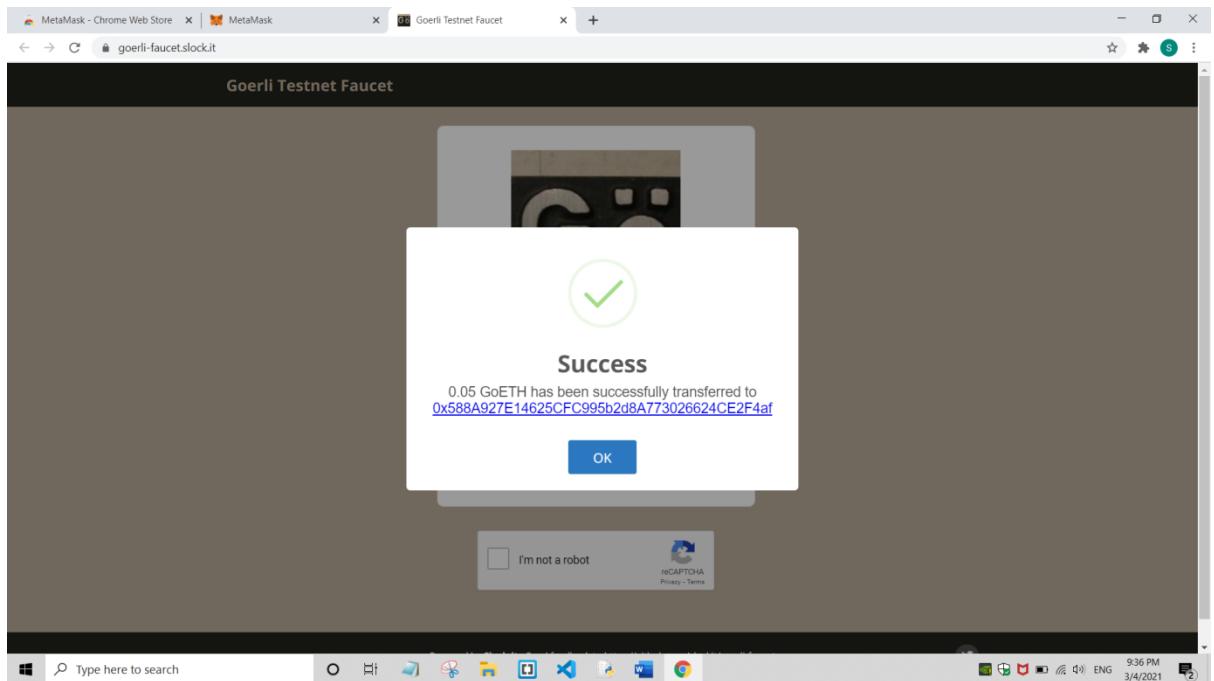
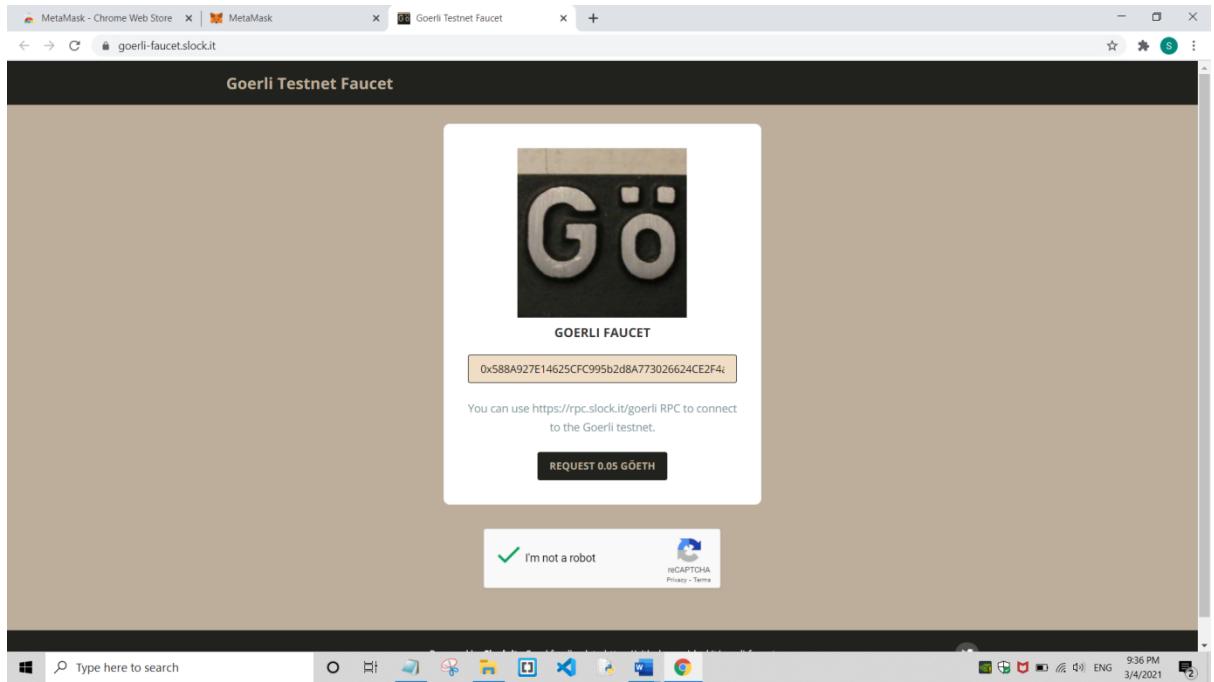




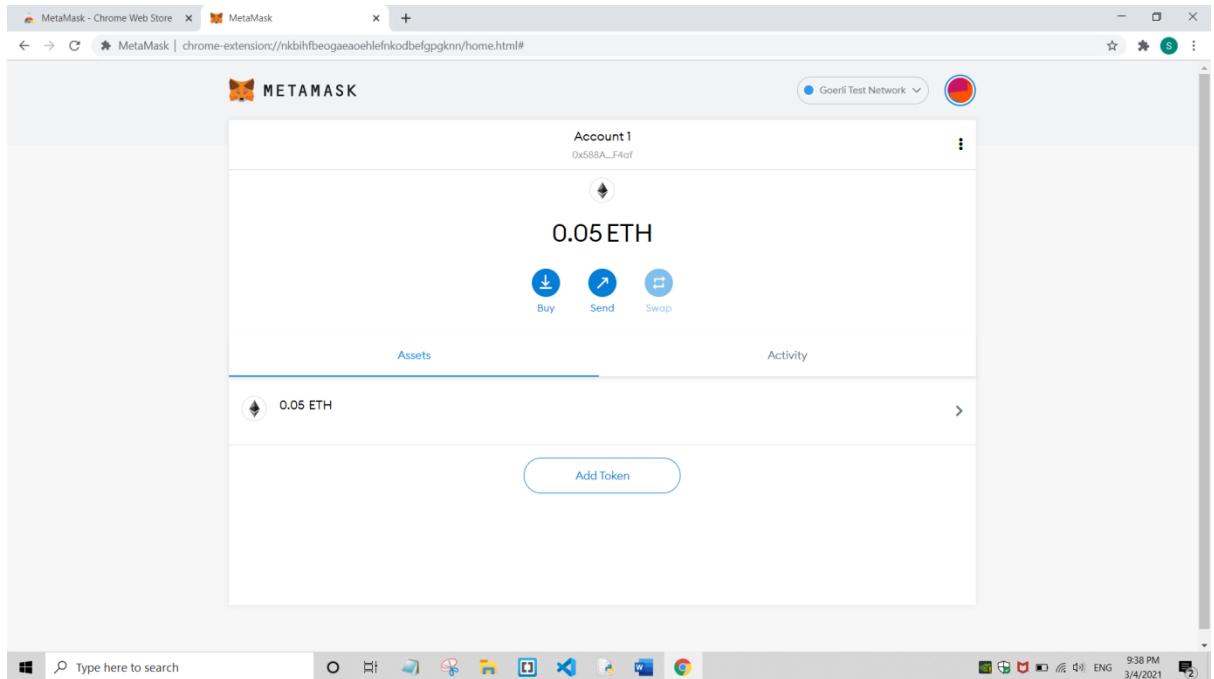
Copy the Account address



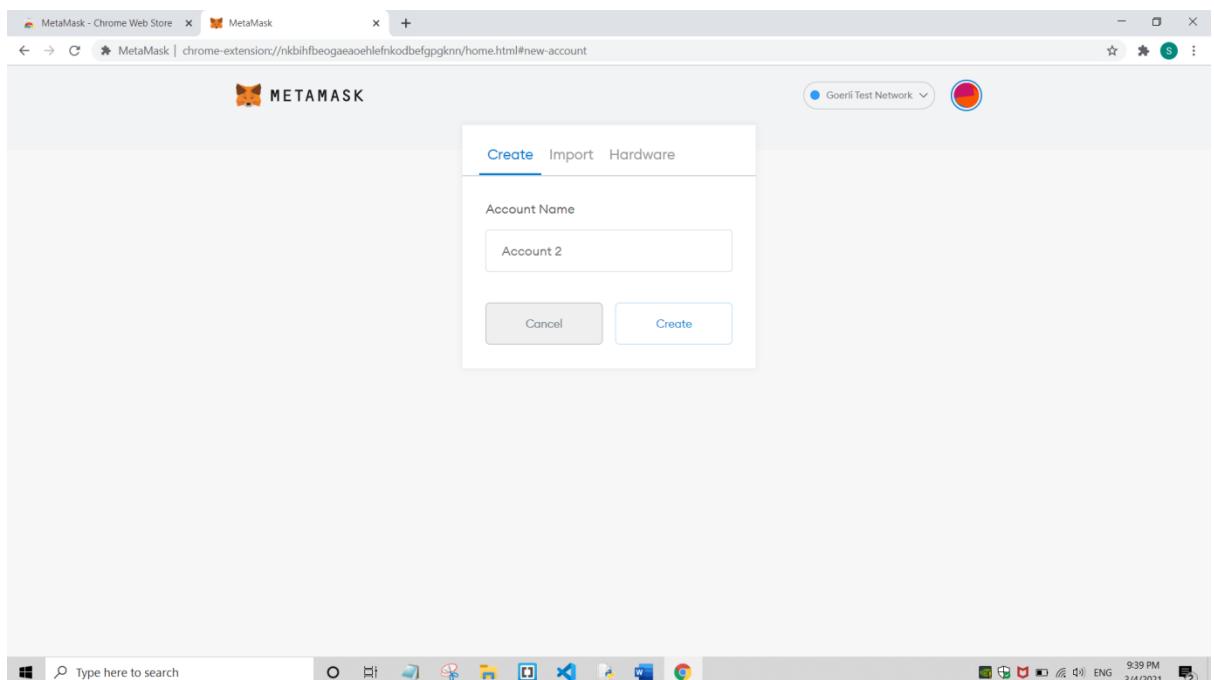
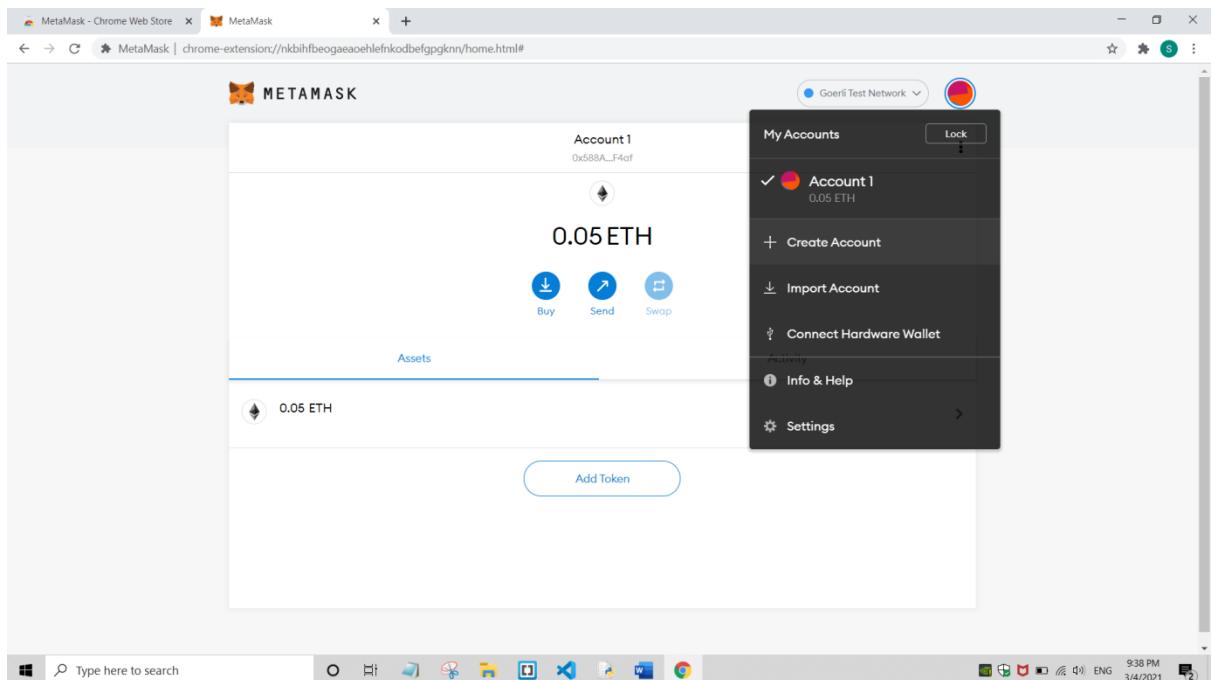
Paste the account address

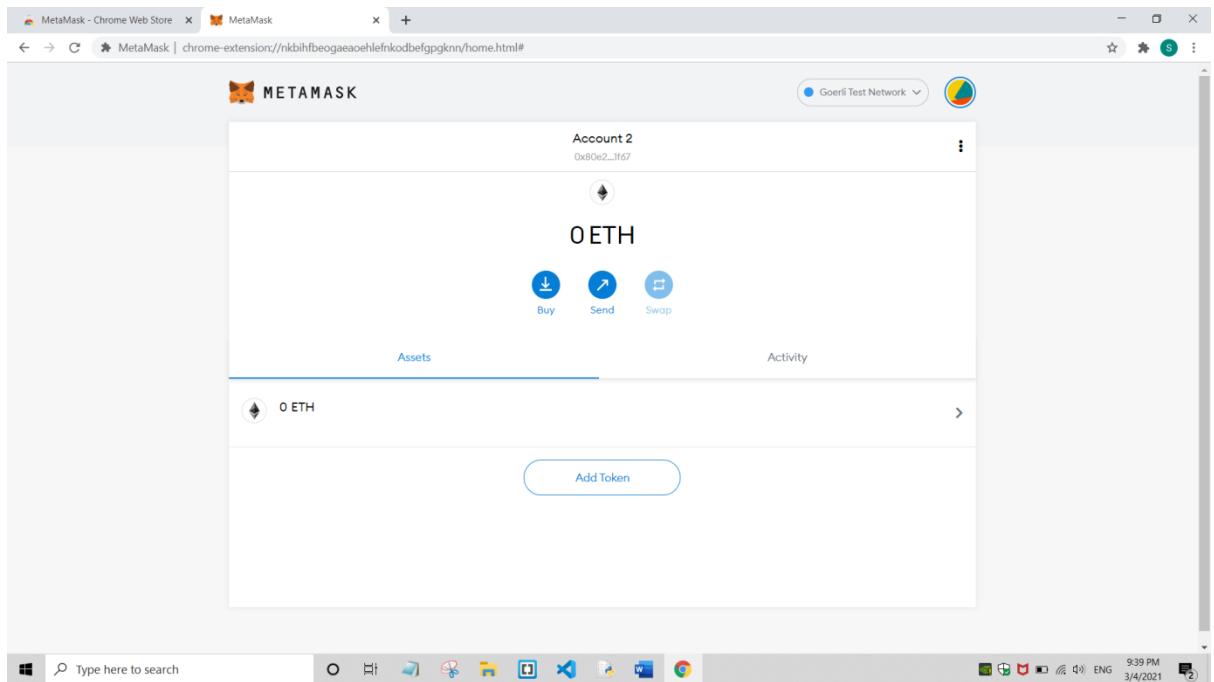


Ether added to Account

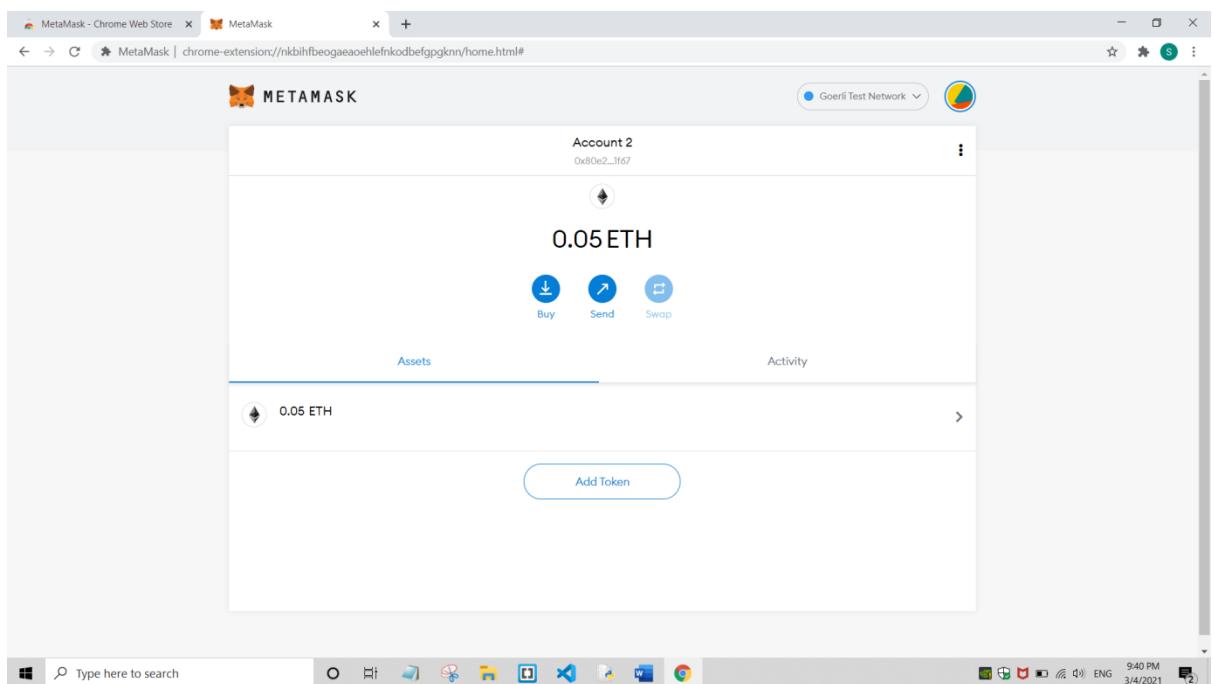


Create another account

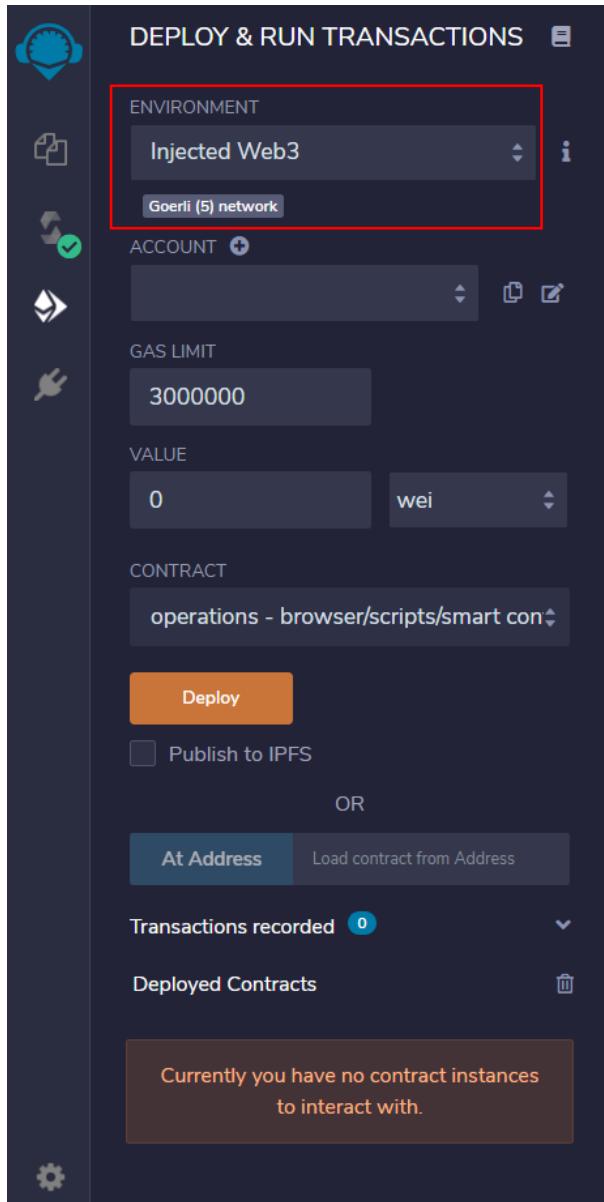




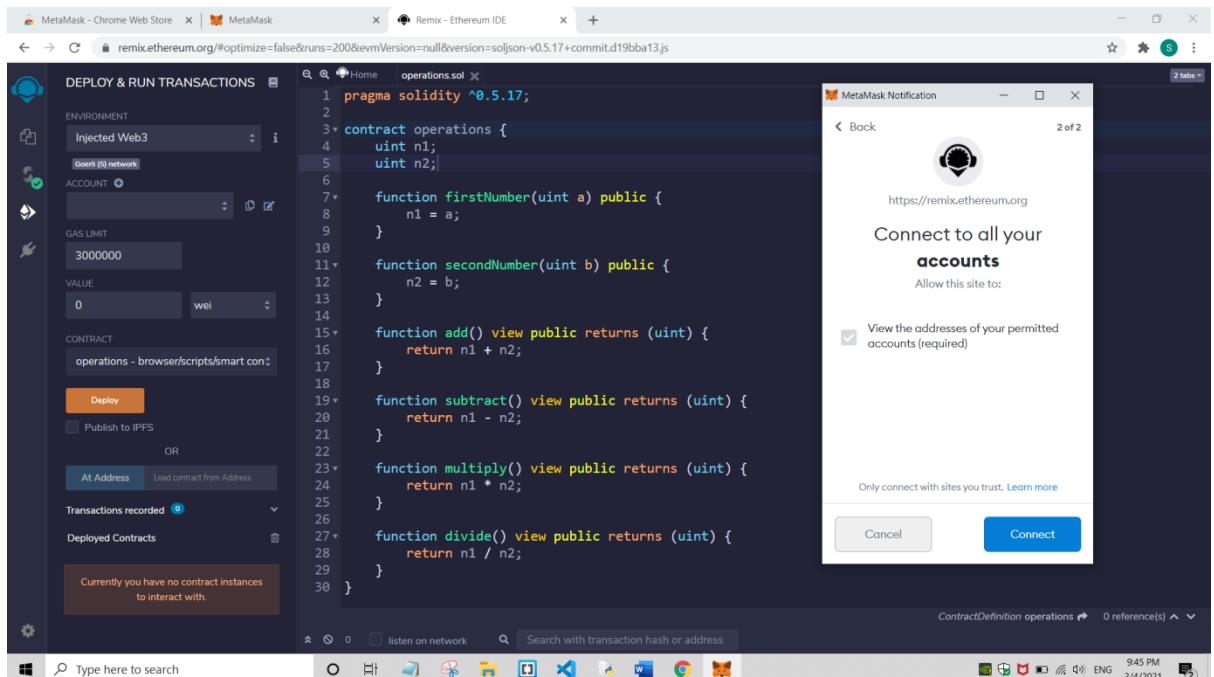
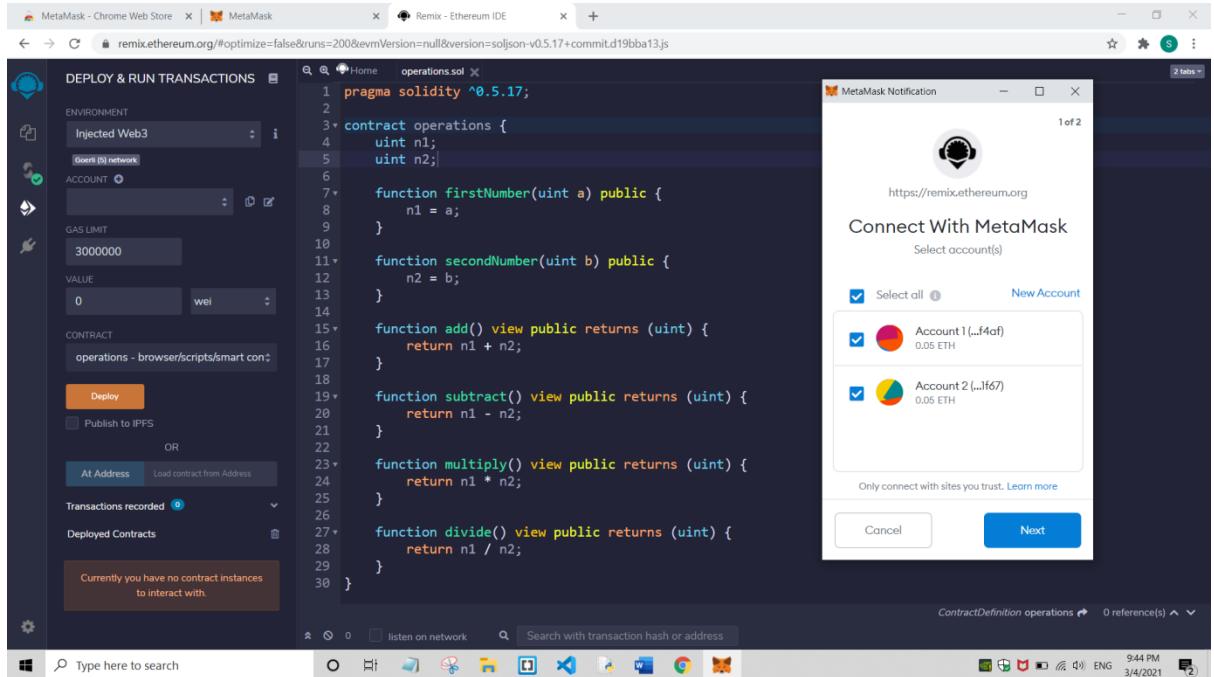
Get Ether for this account(as done for Account1)

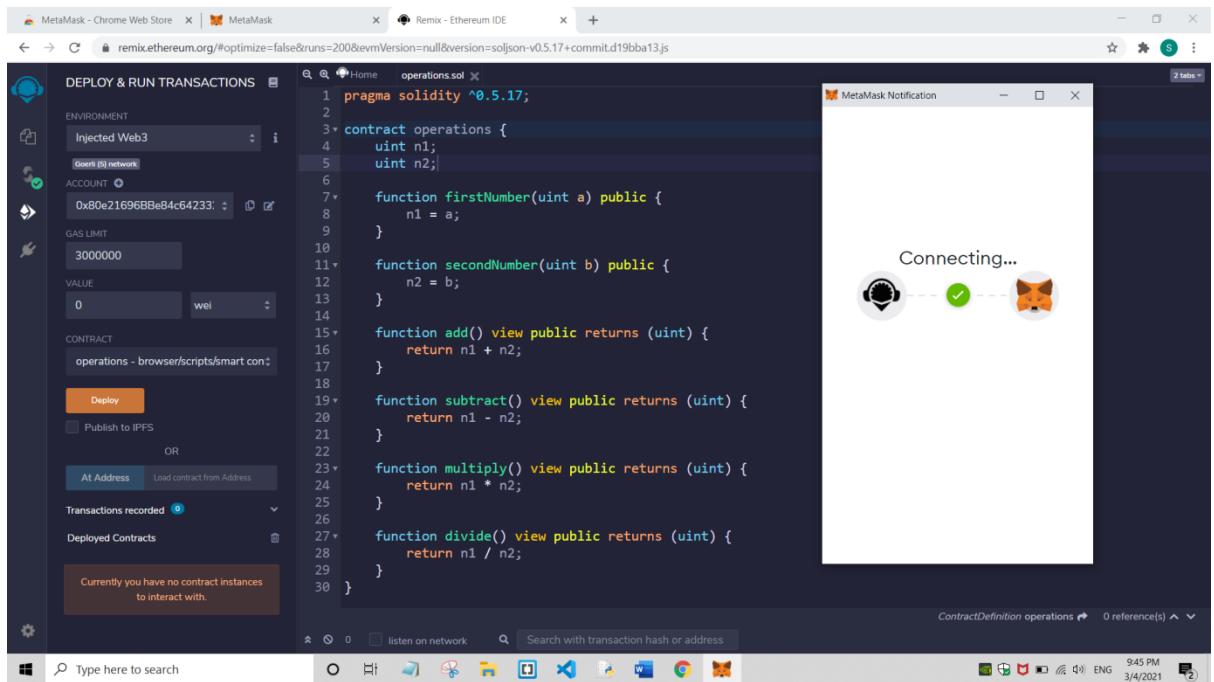


Connect to Goerli Network



Connect the Accounts





Account1

The screenshot shows the MetaMask Deploy & Run Transactions interface. On the left is a sidebar with icons for wallet management, network selection, and settings. The main area has a dark background with white text and blue highlights.

ENVIRONMENT: Injected Web3 (Goerli (5) network)

ACCOUNT: 0x588...2F4af (0.05 ether) (highlighted with a red box)

GAS LIMIT: 3000000

VALUE: 0 wei

CONTRACT: operations - browser/scripts/smart con (highlighted with a red box)

Deploy button (highlighted with a red box)

Publish to IPFS

OR

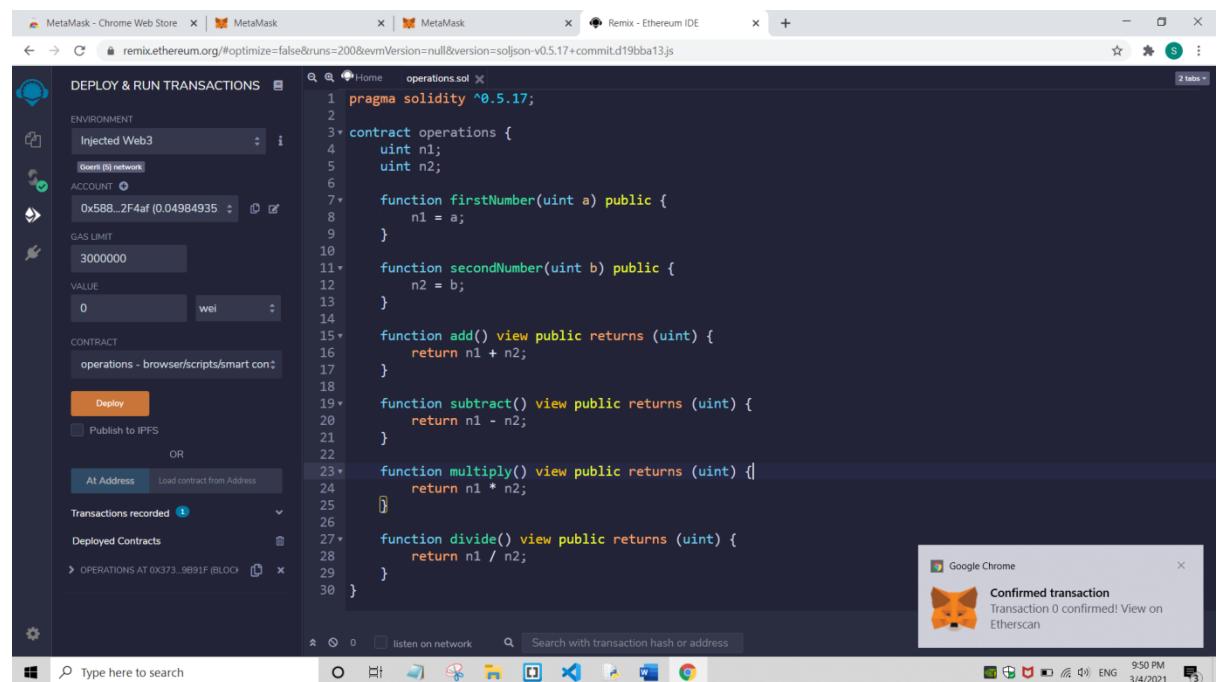
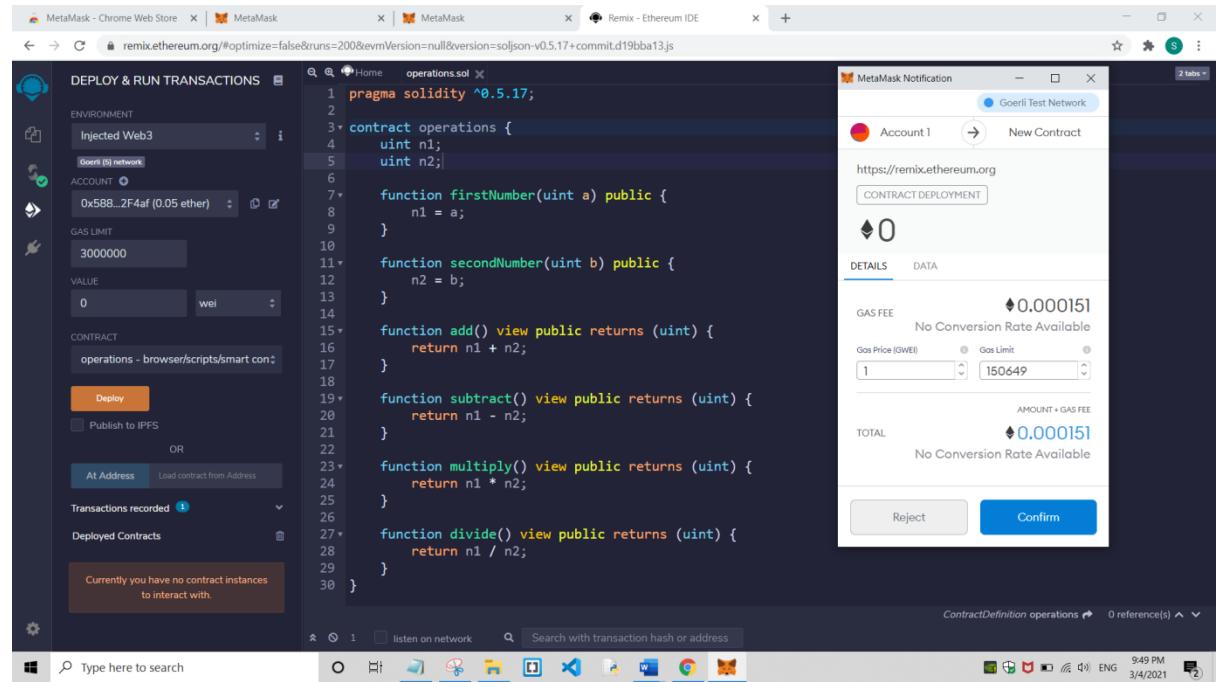
At Address (selected) / Load contract from Address

Transactions recorded: 0

Deployed Contracts: (empty)

Currently you have no contract instances to interact with.

Transaction Confirmation



CONTRACT

operations - browser/scripts/smart con▼

Deploy

Publish to IPFS

OR

At Address

Transactions recorded 1 ▾

Deployed Contracts

▼ OPERATIONS AT 0X373...9B91F (BLOCK 1)

firstNumber uint256 a ▾

secondNumber uint256 b ▾

add

divide

multiply

subtract

CONTRACT

operations - browser/scripts/smart con[▼]

Deploy

Publish to IPFS

OR

At Address

Transactions recorded **1** [▼]

Deployed Contracts [▼]

▼ OPERATIONS AT 0xDC3...A2372 (BLOC

firstNumber 10 [▼]

secondNumber uint256 b [▼]

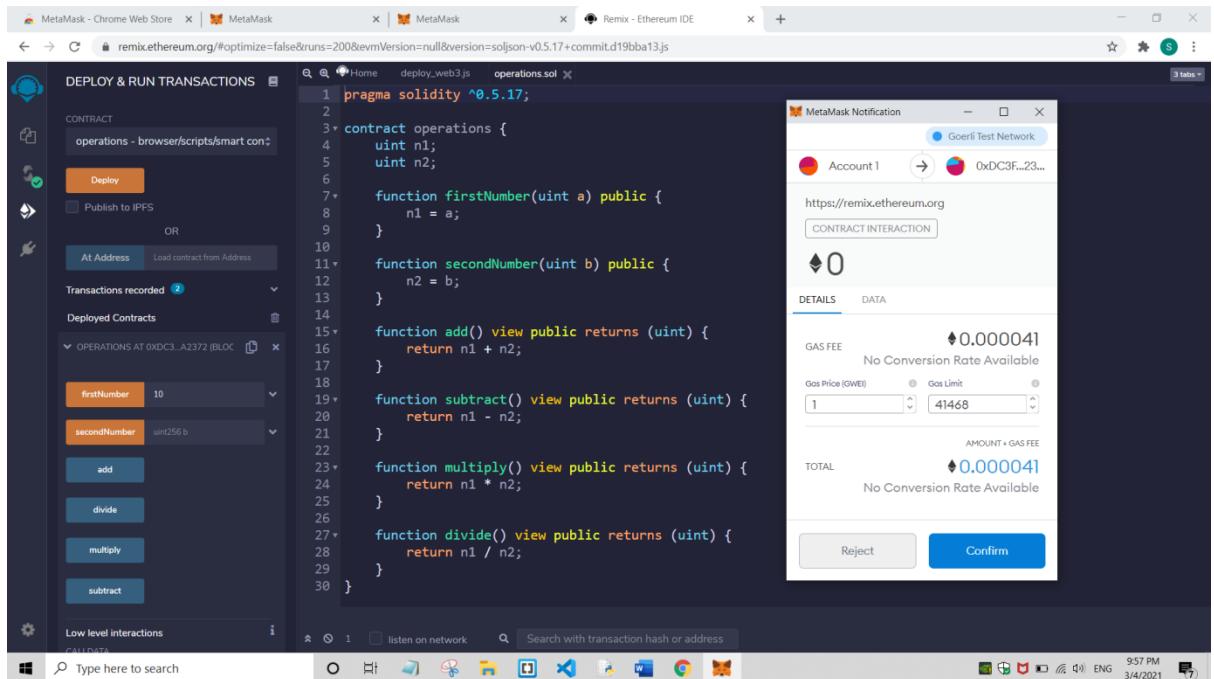
add

divide

multiply

subtract

This screenshot shows a blockchain contract interface. At the top, it displays the contract name 'operations' and provides options to 'Deploy' or 'Publish to IPFS'. Below this, there's a section for 'Transactions recorded' with a count of 1. The main area is titled 'Deployed Contracts' and lists a single contract named 'OPERATIONS AT 0xDC3...A2372 (BLOC'. This contract has two variables: 'firstNumber' set to 10 and 'secondNumber' set to 'uint256 b'. It also contains four functions: 'add', 'divide', 'multiply', and 'subtract'. A red box highlights the 'firstNumber' input field.



CONTRACT

operations - browser/scripts/smart con

Publish to IPFS

OR

Load contract from Address

Transactions recorded 2

Deployed Contracts

▼ OPERATIONS AT 0XDC3...A2372 (BLOC)

firstNumber 10

secondNumber 5

This screenshot shows a user interface for managing smart contracts. At the top, there's a header 'CONTRACT' and a dropdown menu showing 'operations - browser/scripts/smart con'. Below this is a large orange button labeled 'Deploy'. To its right is a checkbox for 'Publish to IPFS'. A horizontal line separates this from the 'OR' section. In the 'OR' section, there are two buttons: 'At Address' (highlighted in blue) and 'Load contract from Address'. Below this is a section titled 'Transactions recorded' with a count of '2'. Under 'Deployed Contracts', a list item is expanded to show details for 'OPERATIONS AT 0XDC3...A2372 (BLOC)'. This expanded view includes fields for 'firstNumber' (set to 10) and 'secondNumber' (set to 5), both of which are highlighted with a red border. Below these fields are four blue buttons labeled 'add', 'divide', 'multiply', and 'subtract'.

The screenshot shows the Ethereum IDE interface with several tabs open:

- MetaMask - Chrome Web Store
- MetaMask
- Remix - Ethereum IDE
- remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljson-v0.5.17+commit.d19bba13.js

The main area displays the Solidity code for a smart contract named `operations`:

```
pragma solidity ^0.5.17;

contract operations {
    uint n1;
    uint n2;

    function firstNumber(uint a) public {
        n1 = a;
    }

    function secondNumber(uint b) public {
        n2 = b;
    }

    function add() view public returns (uint) {
        return n1 + n2;
    }

    function subtract() view public returns (uint) {
        return n1 - n2;
    }

    function multiply() view public returns (uint) {
        return n1 * n2;
    }

    function divide() view public returns (uint) {
        return n1 / n2;
    }
}
```

The `Deploy & Run Transactions` sidebar shows the contract has been deployed at address `0xDC3...A2372`. The `Transactions recorded` section lists the following interactions:

- `firstNumber`: 10
- `secondNumber`: 5
- `add`
- `divide`
- `multiply`
- `subtract`

A `MetaMask Notification` window is open, showing the Goerli Test Network and Account 1 (`0xDC3F...23...`). It displays the total gas fee as `0.000042` GWEI and the total amount as `0.000042`. The `Gas Price (GWEI)` is set to 1 and the `Gas Limit` is set to 41512. Buttons for `Reject` and `Confirm` are present.

The screenshot shows the `Deployed Contracts` list in the Ethereum IDE:

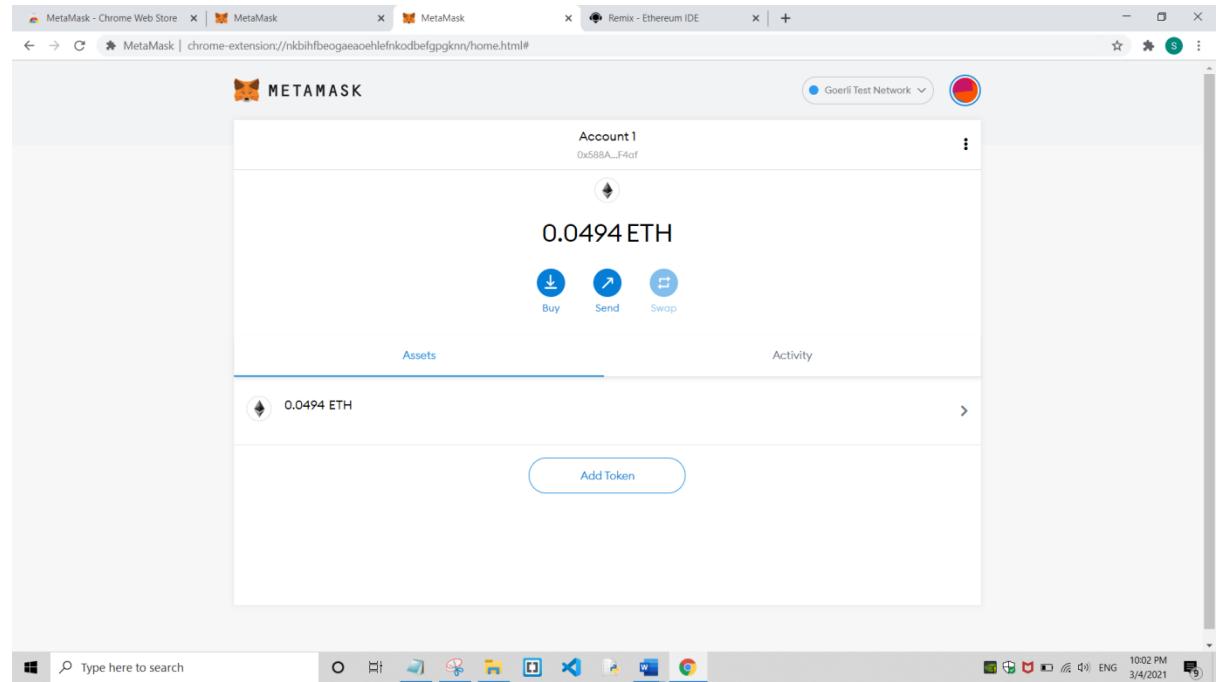
- `OPERATIONS AT 0xDC3...A2372 (BLOC)`

Under this list item, the following values are displayed:

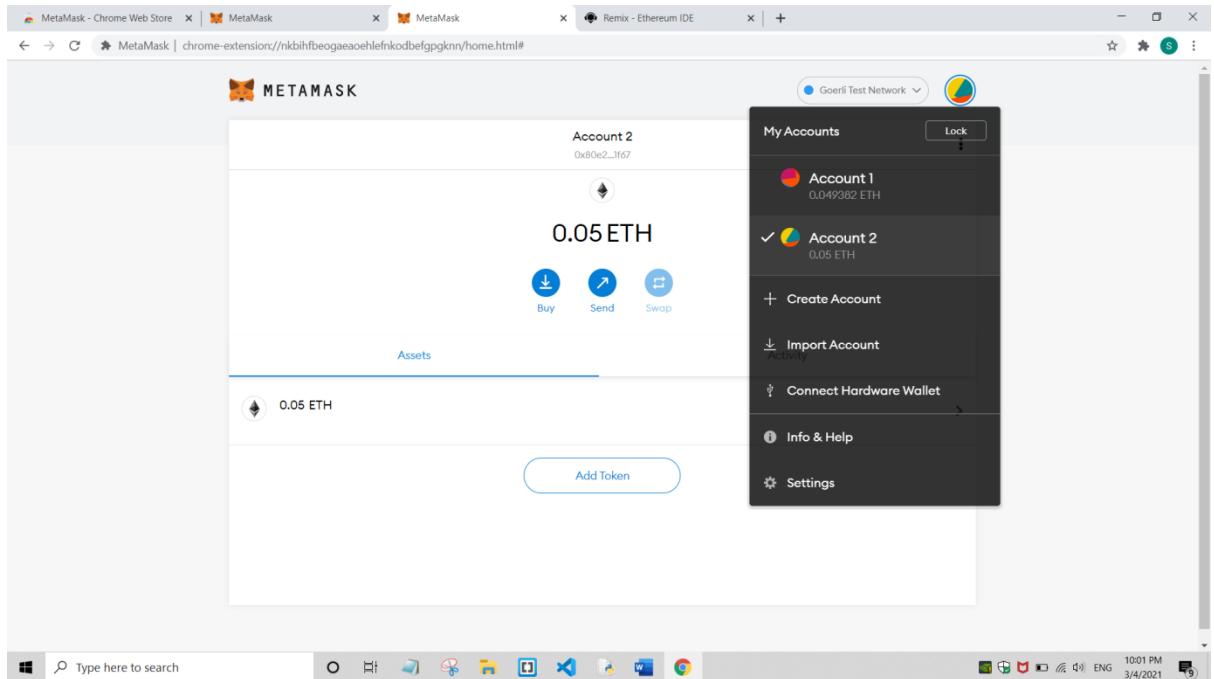
- `firstNumber`: 10
- `secondNumber`: 5
- `add`: 0: uint256: 10
- `divide`: 0: uint256: 2
- `multiply`: 0: uint256: 50
- `subtract`: 0: uint256: 5

A red box highlights the `add`, `divide`, `multiply`, and `subtract` buttons.

Ethers are consumed



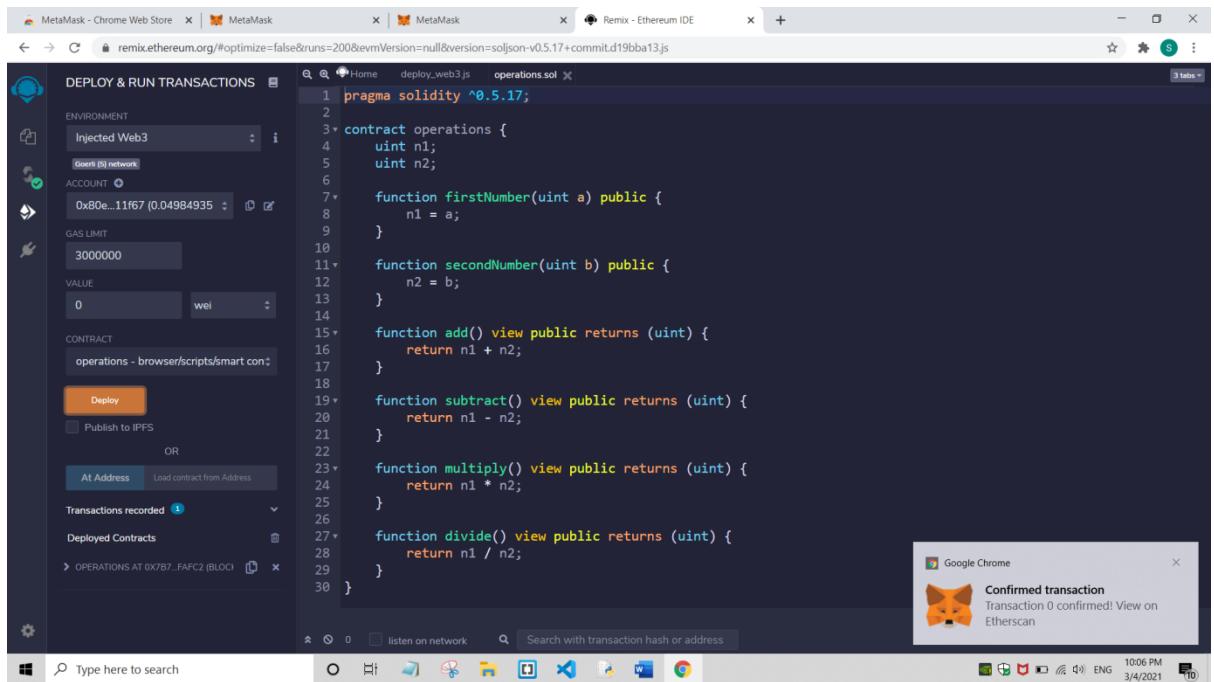
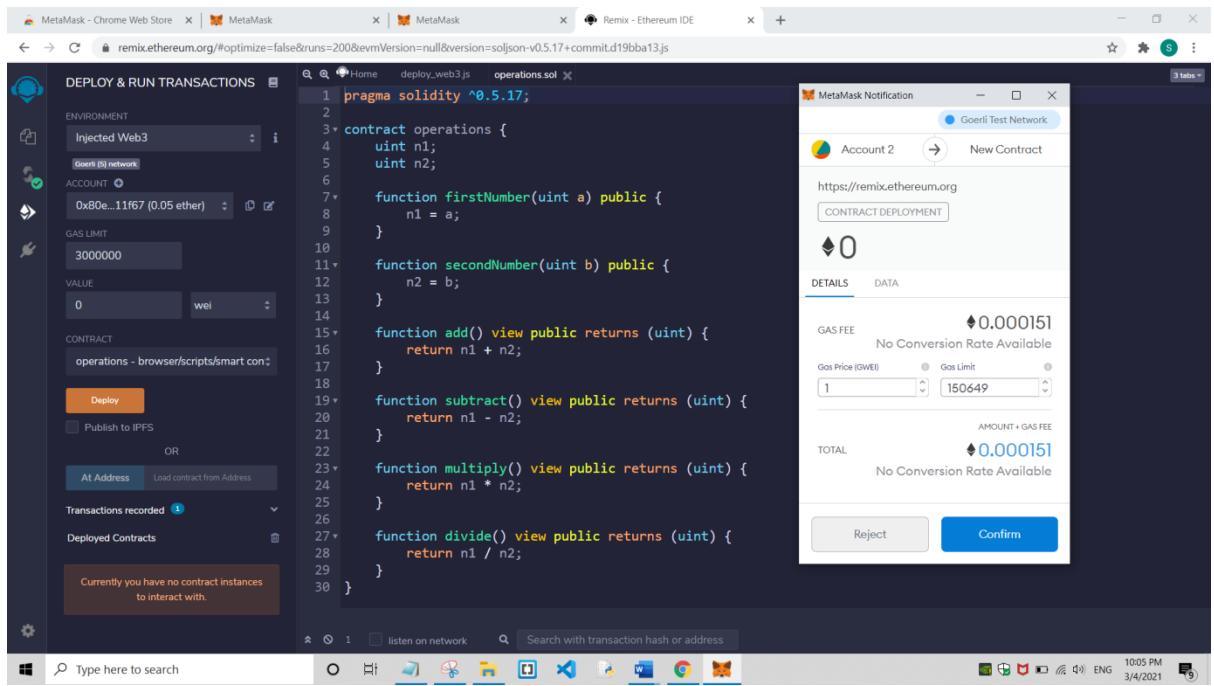
Switch to Account2



Account2

The screenshot shows the "DEPLOY & RUN TRANSACTIONS" section of the MetaMask interface. On the left, there's a sidebar with icons for wallet, accounts, contracts, and settings. The main area has the following fields:

- ENVIRONMENT:** A dropdown menu set to "Injected Web3" (highlighted with a red box), with "Goerli (5) network" listed below it.
- ACCOUNT:** Shows the address "0x80e...11f67 (0.05 ether)" and a dropdown arrow.
- GAS LIMIT:** Set to "3000000".
- VALUE:** Set to "0" wei.
- CONTRACT:** A dropdown menu set to "operations - browser/scripts/smart con" (highlighted with a red box).
- Deploy:** An orange button below the contract selection.
- Publish to IPFS:** A checkbox followed by a link.
- OR:** A section with "At Address" and "Load contract from Address" buttons.
- Transactions recorded:** Shows "0".
- Deployed Contracts:** A list with a delete icon.
- Message:** "Currently you have no contract instances to interact with."



CONTRACT

operations - browser/scripts/smart con

Deploy

Publish to IPFS

OR

At Address Load contract from Address

Transactions recorded 1

Deployed Contracts Delete

▼ OPERATIONS AT 0X7B7...FAFC2 (BLOC Copy X)

firstNumber 50

secondNumber uint256 b

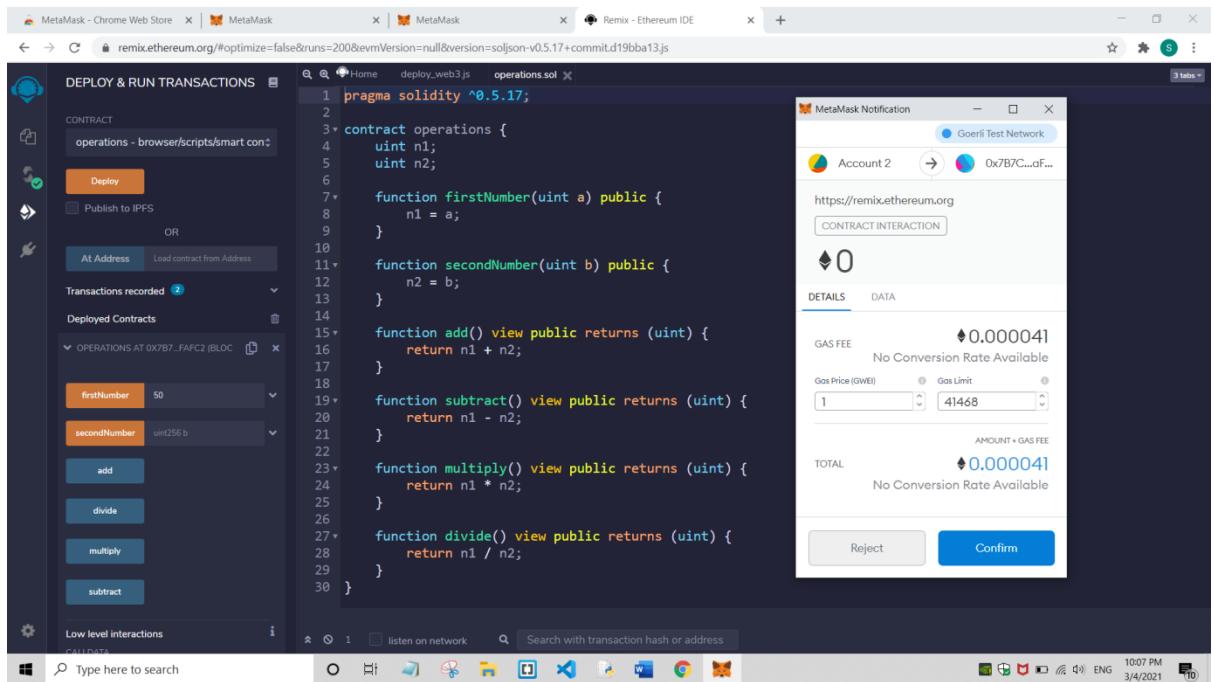
add

divide

multiply

subtract

This screenshot shows a user interface for managing a deployed smart contract. At the top, there's a header 'CONTRACT' followed by the contract name 'operations - browser/scripts/smart con'. Below this is a large orange button labeled 'Deploy'. There's also a checkbox for 'Publish to IPFS'. The word 'OR' is centered below the deployment options. A blue button labeled 'At Address' is highlighted, while another blue button 'Load contract from Address' is shown next to it. Below these buttons, a section titled 'Transactions recorded' shows a single transaction with the number '1'. Under the heading 'Deployed Contracts', there's a list item for 'OPERATIONS AT 0X7B7...FAFC2 (BLOC)' which includes a copy icon and a delete icon. The main content area displays two variables: 'firstNumber' set to '50' and 'secondNumber' set to 'uint256 b'. Below these variables are four blue buttons labeled 'add', 'divide', 'multiply', and 'subtract'.



CONTRACT

operations - browser/scripts/smart con[▼]

Deploy

Publish to IPFS

OR

At Address Load contract from Address

Transactions recorded **2** [▼]

Deployed Contracts [▼] X

▼ OPERATIONS AT 0X7B7...FAFC2 (BLOC Copy X

firstNumber 50 ▼

secondNumber 10 ▼

add

divide

multiply

subtract

This screenshot shows a web-based interface for managing a smart contract. At the top, it displays the contract name 'operations - browser/scripts/smart con'. Below this are buttons for 'Deploy' and 'Publish to IPFS'. An 'OR' button allows loading the contract from an address. A section titled 'Transactions recorded' shows two entries. The main area is titled 'Deployed Contracts' and lists a single contract at address '0X7B7...FAFC2 (BLOC)'. This contract has two input fields: 'firstNumber' set to 50 and 'secondNumber' set to 10. Below these fields are four buttons: 'add', 'divide', 'multiply', and 'subtract'. The 'secondNumber' field is highlighted with a red border.

The screenshot shows a browser window with three tabs: "MetaMask - Chrome Web Store", "MetaMask", and "Remix - Ethereum IDE". The "Remix - Ethereum IDE" tab is active, displaying the Solidity code for a smart contract named "operations". The code defines a contract with four public functions: firstNumber, secondNumber, add, subtract, multiply, and divide. The "firstNumber" and "secondNumber" variables are set to 50 and 10 respectively. The "add" function returns 60, "subtract" returns 5, "multiply" returns 500, and "divide" returns 5.

```
pragma solidity ^0.5.17;

contract operations {
    uint n1;
    uint n2;

    function firstNumber(uint a) public {
        n1 = a;
    }

    function secondNumber(uint b) public {
        n2 = b;
    }

    function add() view public returns (uint) {
        return n1 + n2;
    }

    function subtract() view public returns (uint) {
        return n1 - n2;
    }

    function multiply() view public returns (uint) {
        return n1 * n2;
    }

    function divide() view public returns (uint) {
        return n1 / n2;
    }
}
```

A "MetaMask Notification" window is overlaid on the Remix interface, prompting the user to confirm a transaction. The notification shows the account being used (Account 2, 0x7B7C...aF...) and the transaction details: GAS FEE 0.000042 GWEI, Gas Price (GWEI) 1, Gas Limit 41512, and TOTAL 0.000042 GWEI. The "Confirm" button is highlighted.

The screenshot shows the "Deployed Contracts" section of the Ethereum IDE. It displays the deployed contract "OPERATIONS AT 0X7B7...FAFC2 (BLOC)" with its state variables "firstNumber" (50) and "secondNumber" (10). Below these, four function buttons are listed: "add", "divide", "multiply", and "subtract". A red box highlights the "add", "divide", "multiply", and "subtract" buttons, indicating the area where the user can interact with the deployed contract to perform arithmetic operations.

