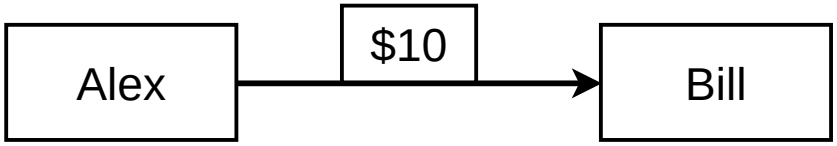
Requirements of a system to record and transfer value

Problem	Theoretical Solution	Ethereum Solution
Need a currency that represents value	Create a scarce currency that exists only in the system.	"Ether" tokens.
Need to represent people and the amount of money they own	Every person participating in the system should have a unique account identifier and a balance tied to that identifier	Accounts system. Anyone can freely create an account with a 20 byte address tied to it that identifies the account.
Need a way to represent transfers of money between users	To transfer money a record of a 'transaction' must be created.	Transactions. Record the source account, target account, and amount to transfer.
Need to keep a record of all transactions	All transactions should be saved to some database.	????

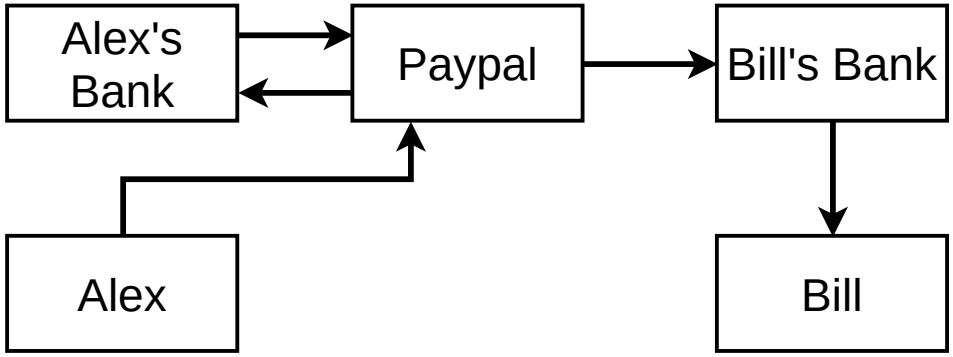
What's the *best way* to understand the blockchain?


WHAT IS THE BLOCKCHAIN?
HOW DOES THE BLOCKCHAIN WORK?
WHERE DOES THE BLOCKCHAIN EXIST
WHAT IS THE PURPOSE OF THE BLOCKCHAIN?


THE BLOCKCHAIN?
imgflip.com

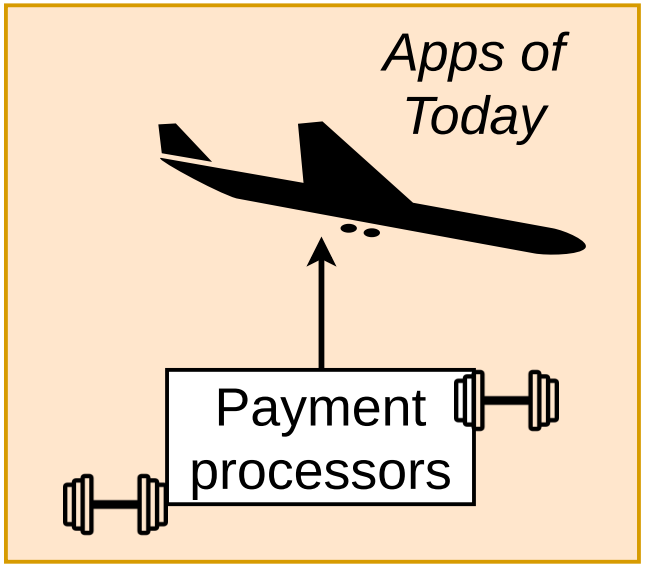


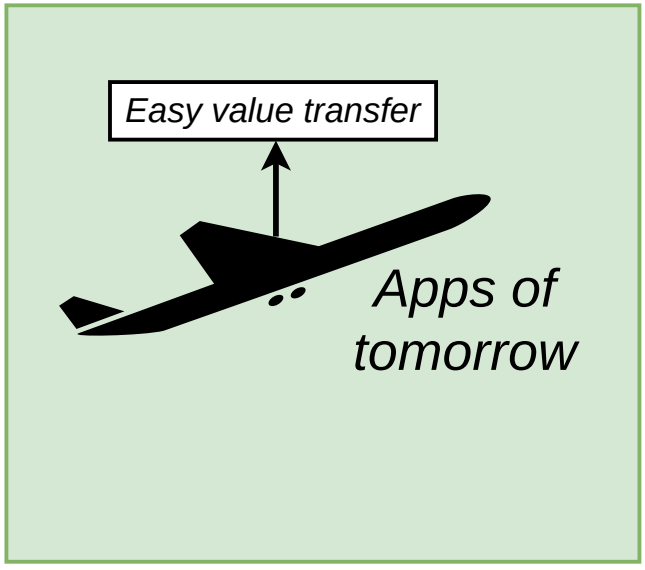
How can Alex send Bill
$10 digitally?

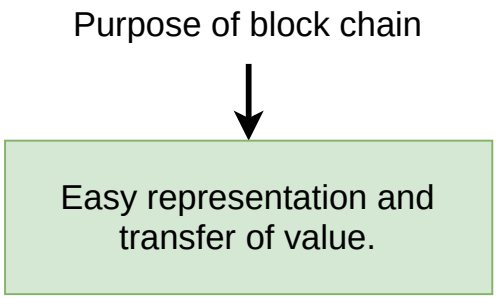

Alex's Bank
Paypal
Bill's Bank
Alex
Bill

Company	Company Valuation
Paypal	$88.4 Billion
Square	$8.96 Billion
Stripe	$9 Billion
Facebook Messenger	???
Google Wallet	???

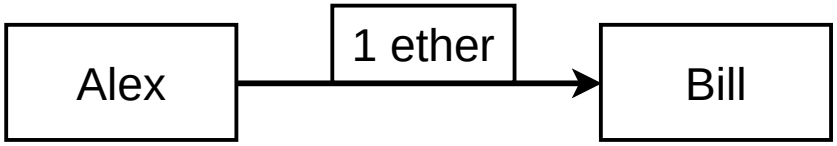
Takeaway → **It costs $100 Billion+ of complexity to move money digitally**


Apps of Today
Payment processors


Easy value transfer
Apps of tomorrow


Purpose of block chain
Easy representation and transfer of value.

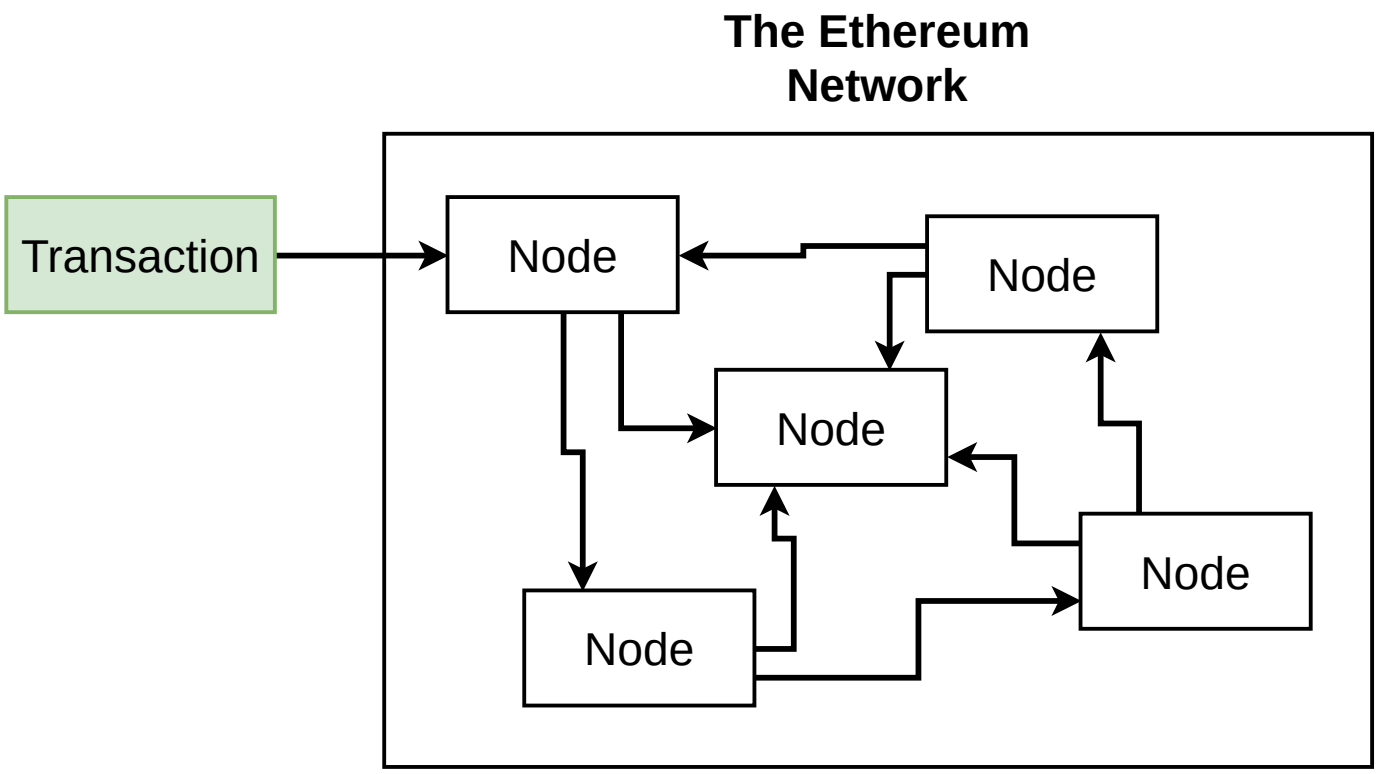
What is the blockchain?	Database of transactions (sending and receiving money) between different users.
How does it work?	Complicatedly.

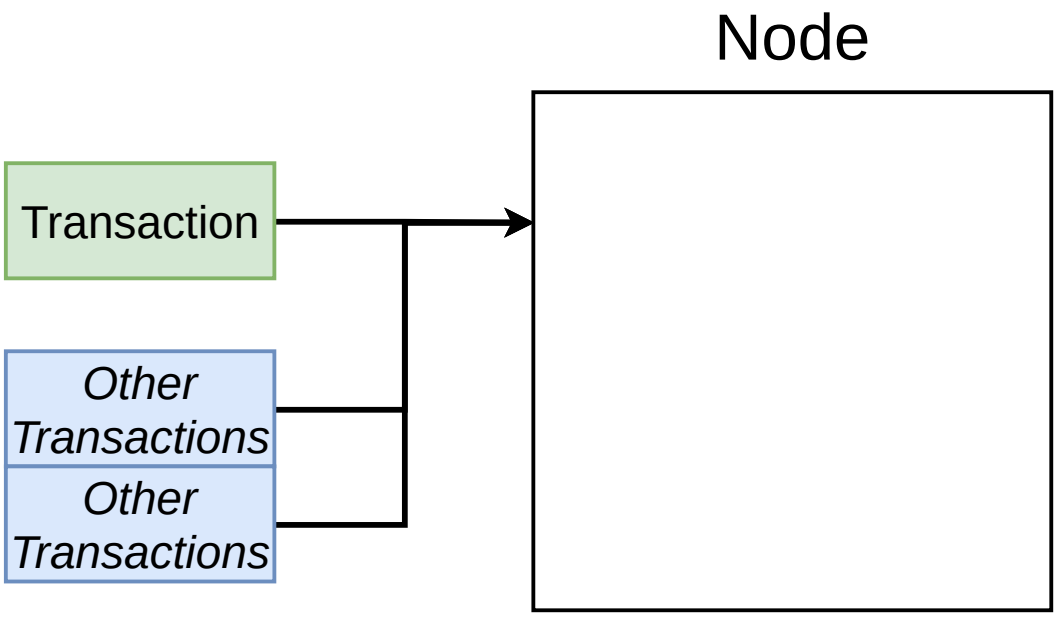


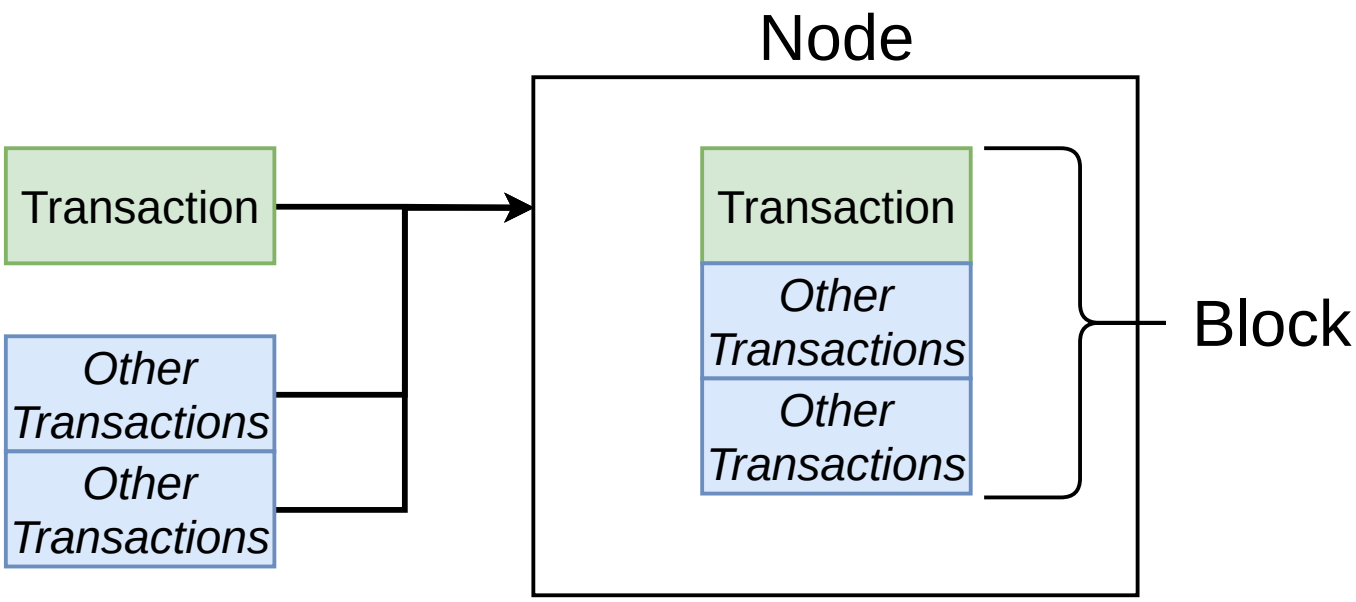
How can Alex send Bill
1 ether?

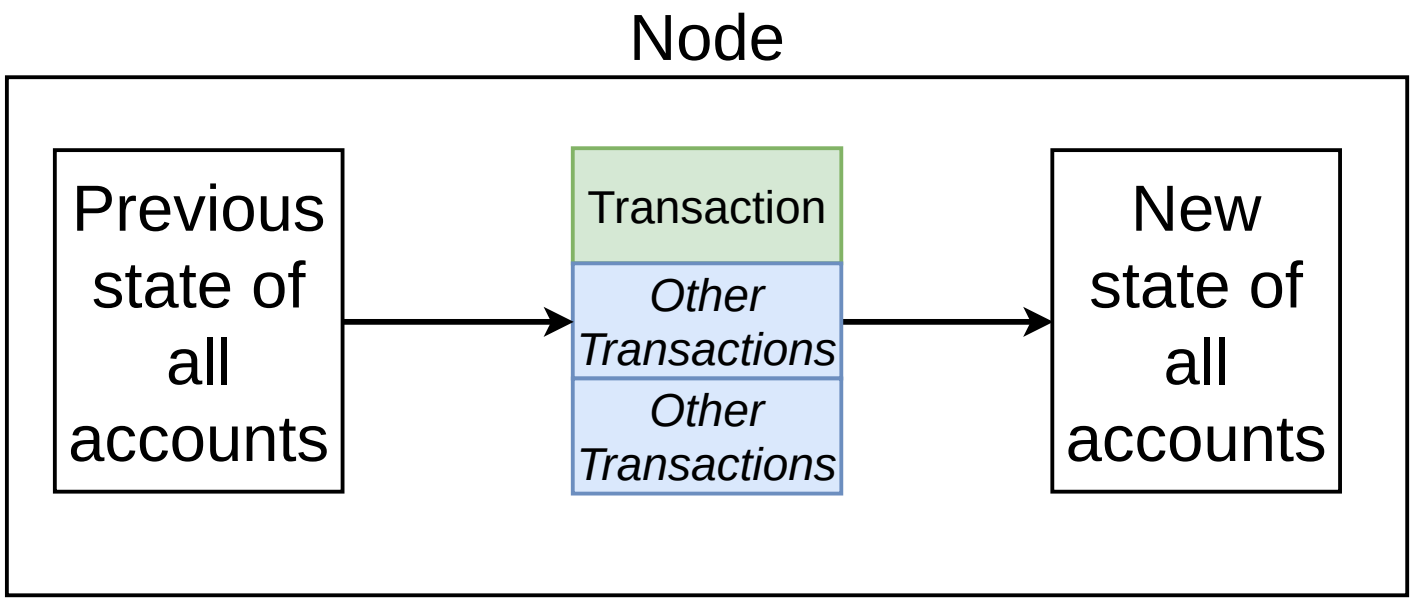
	Account Address / Public Key	Private Key
Alex	0x383723a06541a4f145439f6fcceed2f5e419dc8c	b1c6f656b2661bb15e00d1e7c41d2ce50a299c0963ab45b479e28422610561a3
Bill	0x269e2827ad96de057dde85e69303623eb9104591	746435ff7802460dc5d422960e789dc076601ffdf89ff499c31aad1c65fe278b

Transaction Object	
Property	Purpose
from	Address of person to send ether
to	Address of person to receive ether
value	Amount of ether to send
other fields	*we will discuss later*


The Ethereum Network
Transaction
Node
Node
Node
Node
Node


Node
Transaction
Other Transactions
Other Transactions


Node
Transaction
Other Transactions
Other Transactions
Transaction
Other Transactions
Other Transactions
Block


Node
Previous state of all accounts
Transaction
Other Transactions
Other Transactions
New state of all accounts

Account State	
Address	Amount
Alex	5
Bill	0

Transactions		
From	To	Amount
Alex	Bill	1 Ether
0xa1	*0x84*	*2 Ether*
0x48	*0xb0*	*.01 Ether*

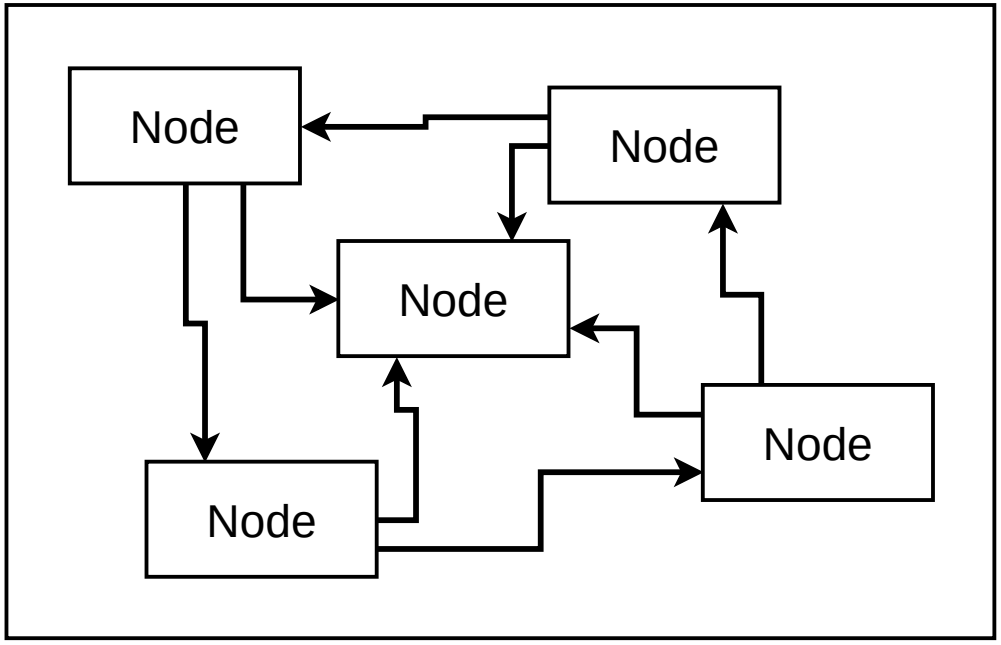
Account State	
Address	Amount
Alex	4
Bill	1

Fro	
Ale	
0xa	
0x4	

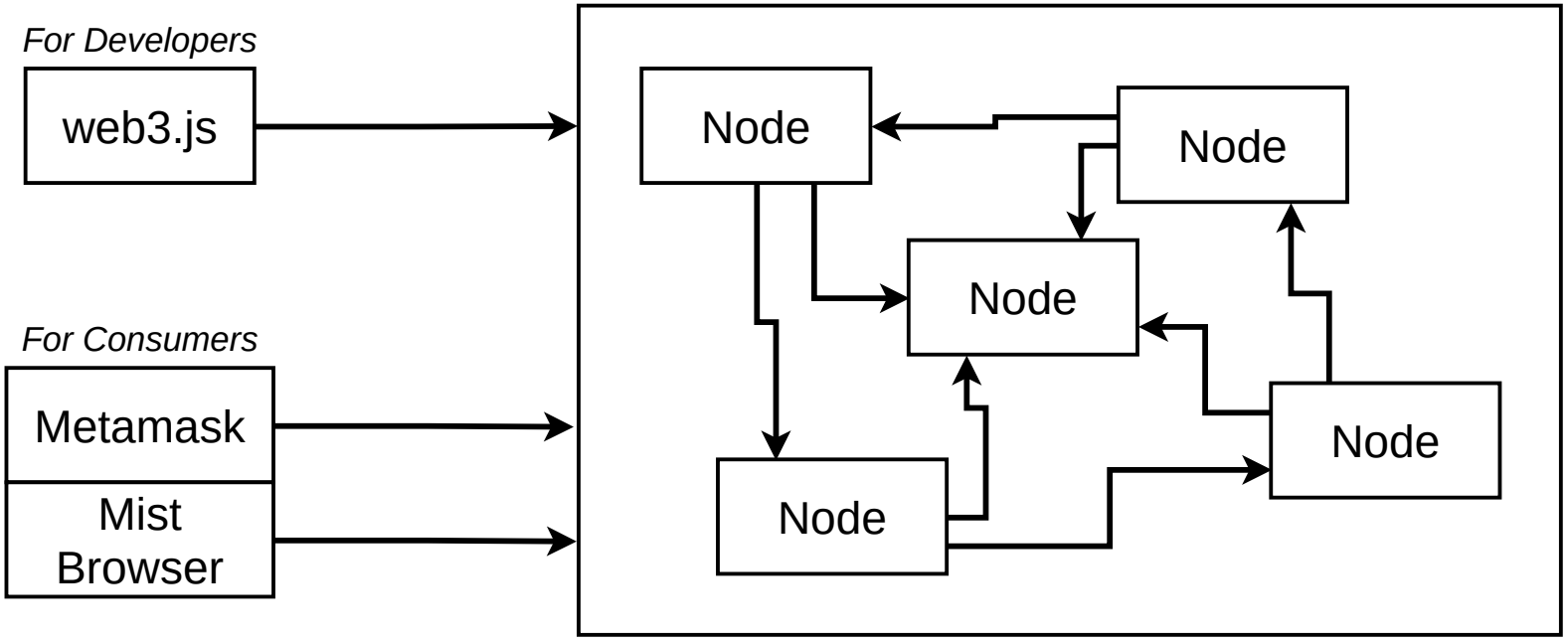
Transactions		
om	To	Amount
ex	Bill	1 Ether
a1	*0x84*	*2 Ether*
48	*0xb0*	*.01 Ether*

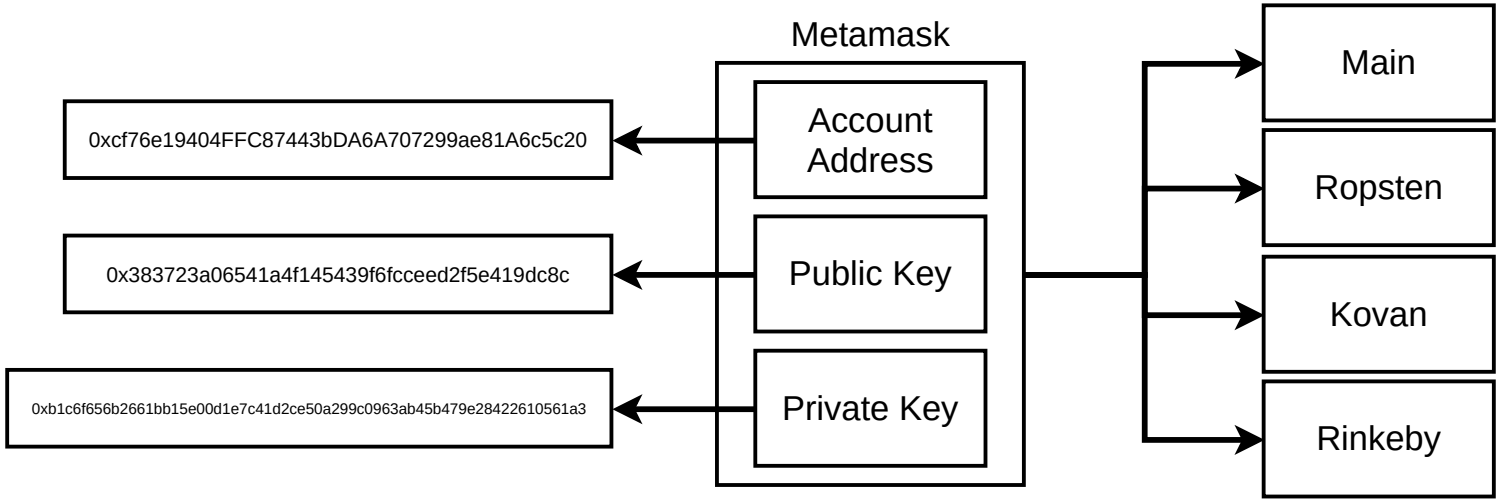
Account State	
Address	Amount
Alex	3
Bill	2

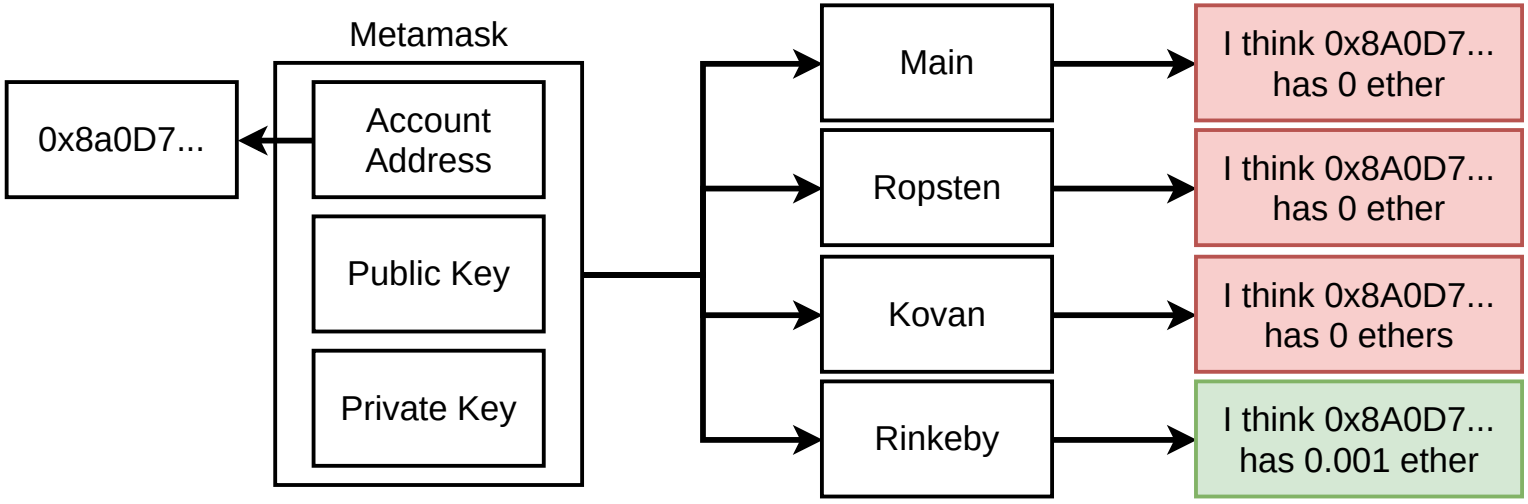
An Ethereum Network

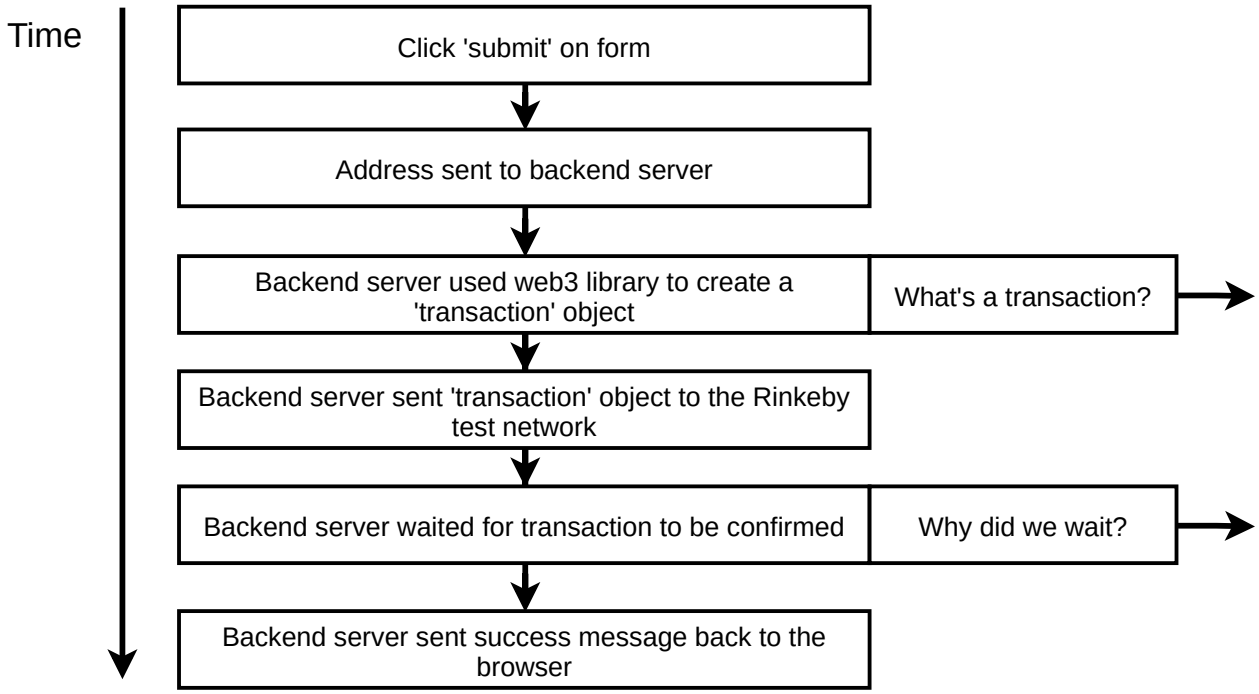


Ethereum networks are used to transfer money and store data
There are many different Ethereum networks.
Networks are formed by one or more nodes.
Each node is a machine running an ethereum client.
Anyone can run a node.
Each node can contain a full copy of the blockchain
The 'blockchain' is a database that stores a record of every transaction that has ever taken place


For Developers
web3.js
For Consumers
Metamask
Mist Browser
Node
Node
Node
Node
Node

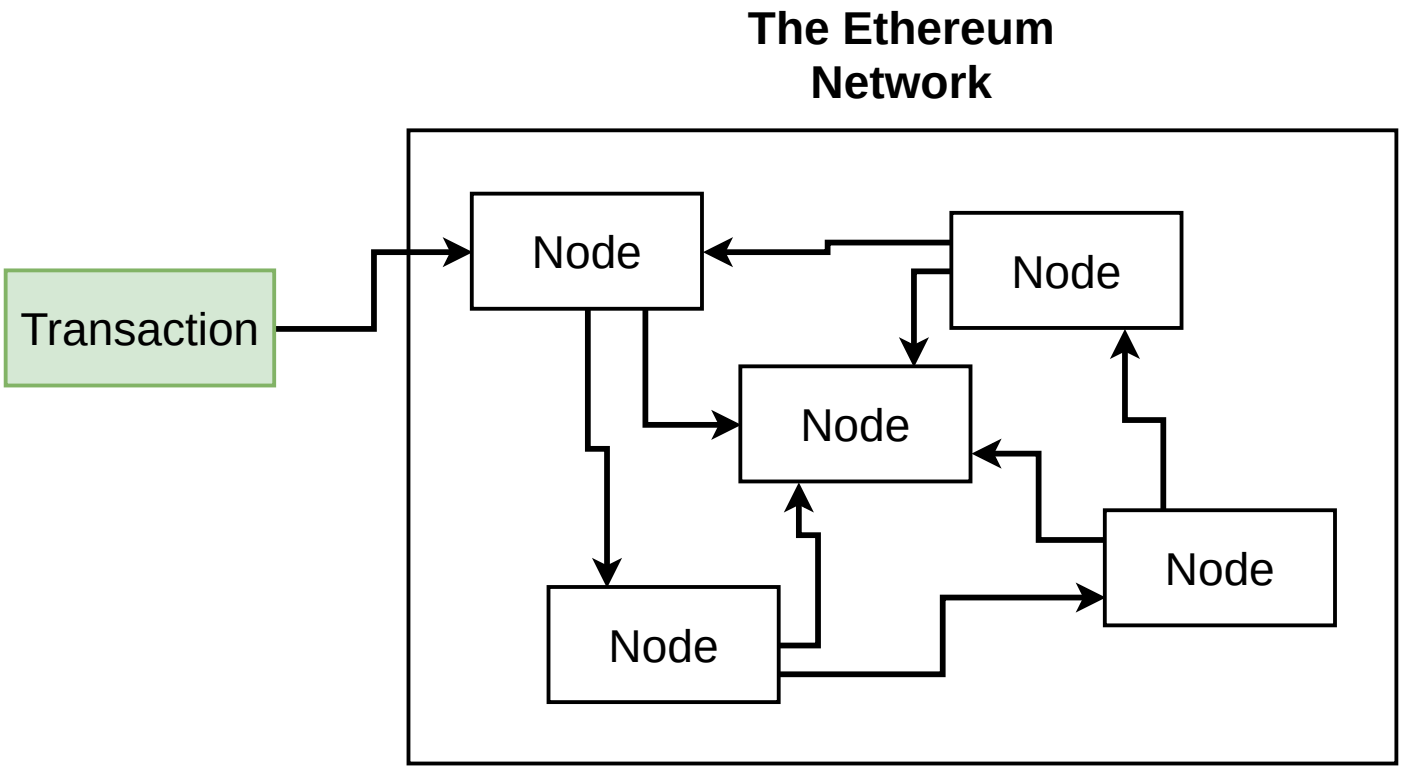

Metamask
Account Address
Public Key
Private Key
0xcf76e19404FFC87443bDA6A707299ae81A6c5c20
0x383723a06541a4f145439f6fcceed2f5e419dc8c
0xb1c6f656b2661bb15e00d1e7c41d2ce50a299c0963ab45b479e28422610561a3
Main
Ropsten
Kovan
Rinkeby

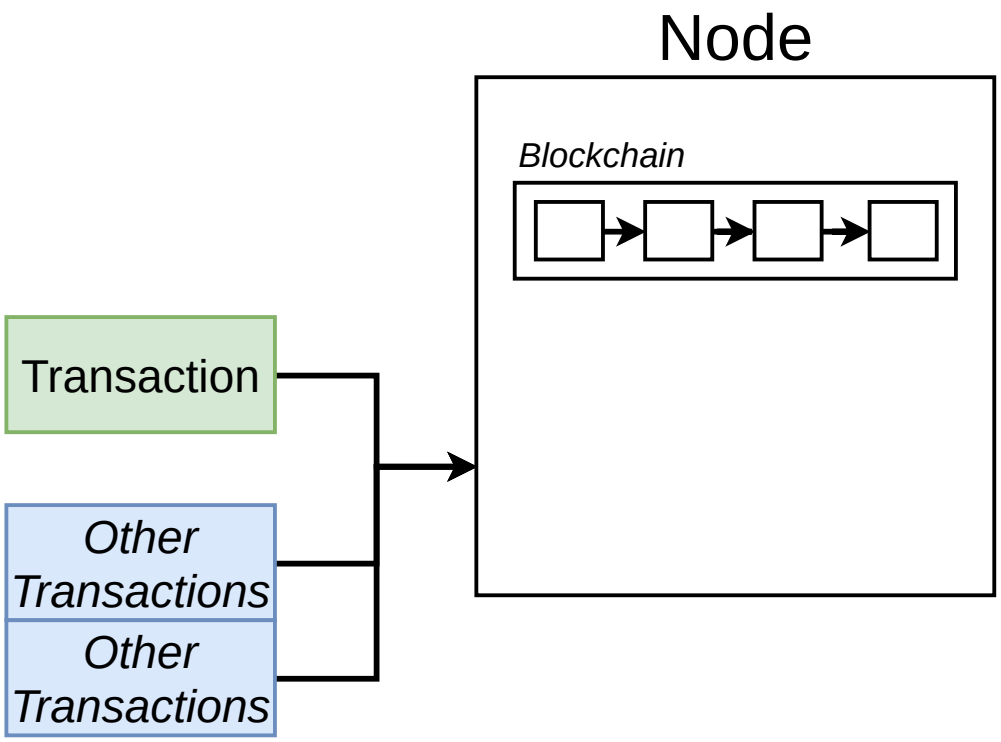

Metamask
0x8a0D7...
Account Address
Public Key
Private Key
Main
Ropsten
Kovan
Rinkeby
I think 0x8A0D7... has 0 ether
I think 0x8A0D7... has 0 ether
I think 0x8A0D7... has 0 ethers
I think 0x8A0D7... has 0.001 ether

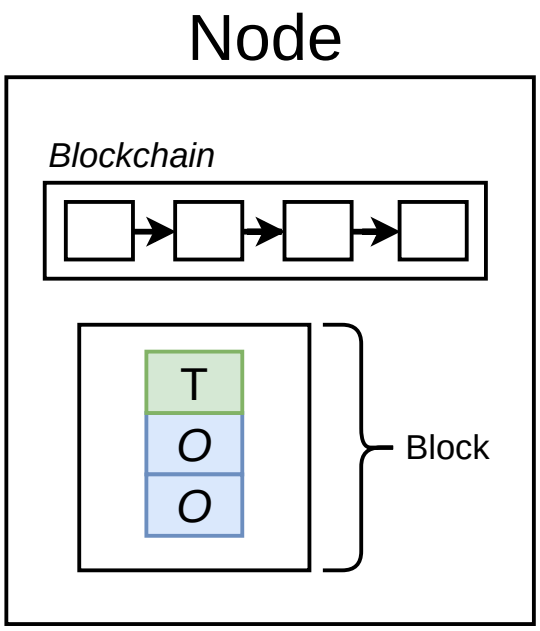

Time
Click 'submit' on form
Address sent to backend server
Backend server used web3 library to create a 'transaction' object
What's a transaction?
Backend server sent 'transaction' object to the Rinkeby test network
Backend server waited for transaction to be confirmed
Why did we wait?
Backend server sent success message back to the browser

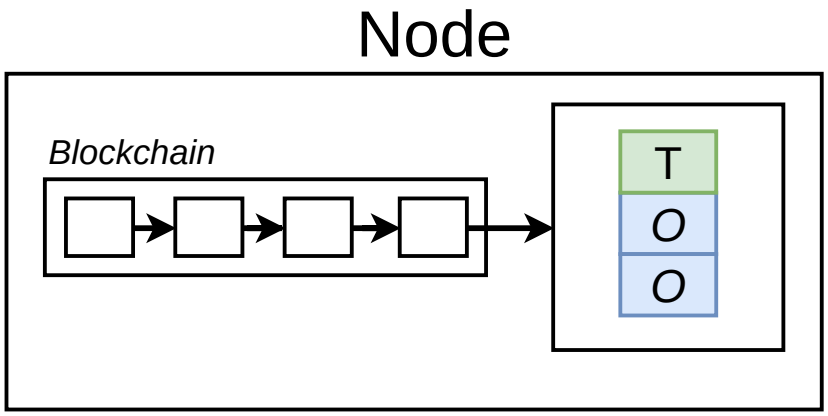
Transaction

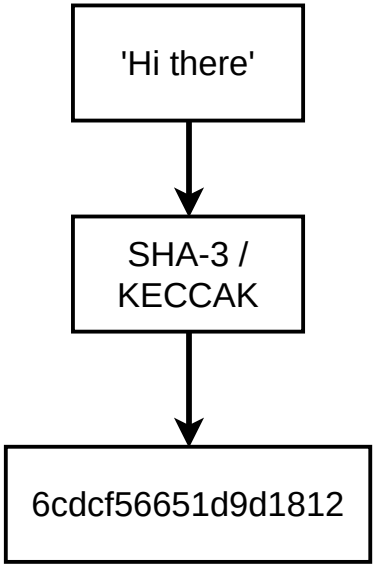
nonce	How many times the sender has sent a transaction
to	Address of account this money is going to
value	Amount of 'Wei' to send to the target address
gasPrice	Amount of Wei the sender is willing to pay per unit gas to get this transaction processed
startGas/gasLimit	Units of gas that this transaction can consume
v	Cryptographic pieces of data that can be used to generate the senders account address. Generated from the *sender's* private key.
r	
s	

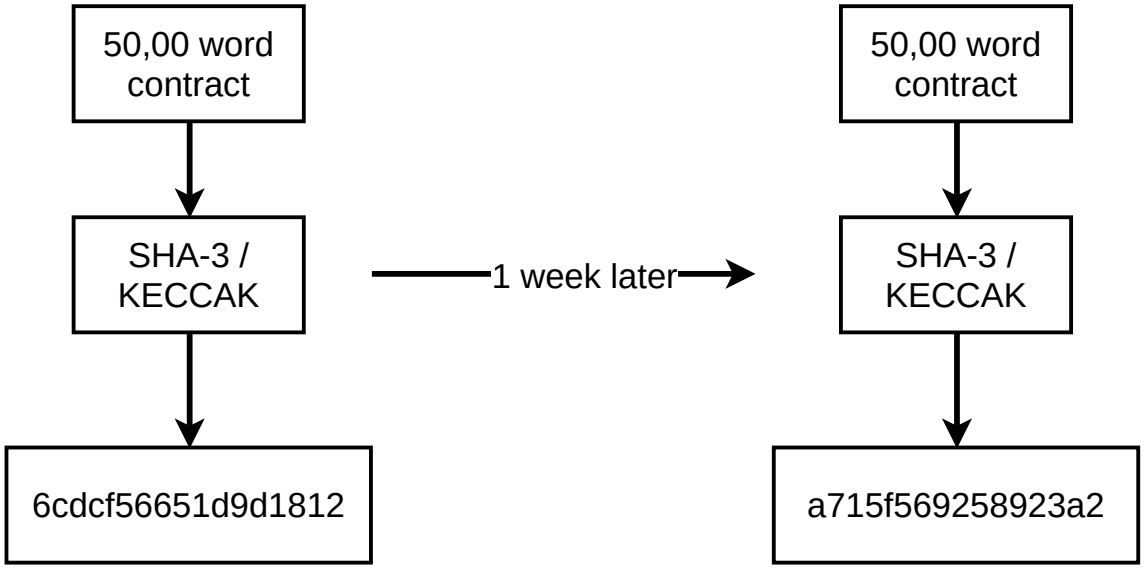

The Ethereum Network
Transaction
Node
Node
Node
Node
Node

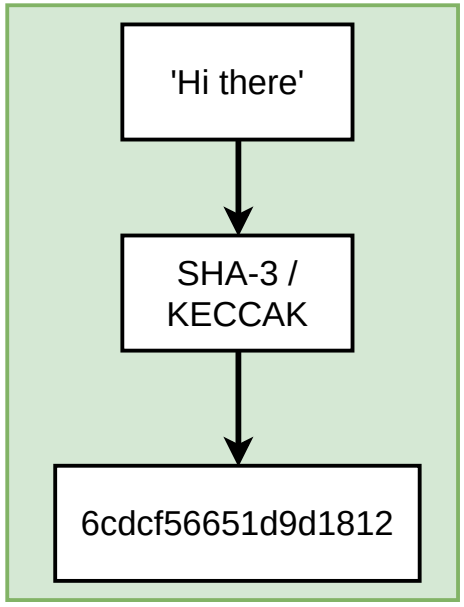

Node
Blockchain
Transaction
Other Transactions
Other Transactions

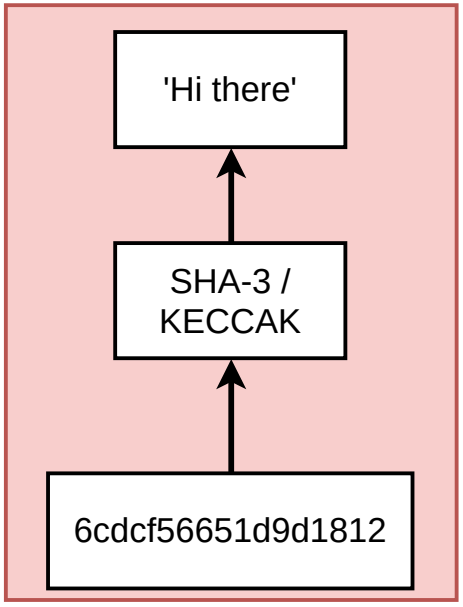

Node
Blockchain
T
O
O
Block


Node
Blockchain
T
O
O


'Hi there'
SHA-3 /
KECCAK
6cdcf56651d9d1812


50,00 word contract
SHA-3 / KECCAK
6cdcf56651d9d1812
1 week later
50,00 word contract
SHA-3 / KECCAK
a715f569258923a2

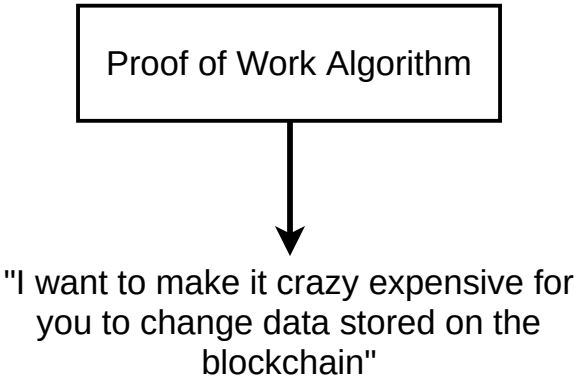

'Hi there'
SHA-3 /
KECCAK
6cdcf56651d9d1812

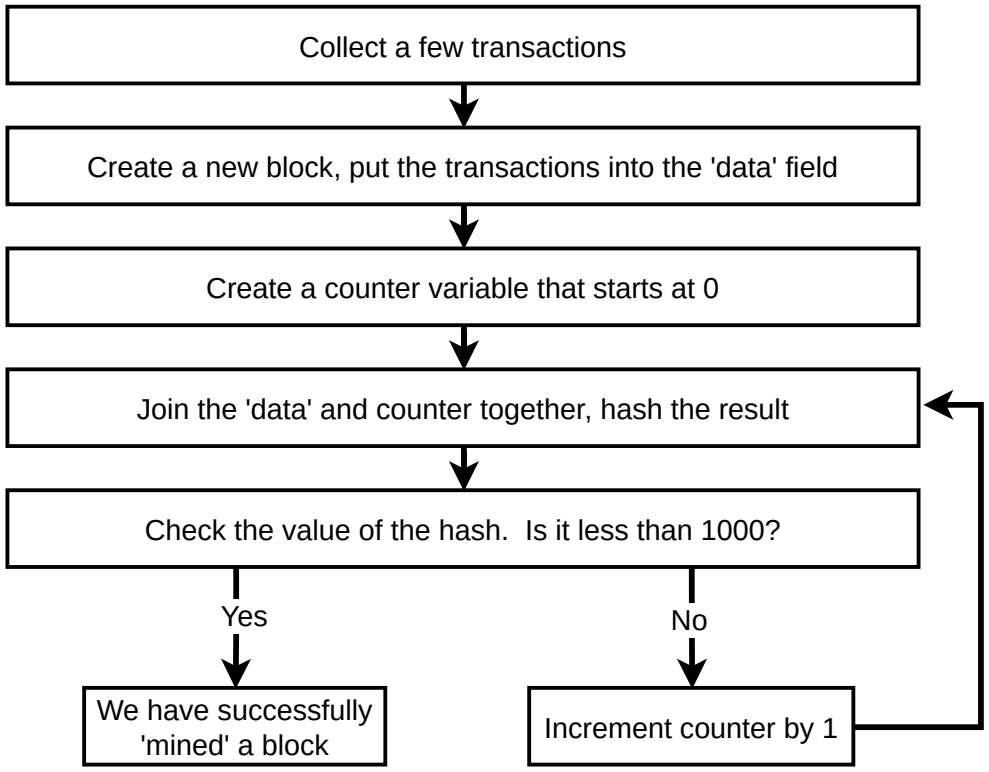

'Hi there'
SHA-3 /
KECCAK
6cdcf56651d9d1812

This is Bitcoin's proof of work algorithm

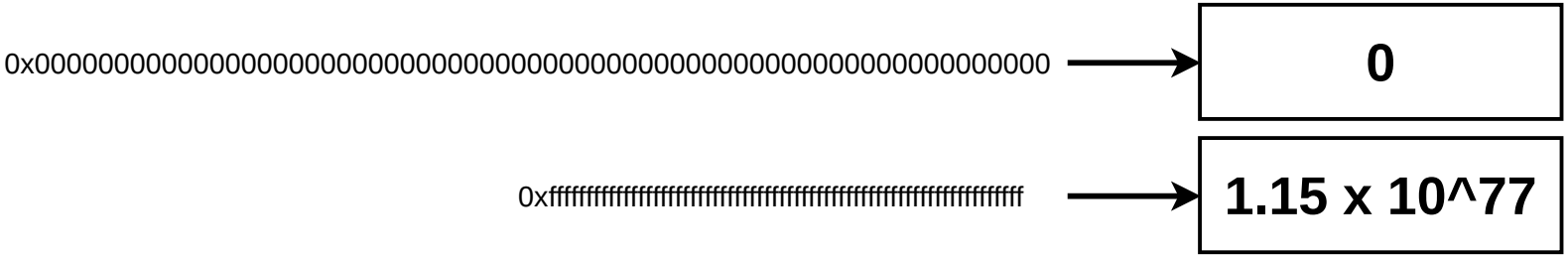
Ethereums is significantly more complicated.

The implementation of these algorithms is not important. The *purpose* is.


Proof of Work Algorithm
"I want to make it crazy expensive for you to change data stored on the blockchain"


Collect a few transactions
Create a new block, put the transactions into the 'data' field
Create a counter variable that starts at 0
Join the 'data' and counter together, hash the result
Check the value of the hash. Is it less than 1000?
Yes
No
We have successfully 'mined' a block
Increment counter by 1

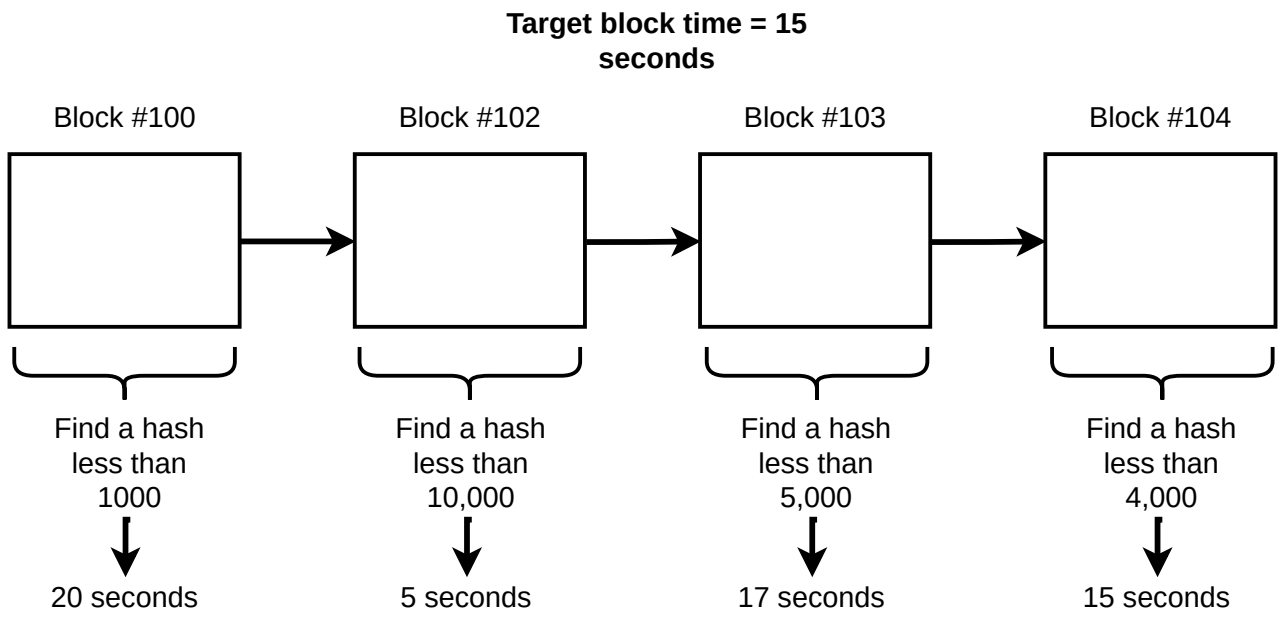
Data	+	Nonce	=	Output Hash	Output hash as a base 10 number	Is this less than 1000?
'Hi There'		0		a23042b2e	178917215	no
'Hi There'		1		cbc1491	29589283	no
'Hi There'		2		0ca24258	94869869	no
'Hi There'		3		d9eed91	13938166	no
'Hi There'		4		1488baec	419386918	no
'Hi There'		5		0077bbb	100	yes

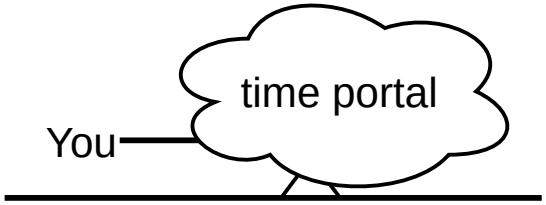

0x0000000000000000000000000000000000000000000000000000000000000000
0
0xffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff
1.15 x 10^77

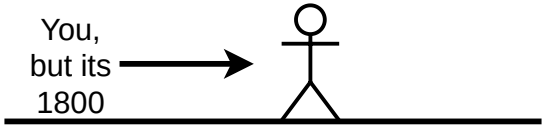
Data	+	Nonce	=	Output Hash	Output hash as a base 10 number	Is this less than 1000?
'Hi There'		0		a23042b2e	178917215	no
'Hi There'		1		cbc1491	29589283	no
'Hi There'		2		0ca24258	94869869	no
'Hi There'		3		d9eed91	13938166	no
'Hi There'		4		1488baec	419386918	no
'Hi There'		5		0077bbb	100	yes

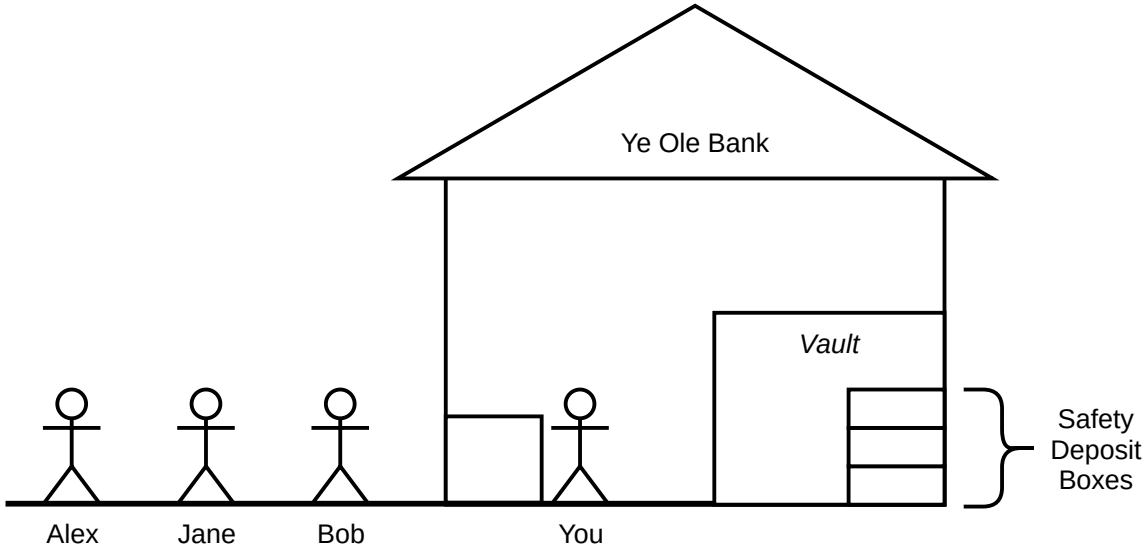
T

Time to find a
solution =
Block TIme

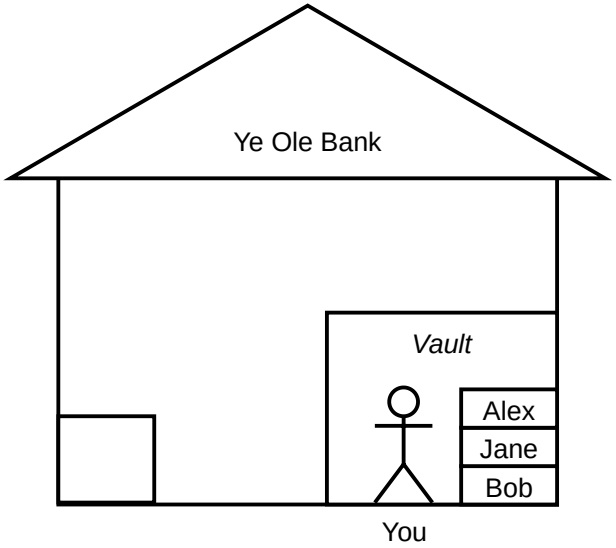

Target block time = 15 seconds
Block #100
Block #102
Block #103
Block #104
Find a hash less than 1000
Find a hash less than 10,000
Find a hash less than 5,000
Find a hash less than 4,000
20 seconds
5 seconds
17 seconds
15 seconds


time portal
You


You,
but its
1800


Ye Ole Bank
Vault
Safety
Deposit
Boxes
Alex
Jane
Bob
You

Today's Ledger		
From	To	Amount
Alex	Jane	$100
Jane	Bob	$200
Alex	Bob	$40
Bob	Alex	$75



Today's Ledger		
From	To	Amount
Alex	Jane	$100
Jane	Bob	$200
Alex	Bob	$40
Bob	Alex	$75

Day #3 Account Balances

$80	$150	$70
Alex	Jane	Bob

Initial Deposits

$0	$0	$0
Alex	Jane	Bob

Day #1 Ledger

From	To	Amount
Alex	Alex	$100
Jane	Jane	$100
Bob	Bob	$100

Balances at end of Day #1

$100	$100	$100
Alex	Jane	Bob

Day #2 Ledger

From	To	Amount
Bob	Jane	$30
Bob	Alex	$30

Balances at end of Day #2

$130	$130	$40
Alex	Jane	Bob

Day #3 Ledger

From	To	Amount
Jane	Alex	$10
Jane	Bob	$30

Balances at end of Day #3

$140	$90	$70
Alex	Jane	Bob

Block of Transactions

Day #1 Ledger		
From	To	Amount
Alex	Alex	$100
Jane	Jane	$100
Bob	Bob	$100

Block of Transactions

Day #2 Ledger		
From	To	Amount
Bob	Jane	$30
Bob	Alex	$30

Block of Transactions

Day #3 Ledger		
From	To	Amount
Jane	Alex	$10
Jane	Bob	$30

Each day's ledger can be referred to as a **block**. A **block** is a group of transactions

Each **block**'s transactions are finalized at the end of the day

The time between finalizing each **block** is referred to as the **block time.**

We can determine the current balance of everyone by replaying all transactions in all **blocks**

Day #1 Ledger		
From	To	Amount
Alex	Alex	$100
Jane	Jane	$100
Bob	Bob	$100

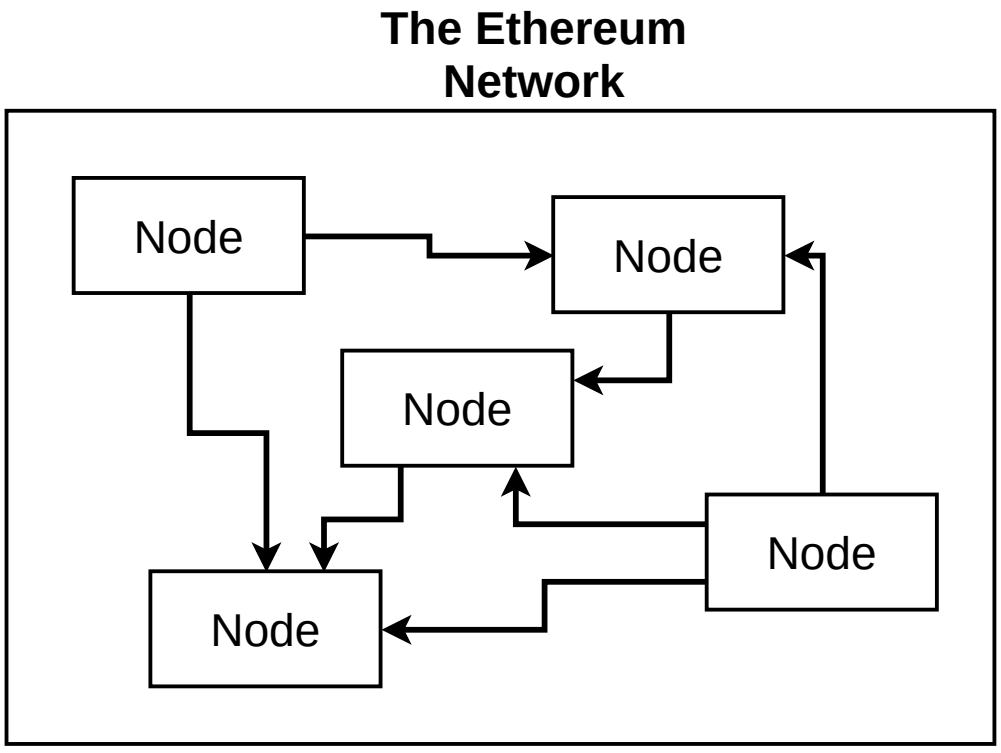
Day #2 Ledger		
From	To	Amount
Bob	Jane	$30
Bob	Alex	$30

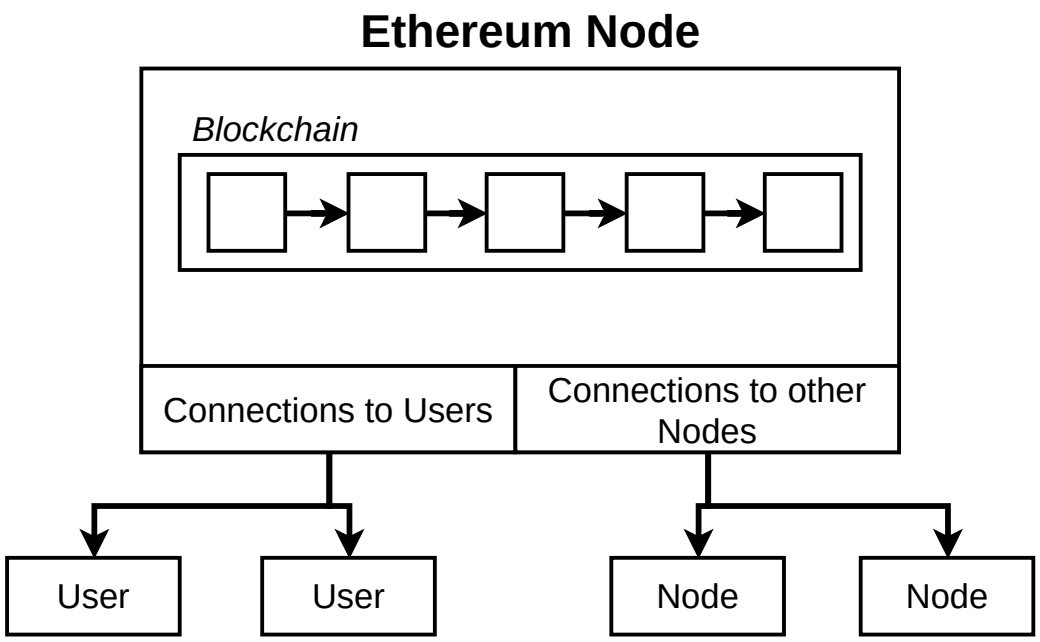
Day #3 Ledger		
From	To	Amount
Jane	Alex	$10
Jane	Bob	$30

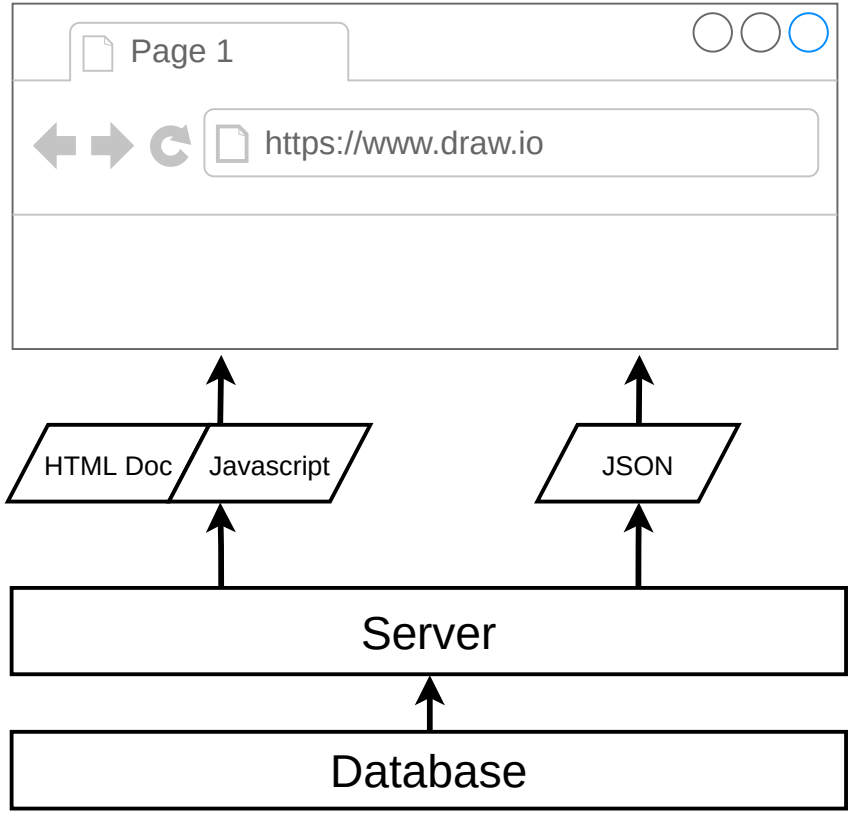
Day #1 Ledger		
From	To	Amount
Alex	Alex	$100
Jane	Jane	$100
Bob	Bob	$100

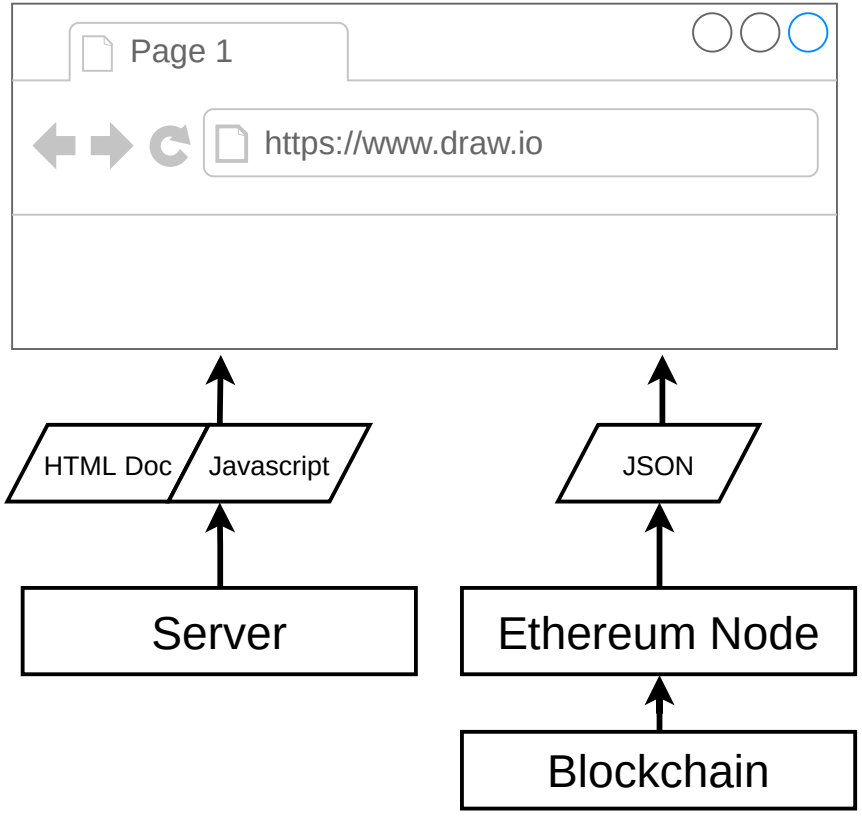
Day #2 Ledger		
From	To	Amount
Bob	Jane	$30
Bob	Alex	$30

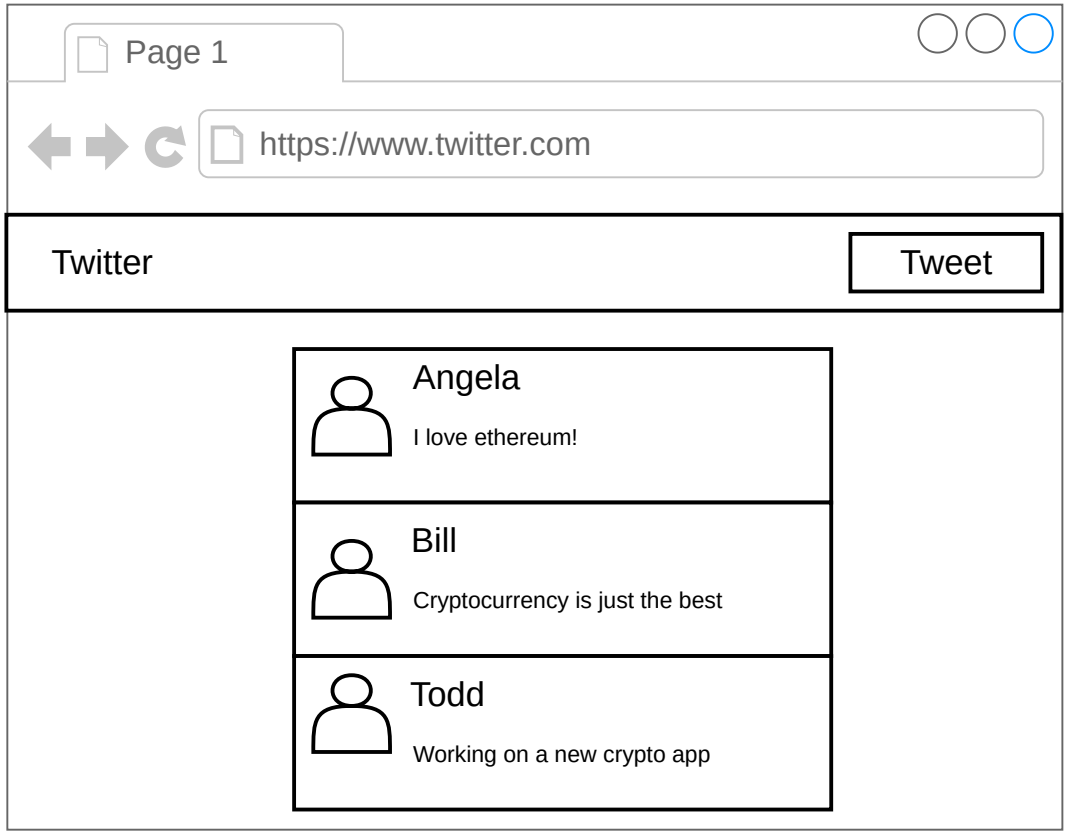
Day #3 Ledger		
From	To	Amount
Jane	Alex	$10
Jane	Bob	$30

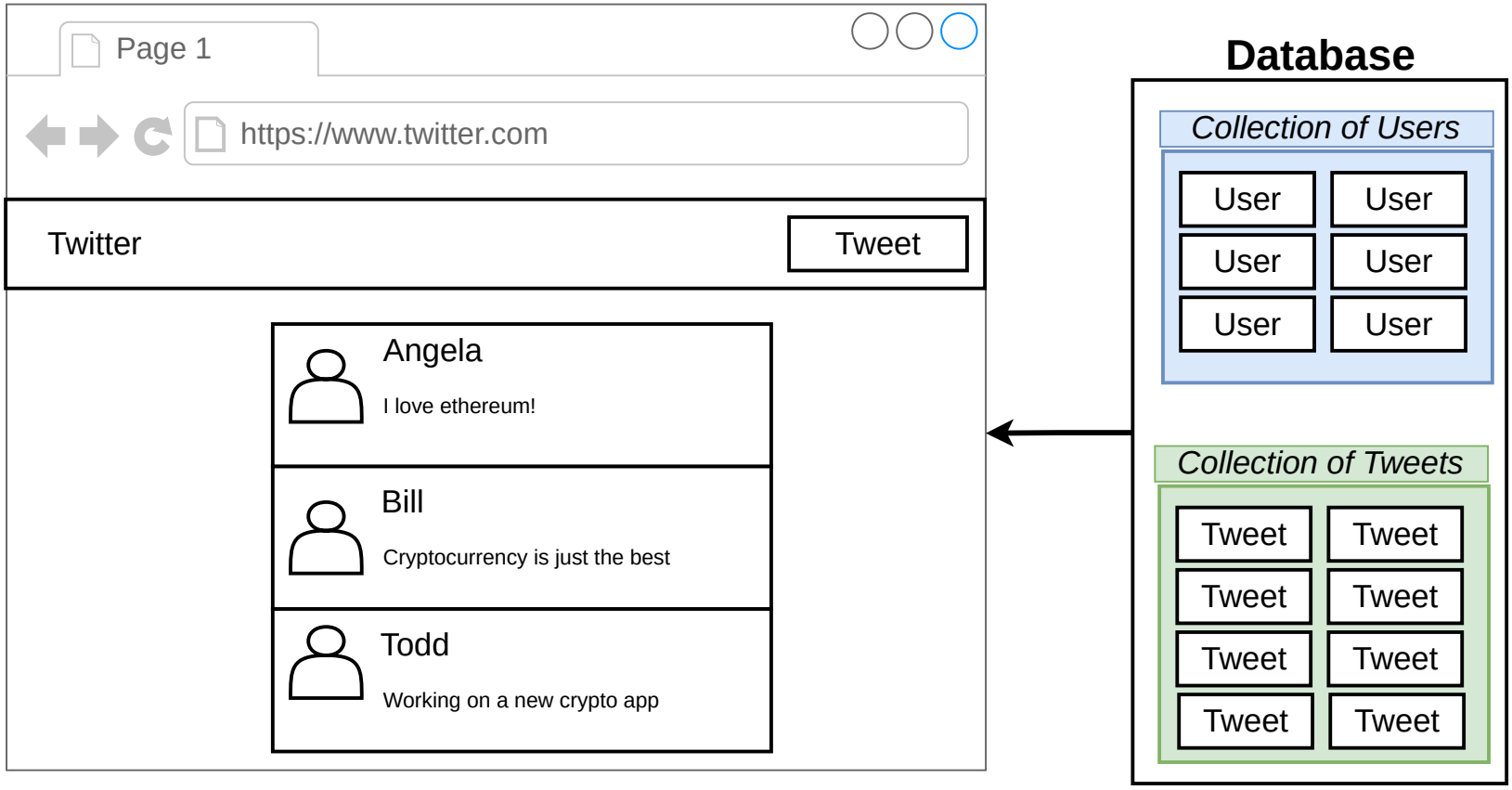

The Ethereum Network
Node
Node
Node
Node
Node

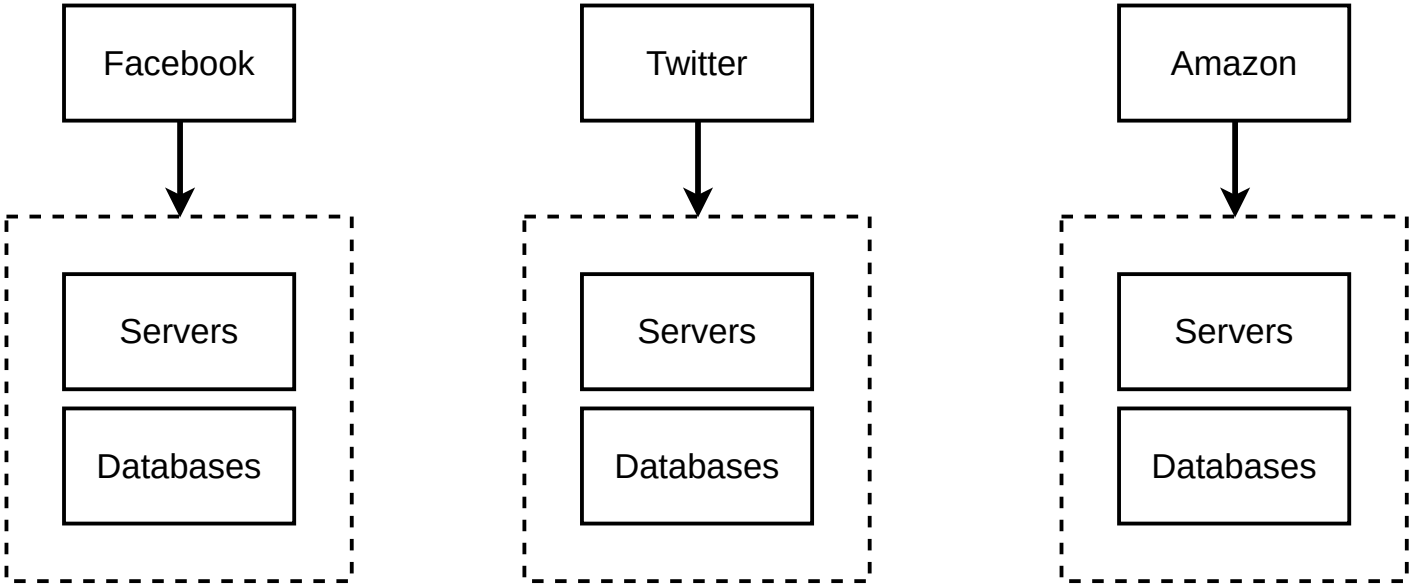

Ethereum Node
Blockchain
Connections to Users
Connections to other Nodes
User
User
Node
Node

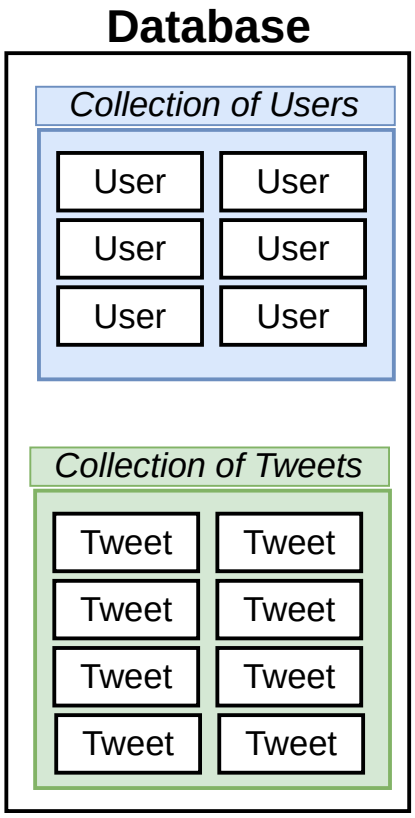

Page 1
https://www.draw.io
HTML Doc
Javascript
JSON
Server
Database

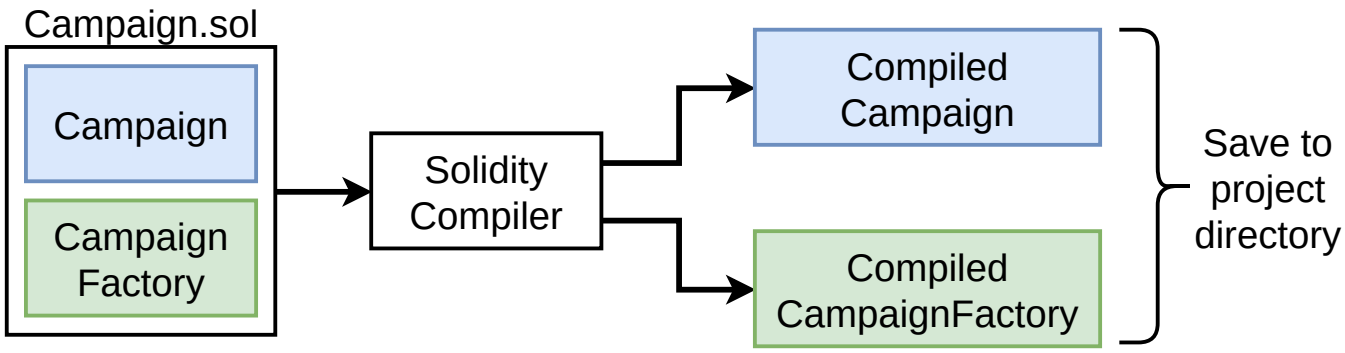

Page 1
https://www.draw.io
HTML Doc
Javascript
JSON
Server
Ethereum Node
Blockchain

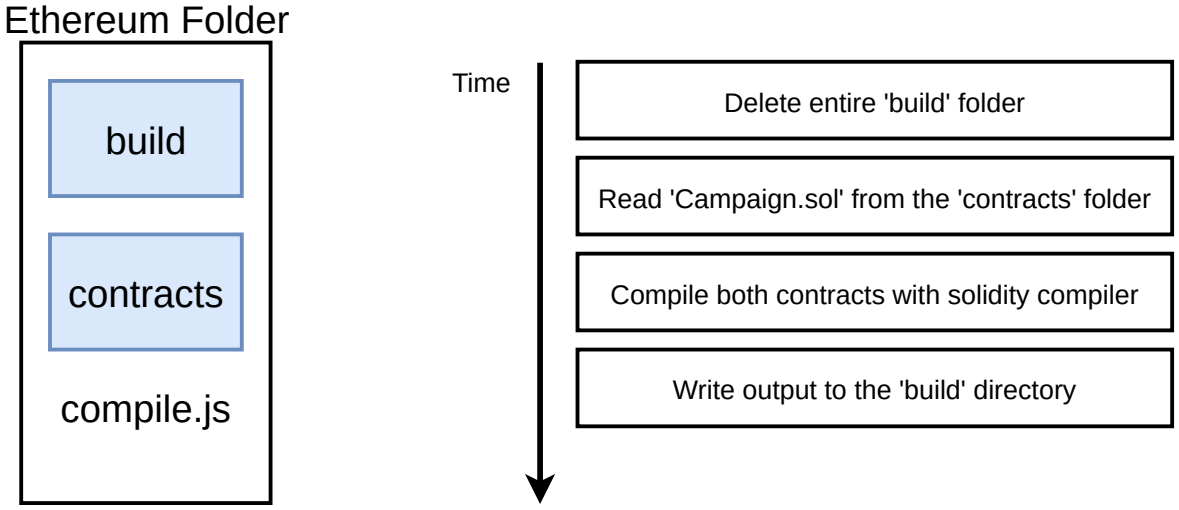

Page 1
https://www.twitter.com
Twitter
Tweet
Angela
I love ethereum!
Bill
Cryptocurrency is just the best
Todd
Working on a new crypto app

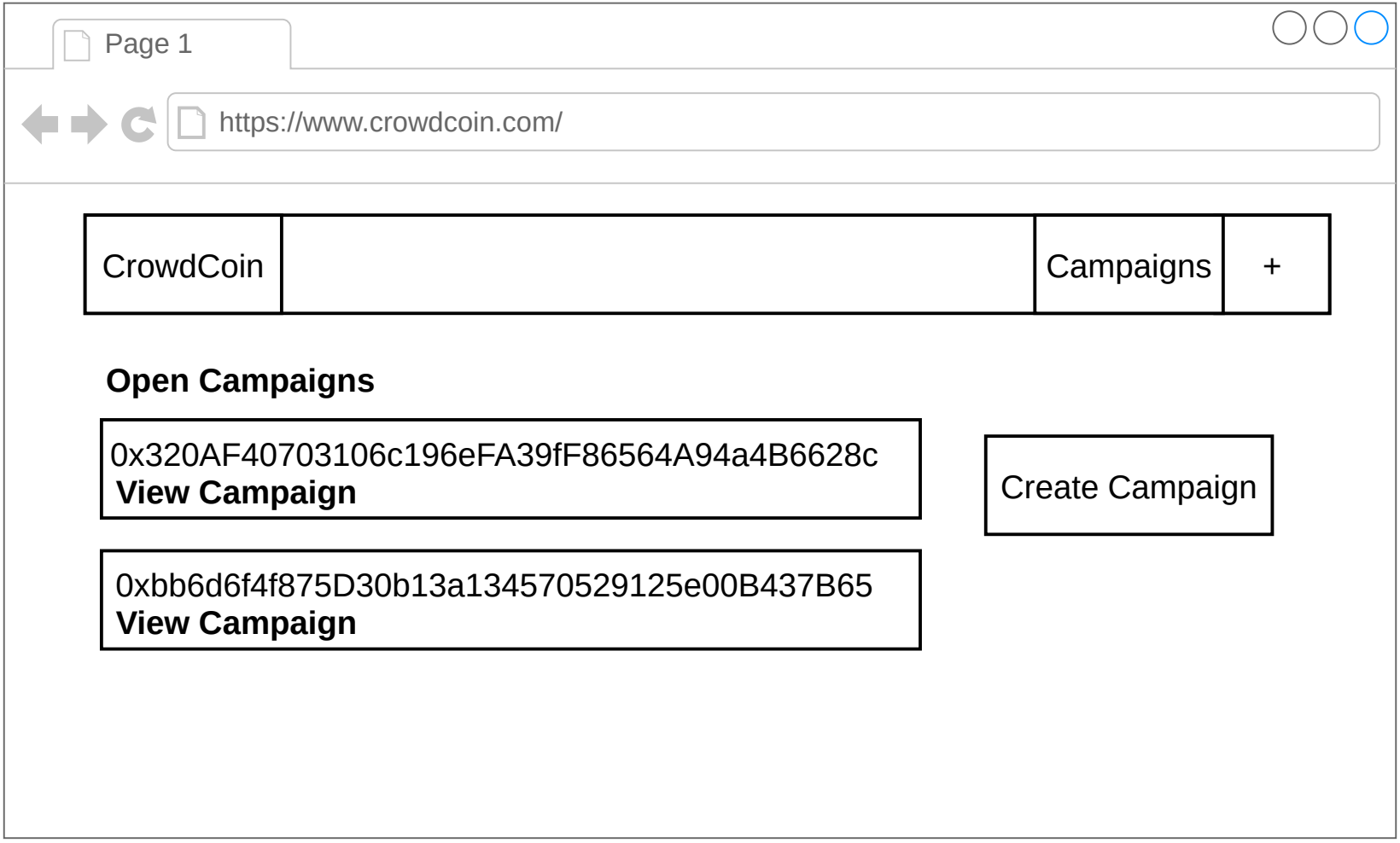

Page 1
https://www.twitter.com
Twitter
Tweet
Angela
I love ethereum!
Bill
Cryptocurrency is just the best
Todd
Working on a new crypto app
Database
Collection of Users
User
User
User
User
User
User
Collection of Tweets
Tweet
Tweet
Tweet
Tweet
Tweet
Tweet
Tweet
Tweet

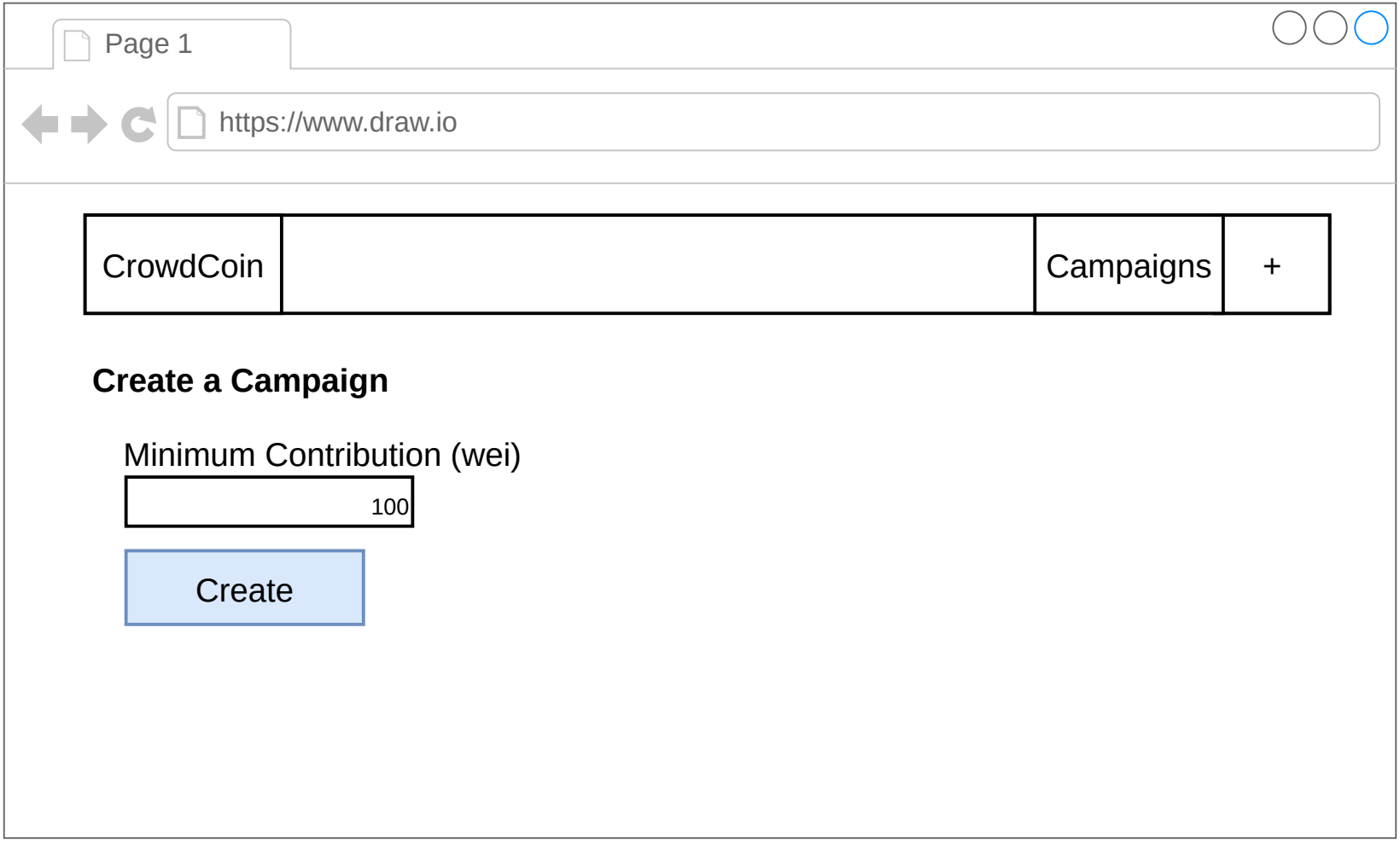

Facebook
Servers
Databases
Twitter
Servers
Databases
Amazon
Servers
Databases

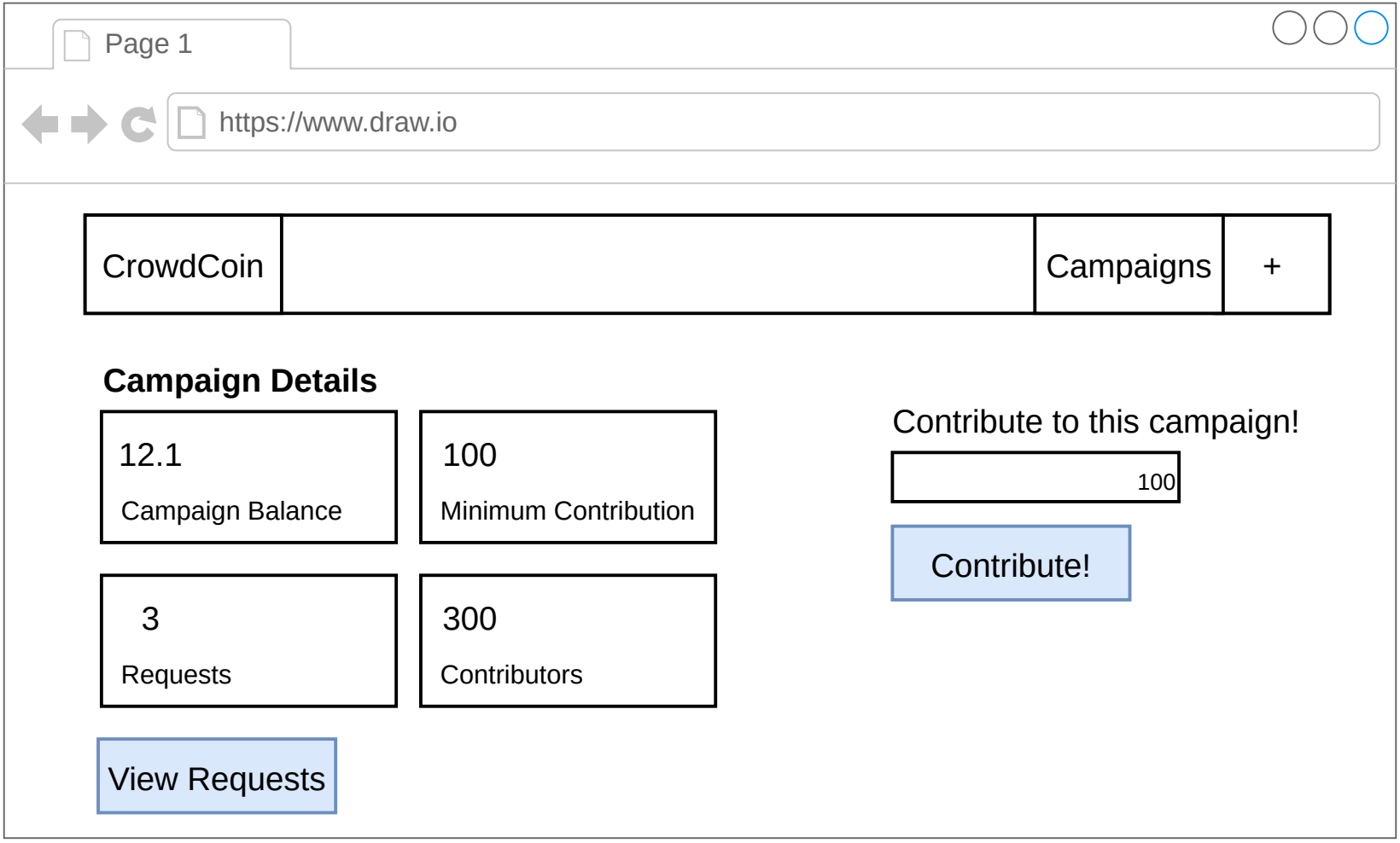

Database
Collection of Users
User
User
User
User
User
User
Collection of Tweets
Tweet
Tweet
Tweet
Tweet
Tweet
Tweet
Tweet
Tweet


Campaign.sol
Campaign
Campaign Factory
Solidity Compiler
Compiled Campaign
Compiled CampaignFactory
Save to project directory


Ethereum Folder
build
contracts
compile.js
Time
Delete entire 'build' folder
Read 'Campaign.sol' from the 'contracts' folder
Compile both contracts with solidity compiler
Write output to the 'build' directory


Page 1
https://www.crowdcoin.com/
CrowdCoin
Campaigns
+
Open Campaigns
0x320AF40703106c196eFA39fF86564A94a4B6628c
View Campaign
Create Campaign
0xbb6d6f4f875D30b13a134570529125e00B437B65
View Campaign


Page 1
https://www.draw.io
CrowdCoin
Campaigns
+
Create a Campaign
Minimum Contribution (wei)
100
Create


Page 1
https://www.draw.io
CrowdCoin
Campaigns
+
Campaign Details
12.1
Campaign Balance
100
Minimum Contribution
3
Requests
300
Contributors
View Requests
Contribute to this campaign!
100
Contribute!

Page 1

https://www.draw.io

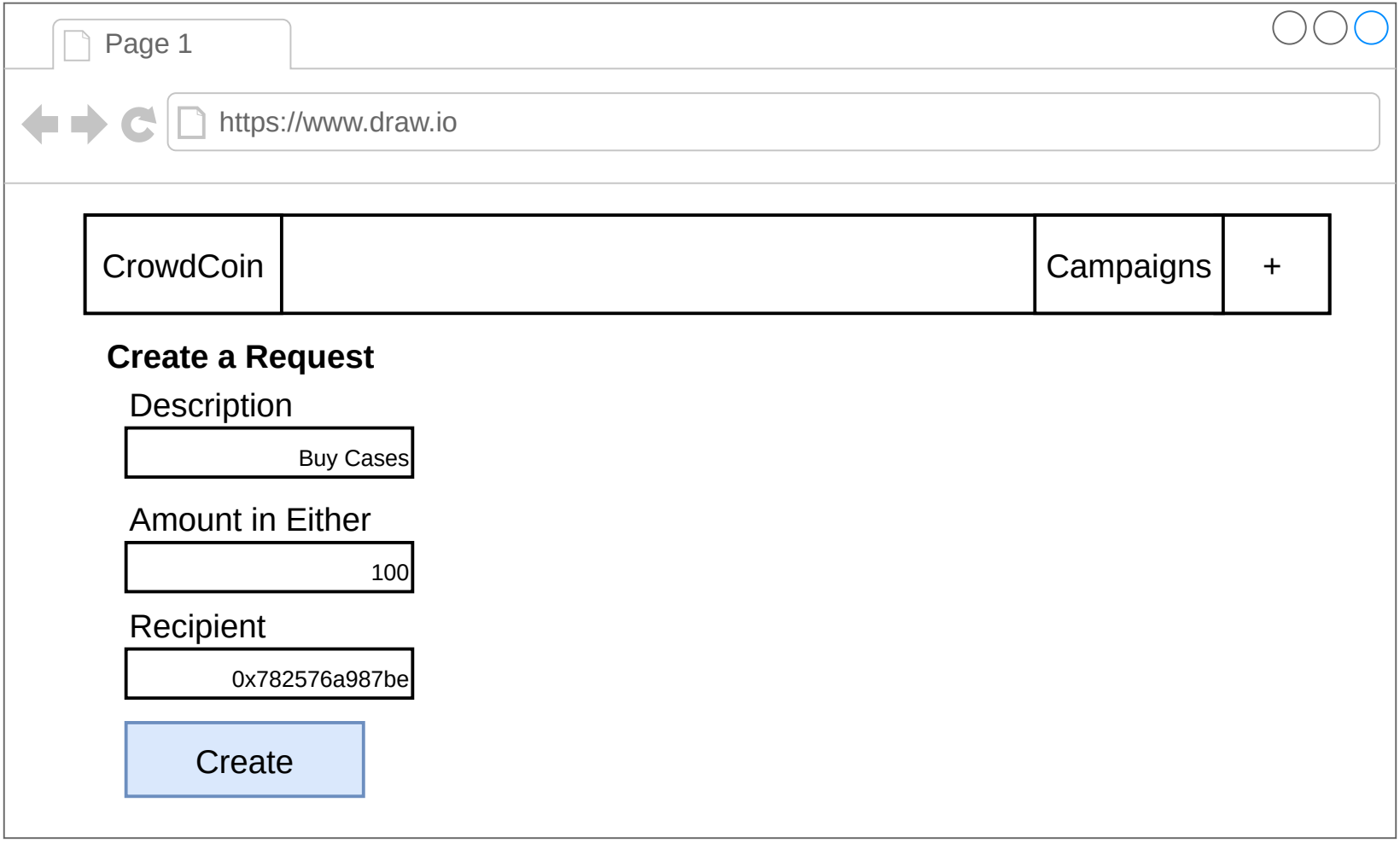
CrowdCoin		Campaigns	+

Pending Requests

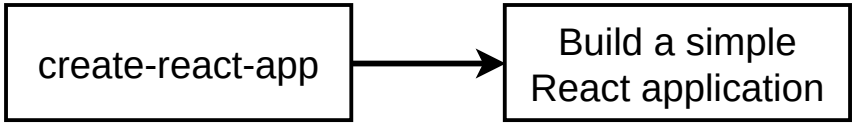
Add Request

ID	Description	Amount	Recipient	Approval Count	Approve	Finalize
1	Buy Batteries	1	0x65ace	120/300	Approve	Finalize

Found 1 Request


Page 1
https://www.draw.io
CrowdCoin
Campaigns
+
Create a Request
Description
Buy Cases
Amount in Either
100
Recipient
0x782576a987be
Create

Routing	
Path	We should show...
/	List of Campaigns
/campaigns/new	Form to make a campaign
/campaigns/0x123aec	Campaign details for campaign at address 0x8147
/campaigns/0x8147/requests	Requests for campaign at address 0x8147
/campaigns/0x8147/requests/new	Form to create a request for campaign at address 0x8147



By default, doesn't include anything for navigation, data loading, etc, etc, etc

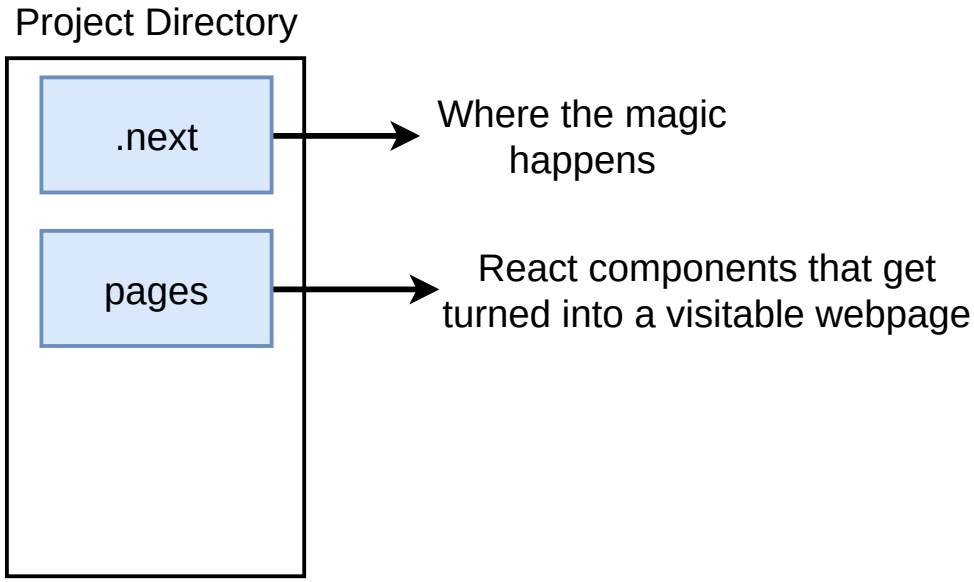
Next.js

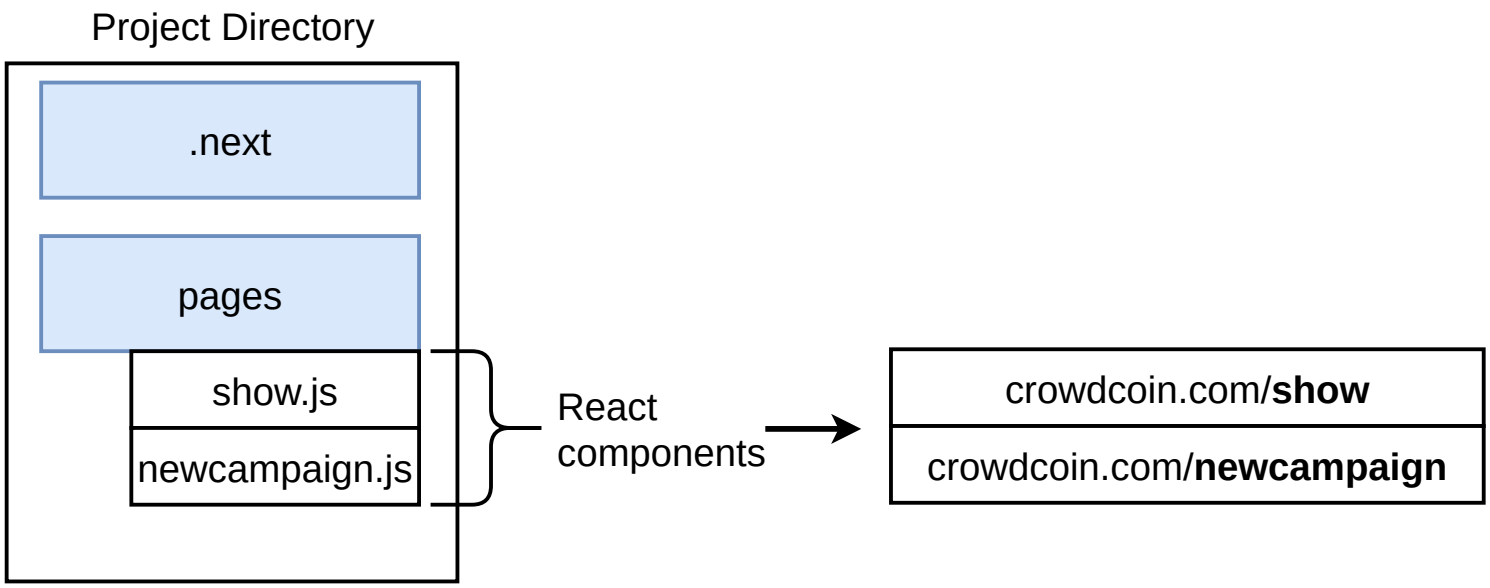
Wraps up React + associated tools into one package

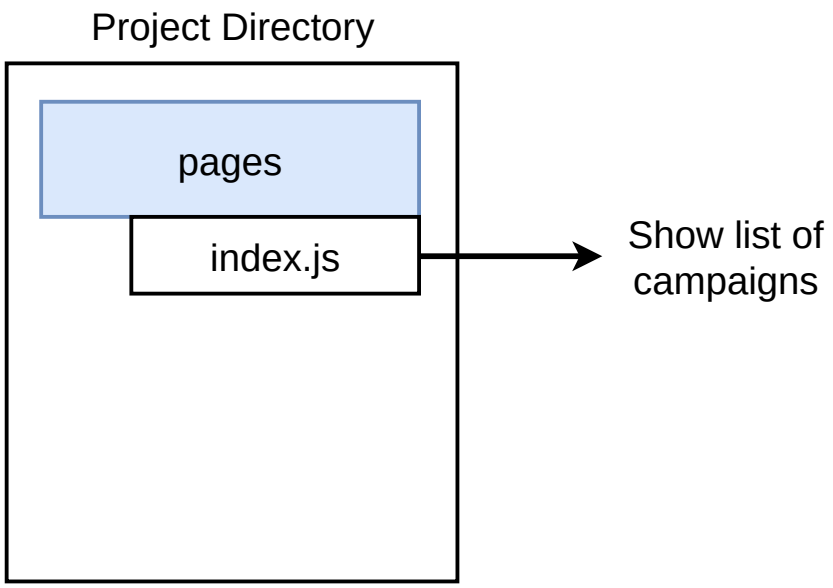
Lots of fancy features included out of the box

Routing
Server side rendering
Hot module reload

Makes it really, really easy to use React to make a multi-page application


Project Directory
.next
Where the magic happens
pages
React components that get turned into a visitable webpage

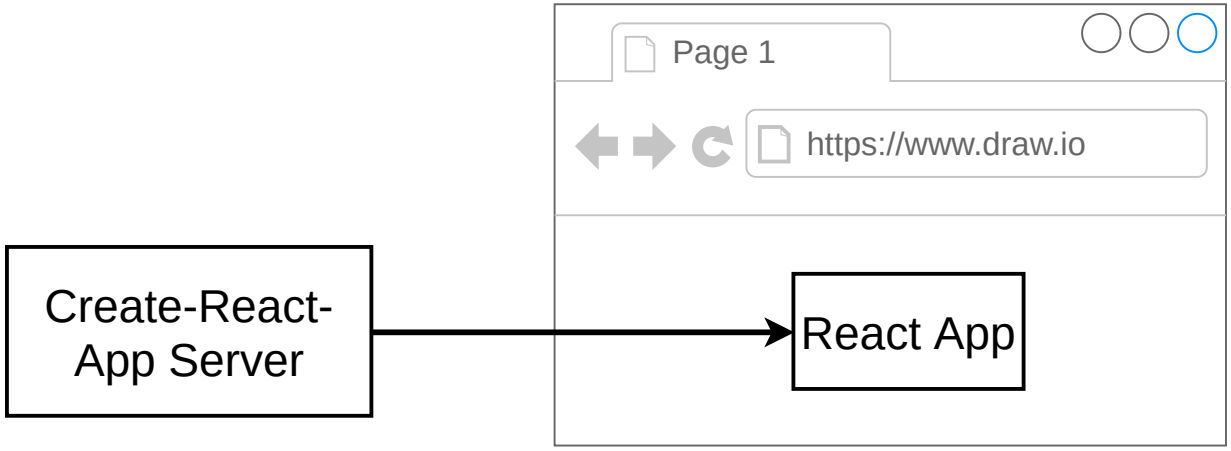

Project Directory
.next
pages
show.js
newcampaign.js
React
components
crowdcoin.com/**show**
crowdcoin.com/**newcampaign**

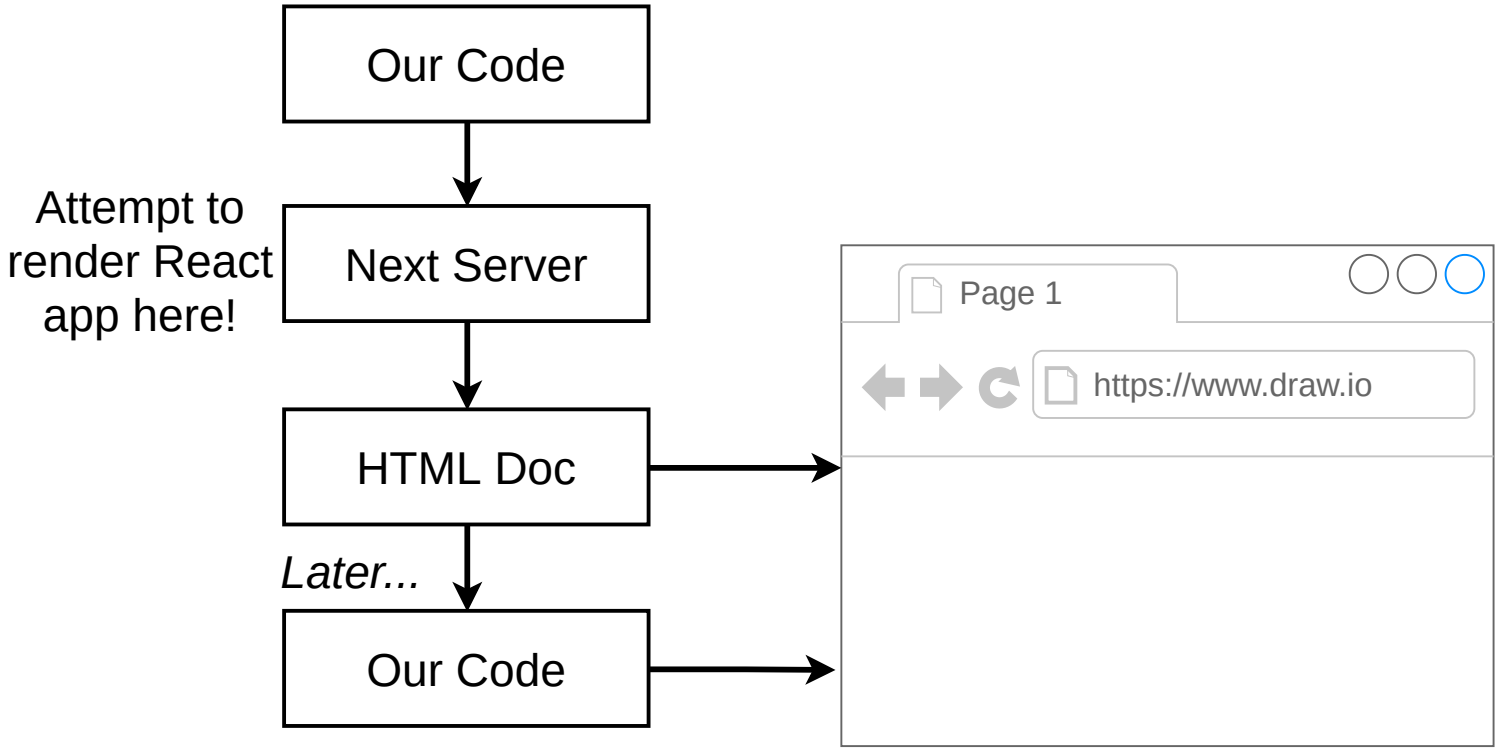

Project Directory
pages
index.js
Show list of campaigns

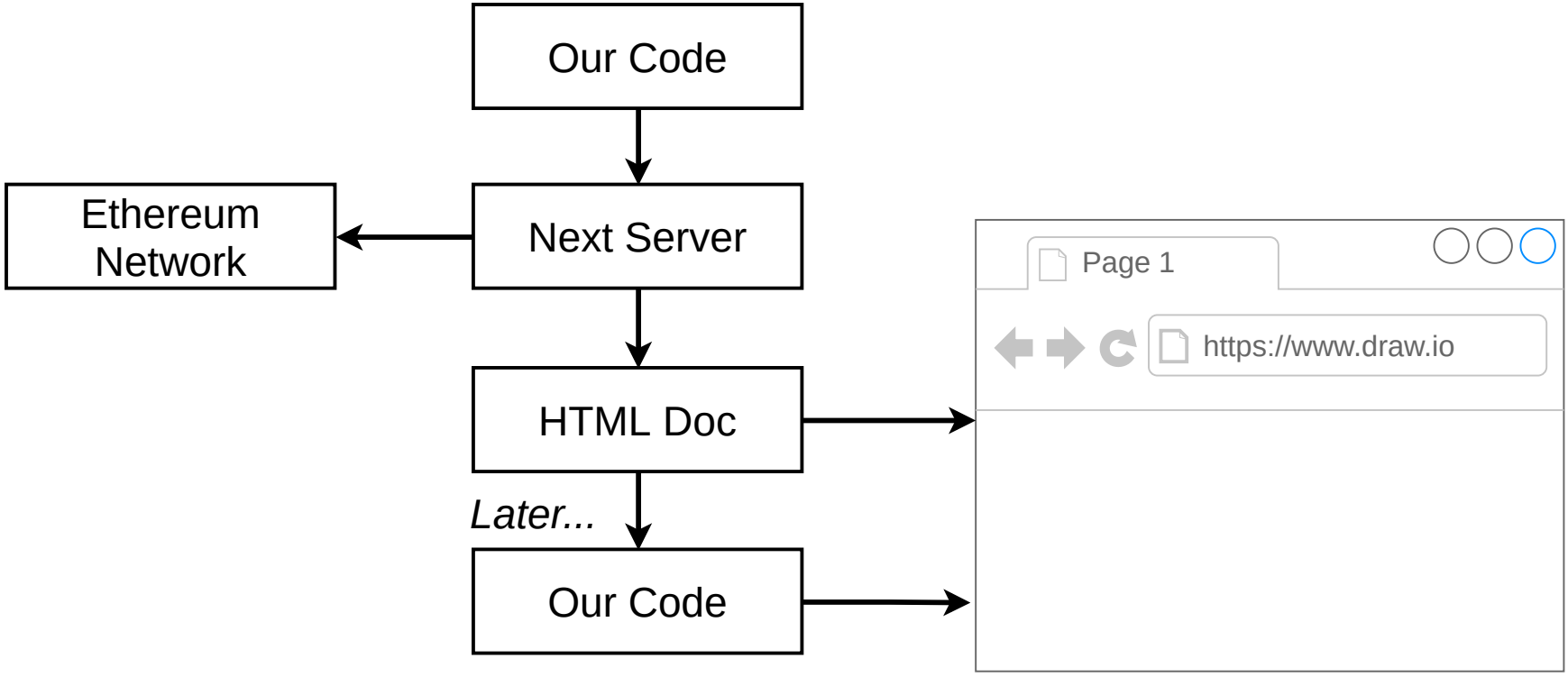
To-do's for the Campaign List Page

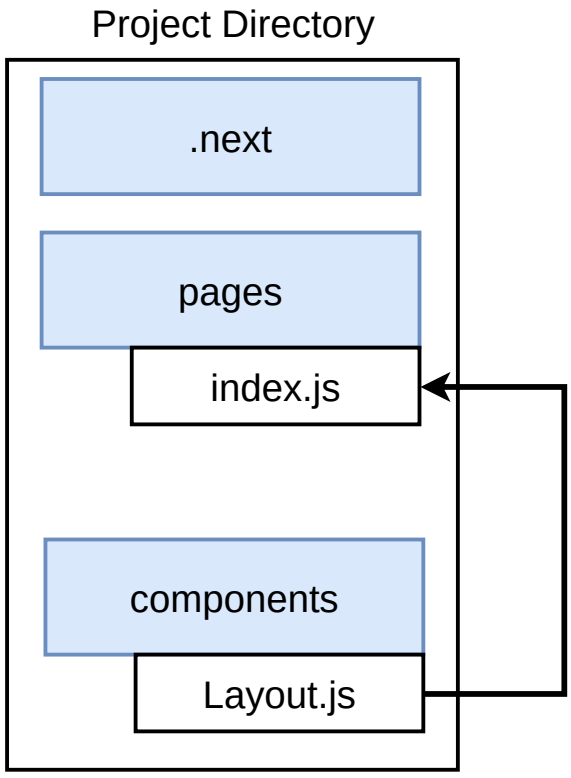
Steps

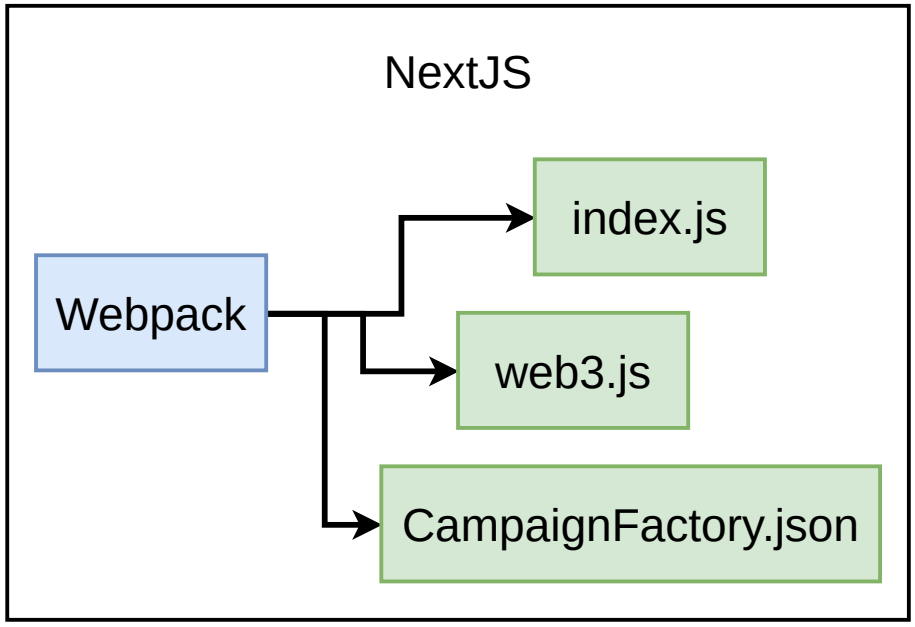
- Configure web3 with a provider from metamask
- Tell web3 that a deployed copy of the 'CampaignFactory' exists
- Use Factory instance to retrieve a list of deployed campaigns
- Use React to show something about each campaign

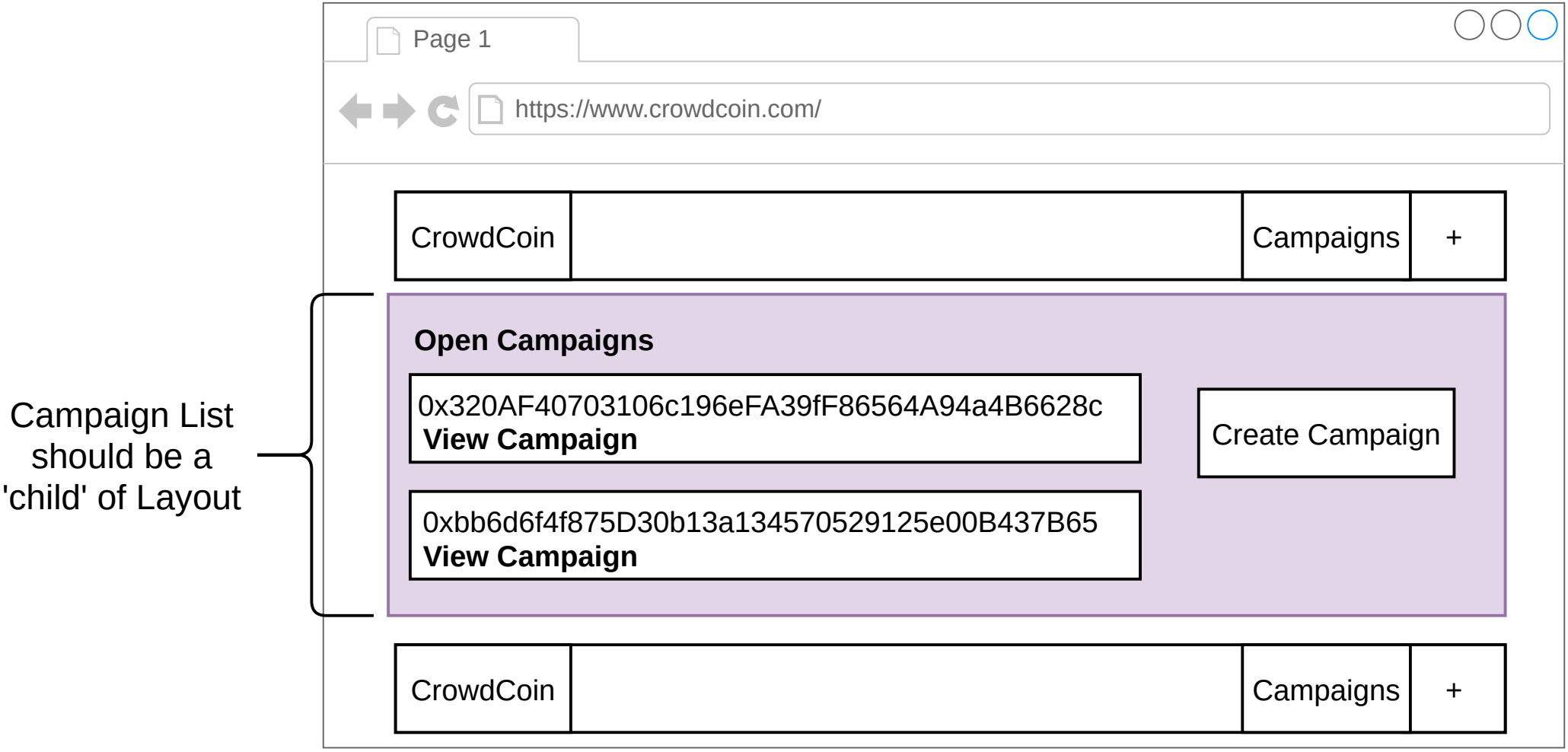

Page 1
https://www.draw.io
Create-React-App Server
React App

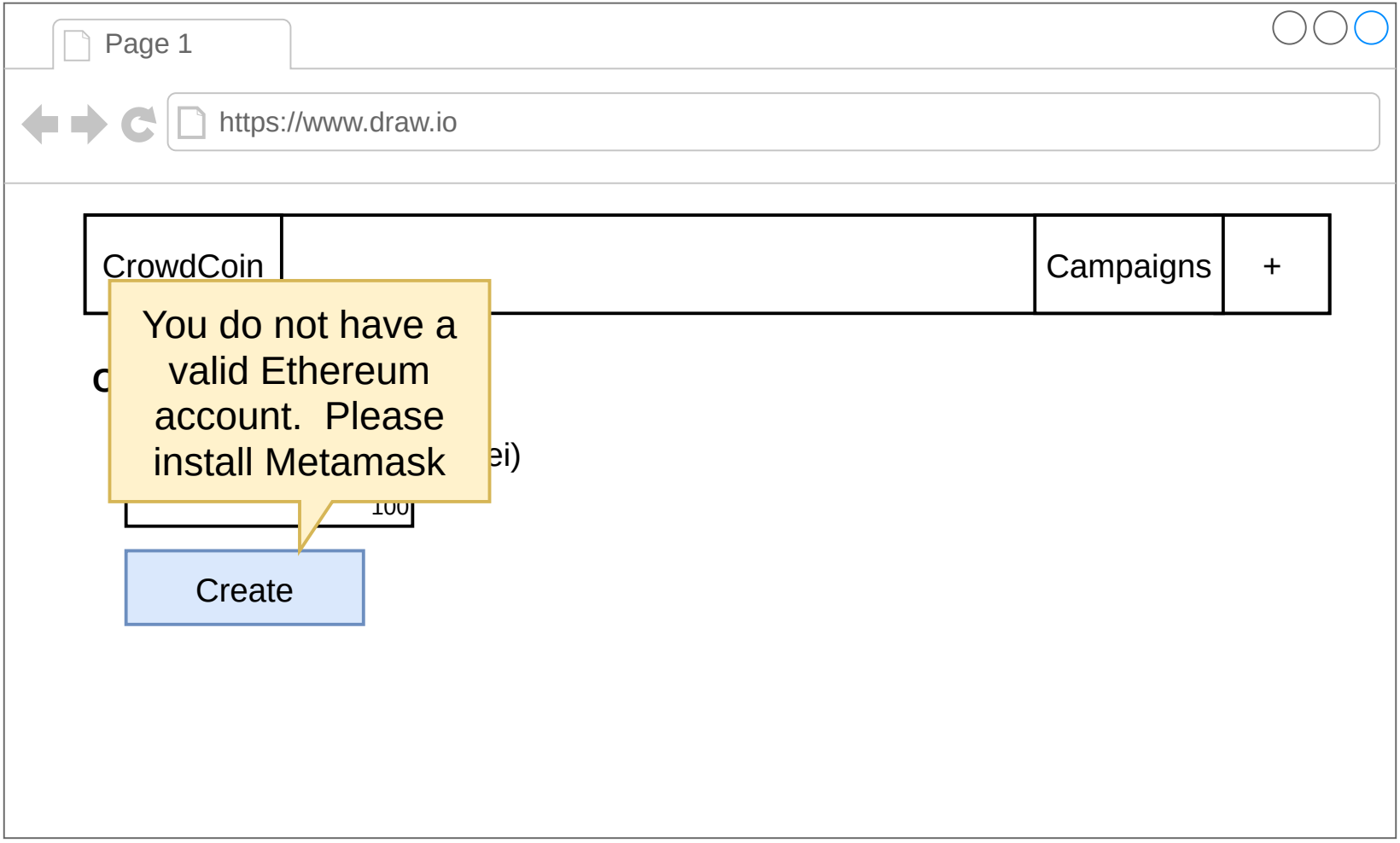

Our Code
Attempt to render React app here!
Next Server
HTML Doc
Later...
Our Code
Page 1
https://www.draw.io

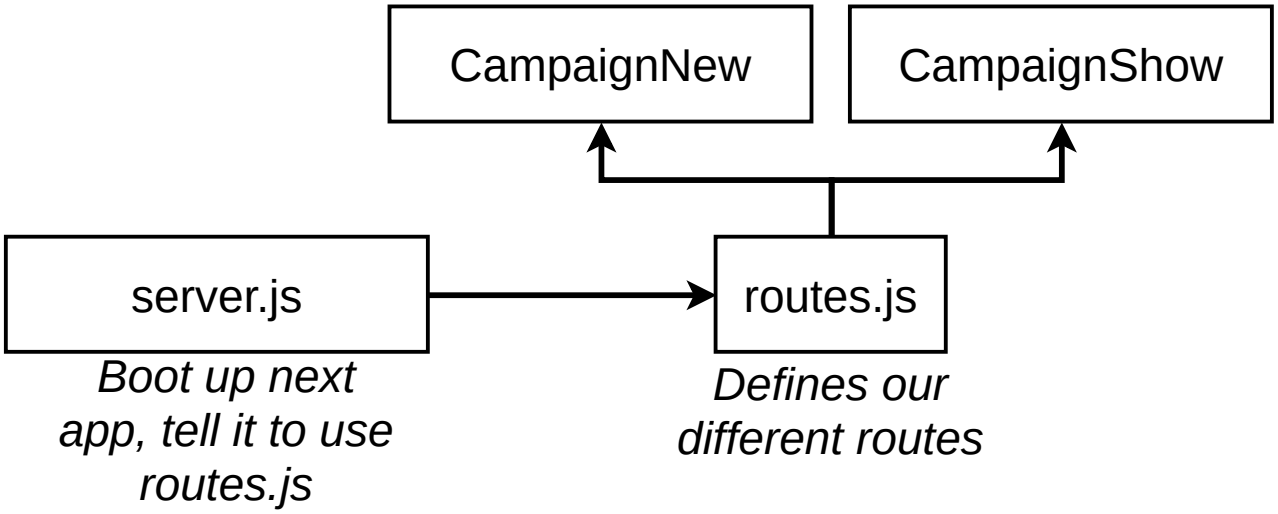

Our Code
Ethereum Network
Next Server
HTML Doc
Later...
Our Code
Page 1
https://www.draw.io

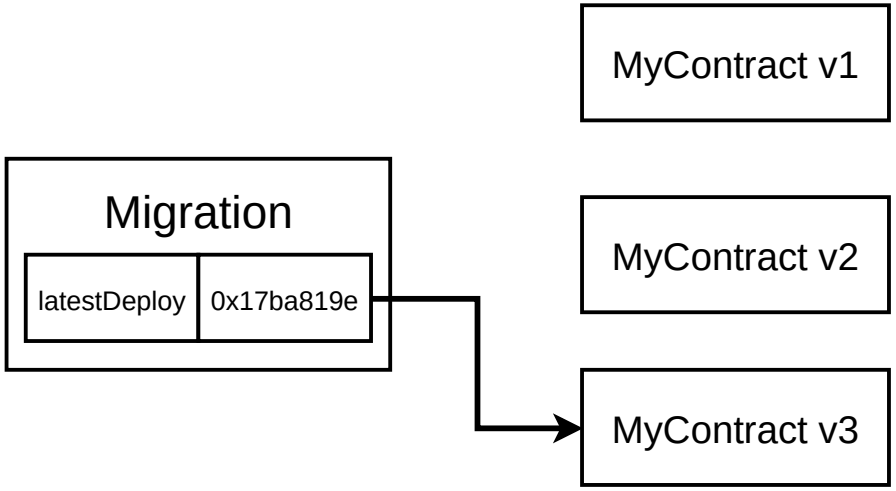

Project Directory
.next
pages
index.js
components
Layout.js

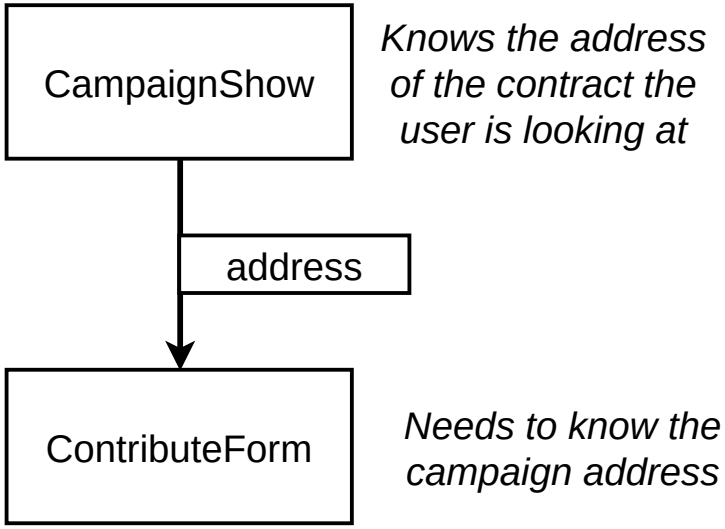

NextJS
Webpack
index.js
web3.js
CampaignFactory.json

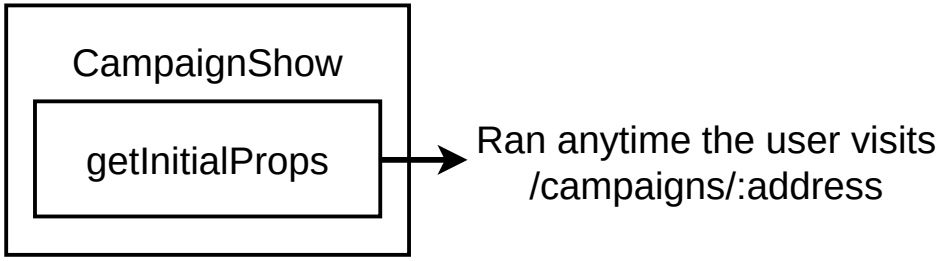

Page 1
https://www.crowdcoin.com/
CrowdCoin
Campaigns
+
Open Campaigns
0x320AF40703106c196eFA39fF86564A94a4B6628c
View Campaign
0xbb6d6f4f875D30b13a134570529125e00B437B65
View Campaign
Create Campaign
Campaign List should be a 'child' of Layout
CrowdCoin
Campaigns
+

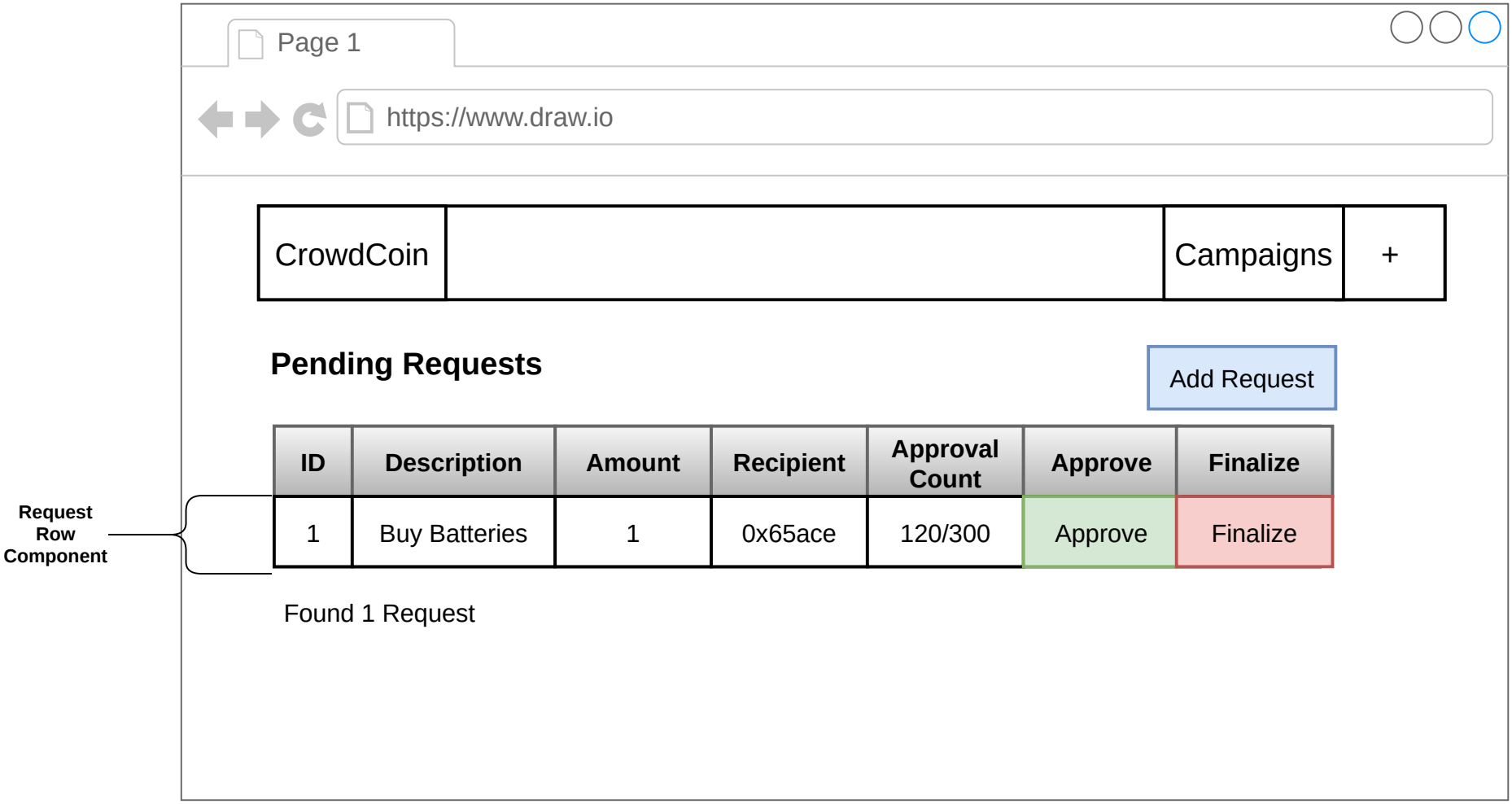

Page 1
https://www.draw.io
CrowdCoin
Campaigns
+
You do not have a valid Ethereum account. Please install Metamask
Create

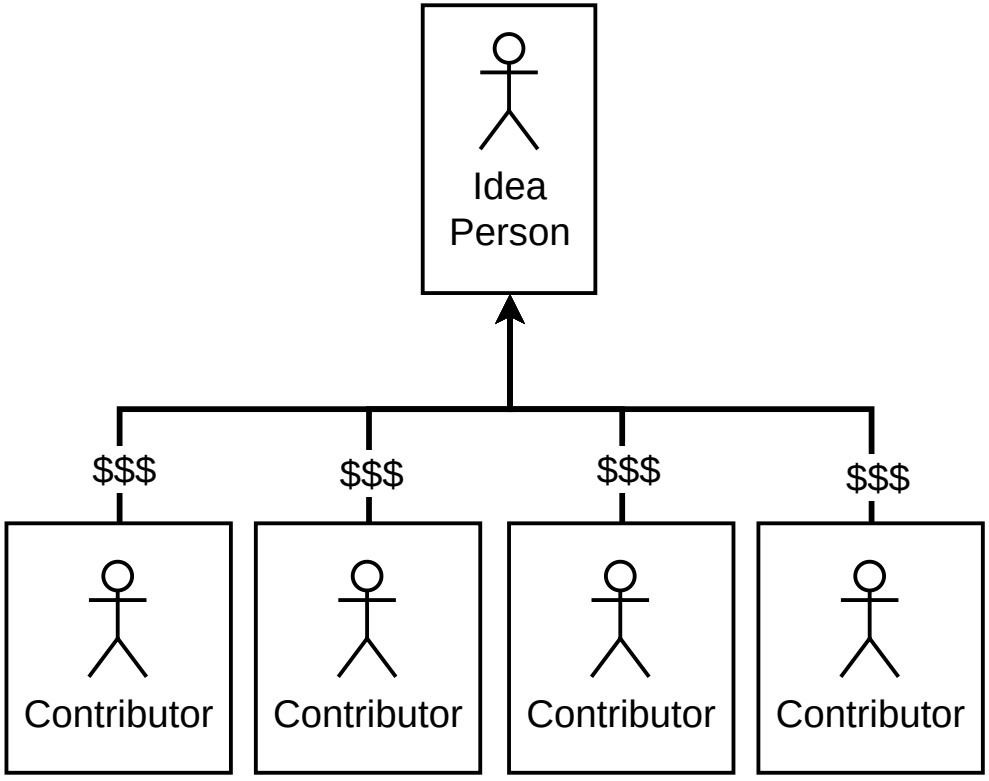

CampaignNew
CampaignShow
server.js
routes.js
Boot up next app, tell it to use routes.js
Defines our different routes

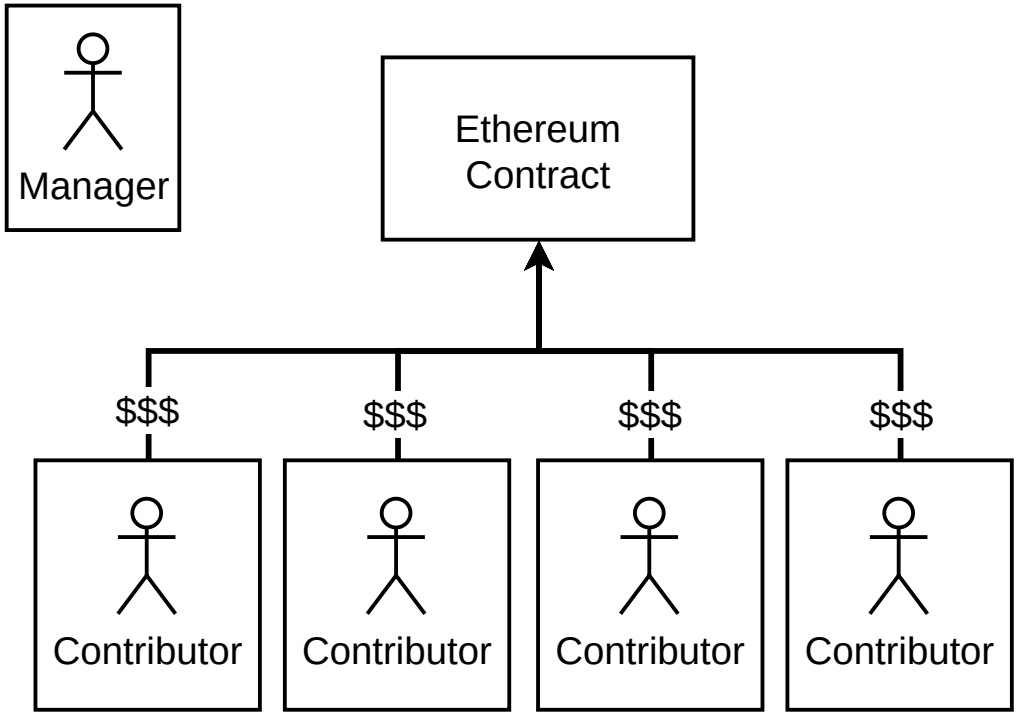

MyContract v1
Migration
latestDeploy
0x17ba819e
MyContract v2
MyContract v3

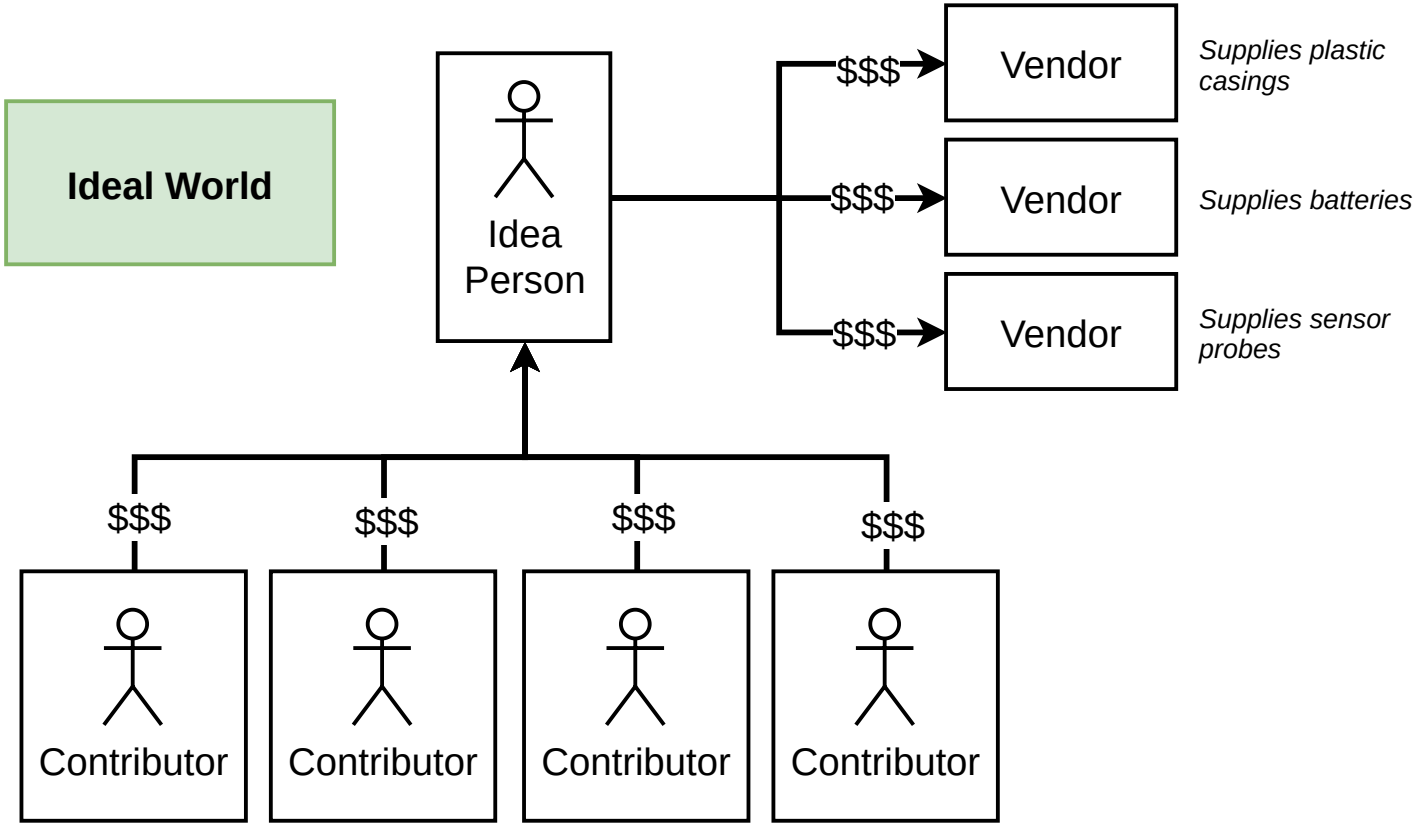

CampaignShow
Knows the address of the contract the user is looking at
address
ContributeForm
Needs to know the campaign address

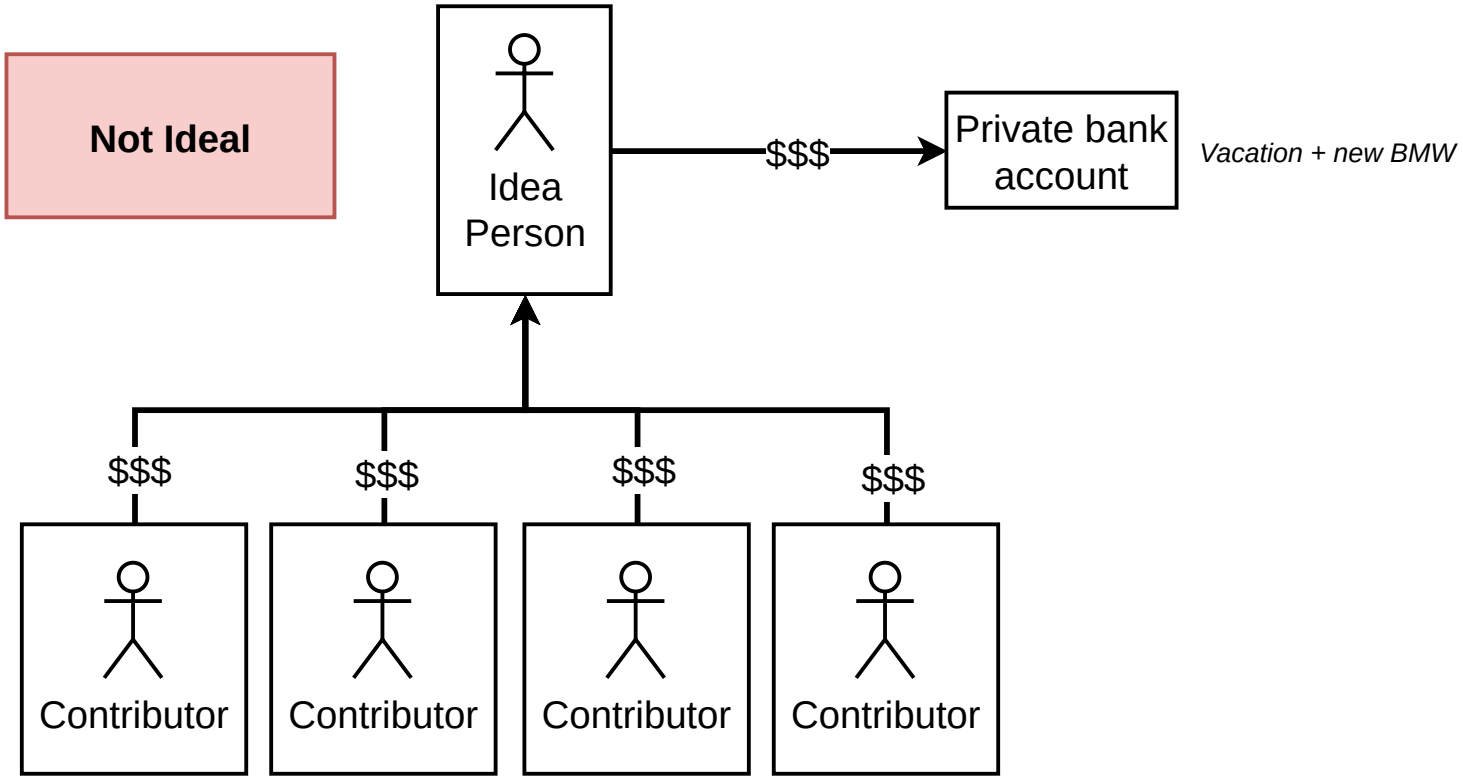

CampaignShow
getInitialProps
Ran anytime the user visits
/campaigns/:address

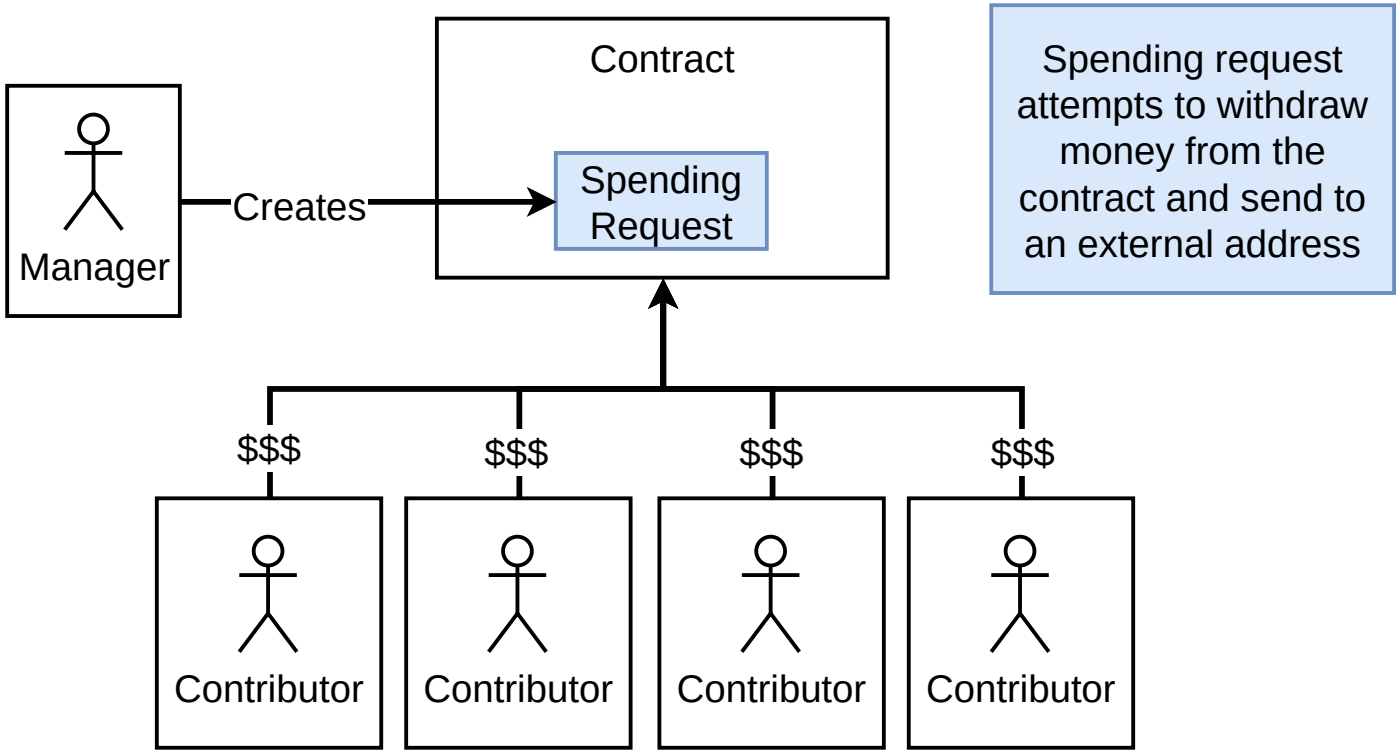

Page 1
https://www.draw.io
CrowdCoin
Campaigns
+
Pending Requests
Add Request
ID
Description
Amount
Recipient
Approval Count
Approve
Finalize
1
Buy Batteries
1
0x65ace
120/300
Approve
Finalize
Found 1 Request
Request Row Component

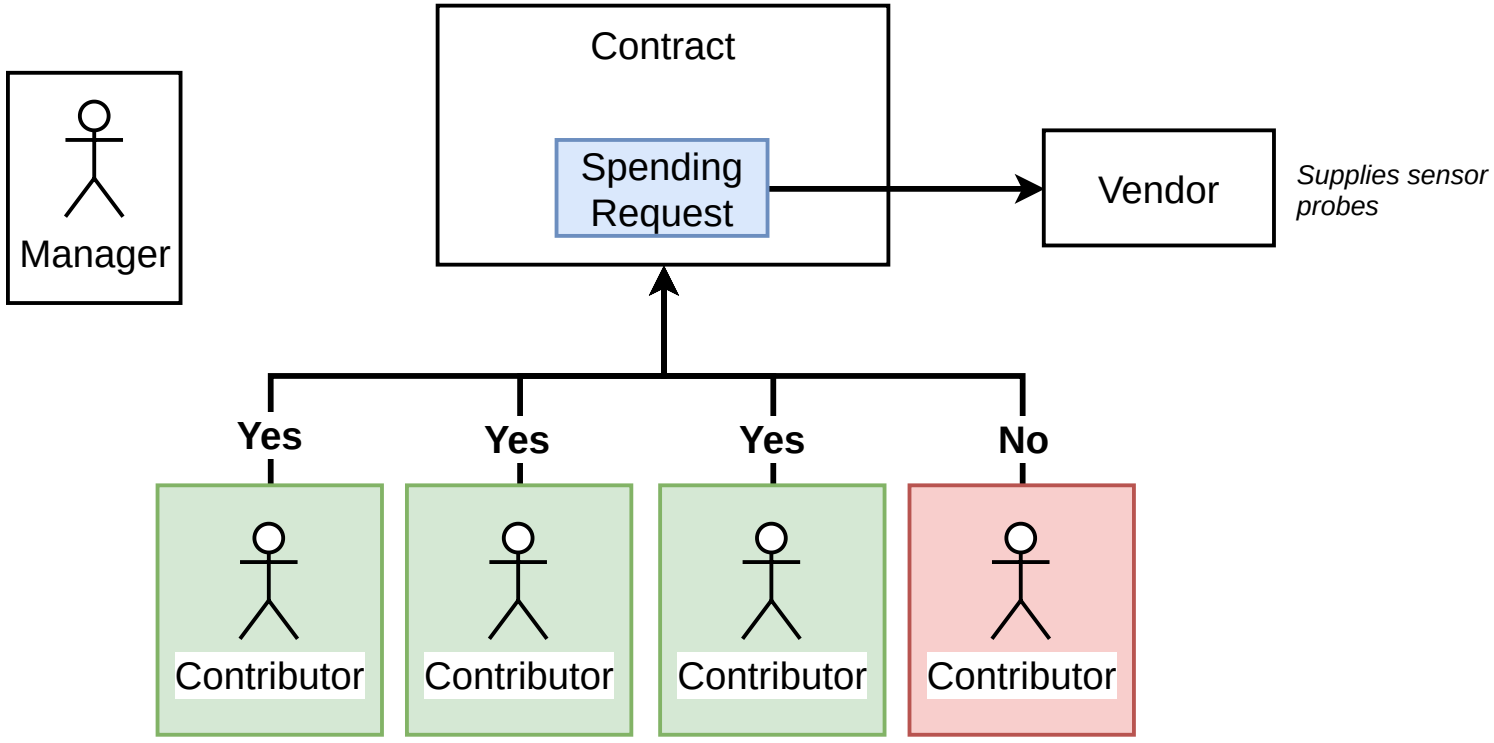

Idea
Person
$$$
$$$
$$$
$$$
Contributor
Contributor
Contributor
Contributor


Manager
Ethereum
Contract
$$$
$$$
$$$
$$$
Contributor
Contributor
Contributor
Contributor


Ideal World
Idea Person
$$$
Vendor
Supplies plastic casings
$$$
Vendor
Supplies batteries
$$$
Vendor
Supplies sensor probes
$$$
$$$
$$$
$$$
Contributor
Contributor
Contributor
Contributor


Not Ideal
Idea Person
$$$
Private bank account
Vacation + new BMW
$$$
$$$
$$$
$$$
Contributor
Contributor
Contributor
Contributor


Contract
Spending Request
Manager
Creates
Spending request attempts to withdraw money from the contract and send to an external address
$$$
$$$
$$$
$$$
Contributor
Contributor
Contributor
Contributor


Contract
Spending Request
Vendor
Supplies sensor probes
Manager
Yes
Yes
Yes
No
Contributor
Contributor
Contributor
Contributor

Campaign Contract

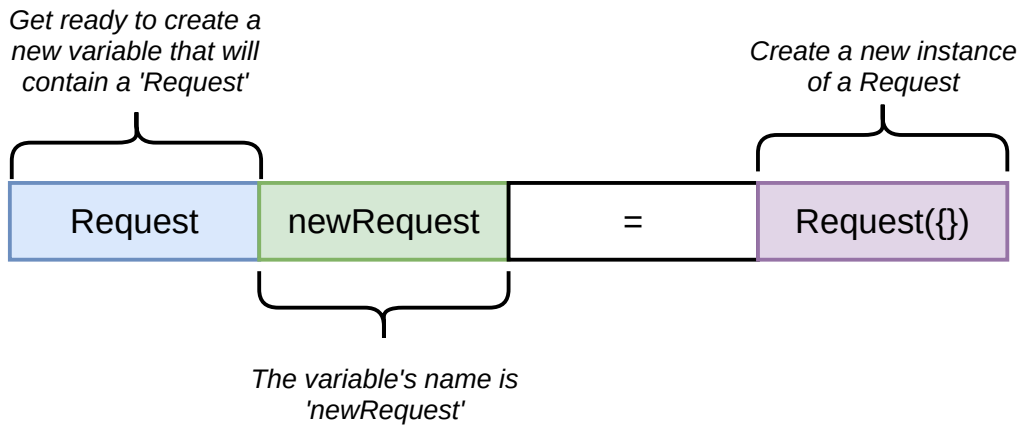
Variables

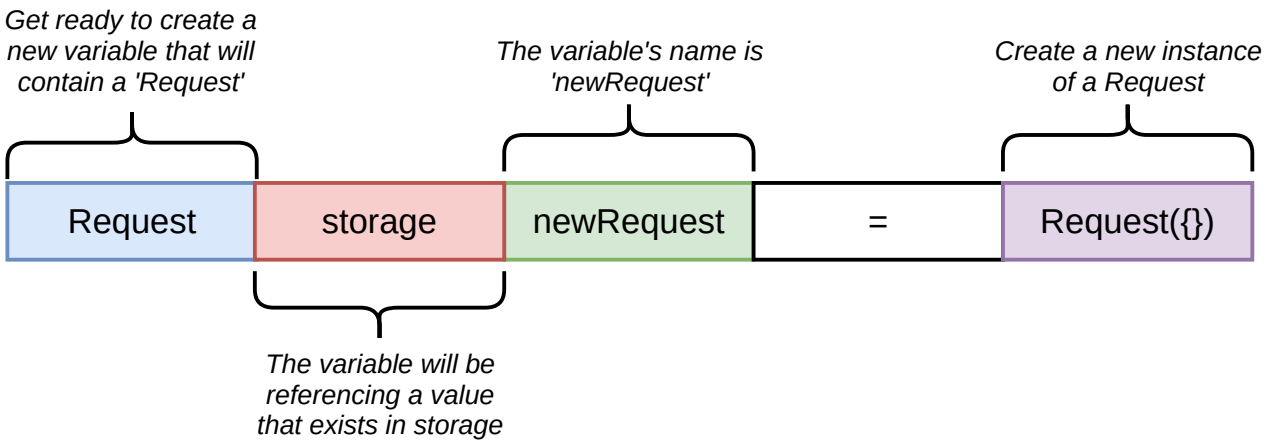
manager	address	address of the person who is managing this campaign	
minimumContribution	uint	Minimum donation required to be considered a contributor or 'approver'	
approvers	address[]	List of addresses for every person who has donated money	!!!!
requests	Request[]	List of requests that the manager has created.	

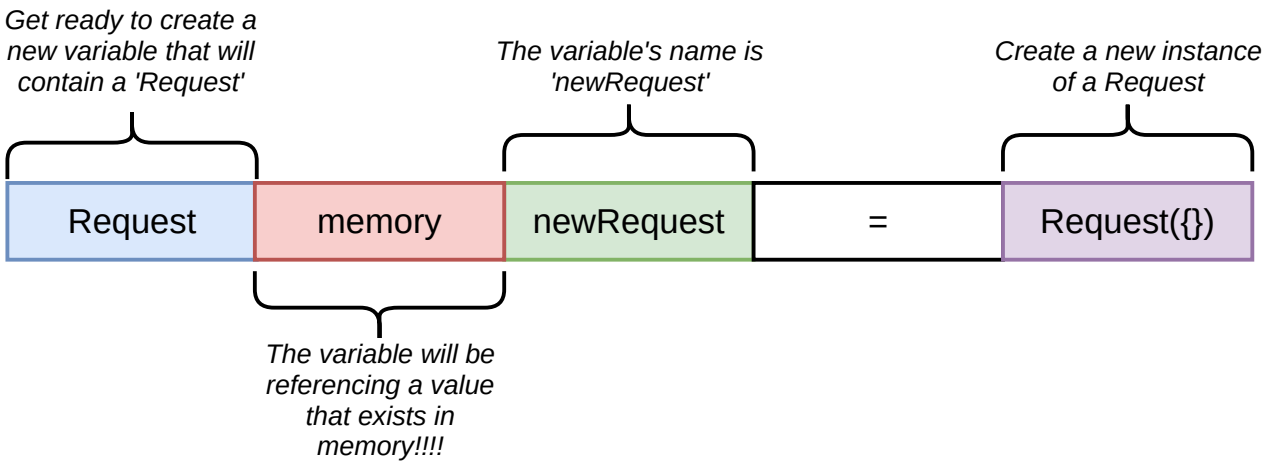
Functions

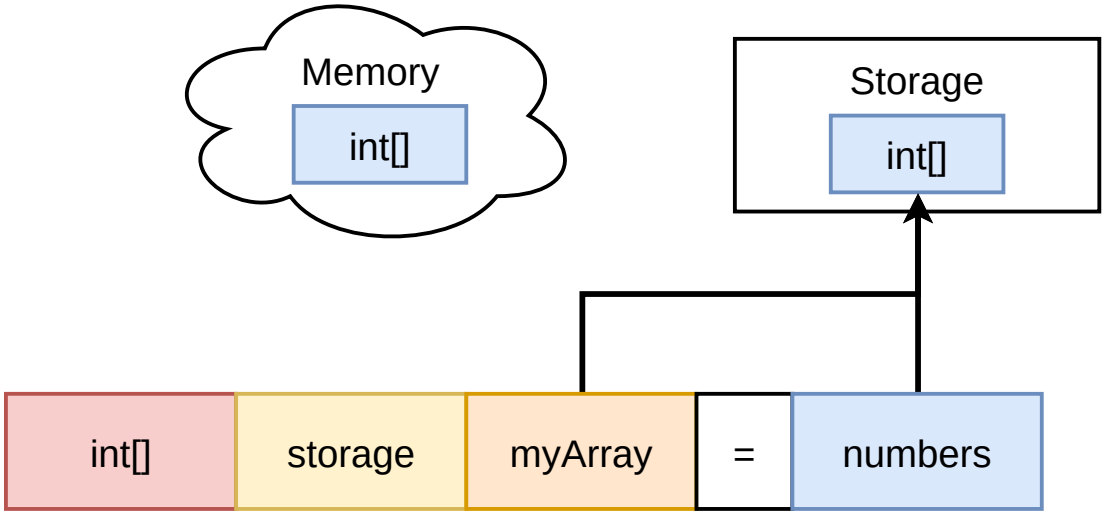
Campaign	Constructor function that sets the minimumContribution and the owner	
contribute	Called when someone wants to donate money to the campaign and become an 'approver'	
createRequest	Called by the manager to create a new 'spending request'	
approveRequest	Called by each contributor to approve a spending request	!!!!
finalizeRequest	After a request has gotten enough approvals, the manager can call this to get money sent to the vendor	

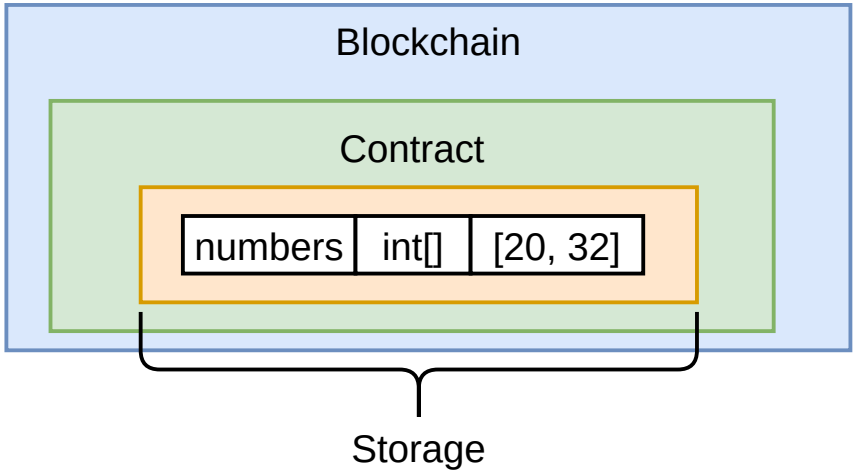
Request Struct		
Name	Type	Purpose
description	string	Describes why the request is being created.
value	uint	Amount of money that the manager wants to send to the vendor
recipient	address	Address that the money will be sent to.
complete	bool	True if the request has already been processed (money sent)
???	???	Voting mechanism!


Get ready to create a new variable that will contain a 'Request'
Create a new instance of a Request
Request
newRequest
=
Request({})
The variable's name is 'newRequest'

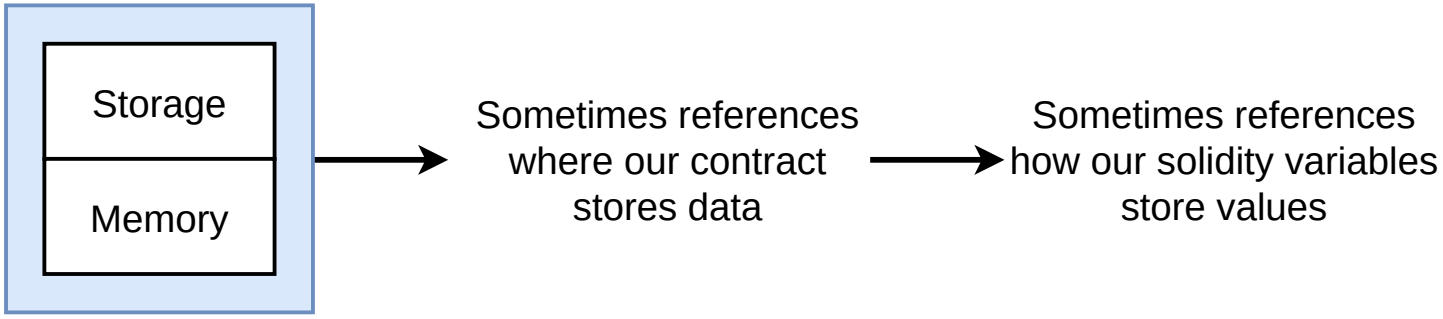

Get ready to create a new variable that will contain a 'Request'
The variable's name is 'newRequest'
Create a new instance of a Request
Request
storage
newRequest
=
Request({})
The variable will be referencing a value that exists in storage

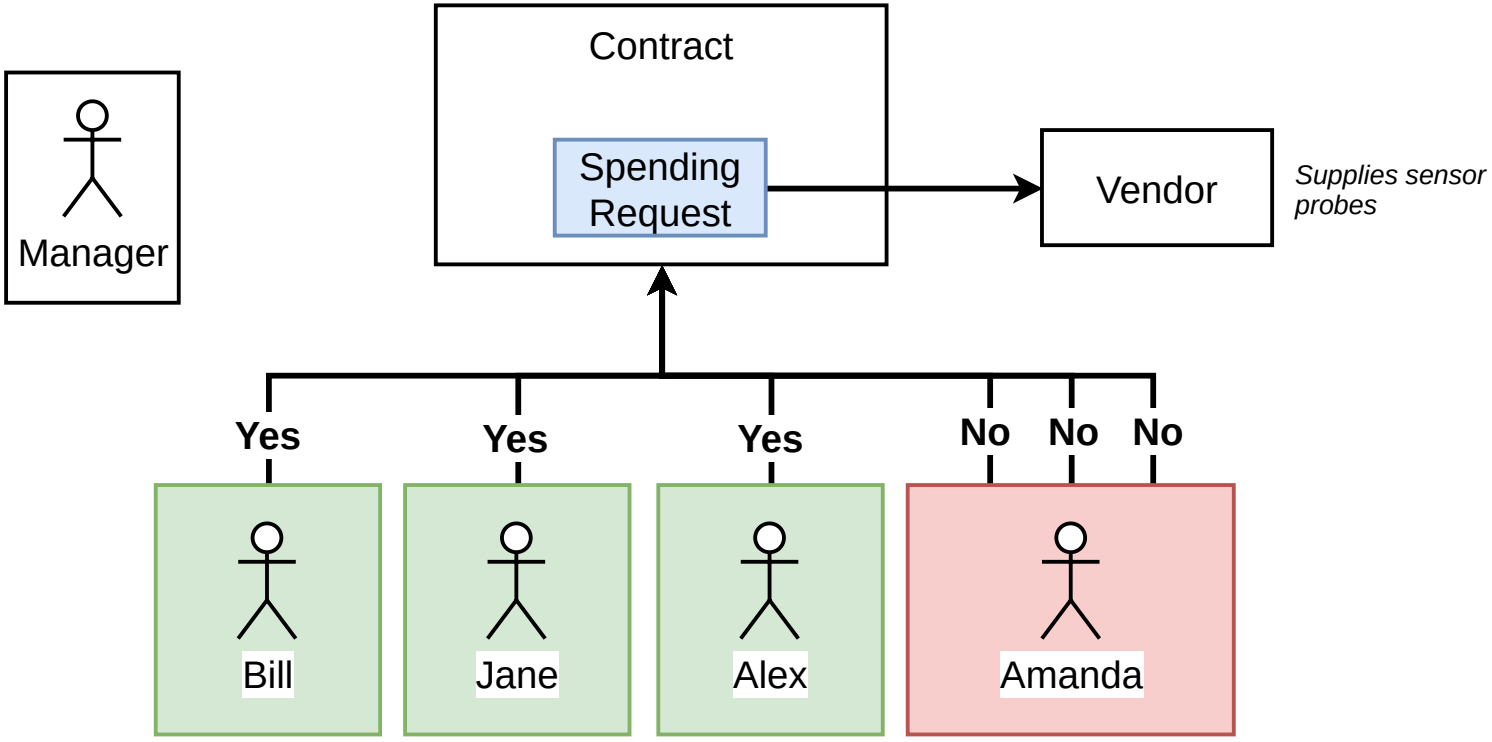

Get ready to create a new variable that will contain a 'Request'
The variable's name is 'newRequest'
Create a new instance of a Request
Request
memory
newRequest
=
Request({})
The variable will be referencing a value that exists in memory!!!!

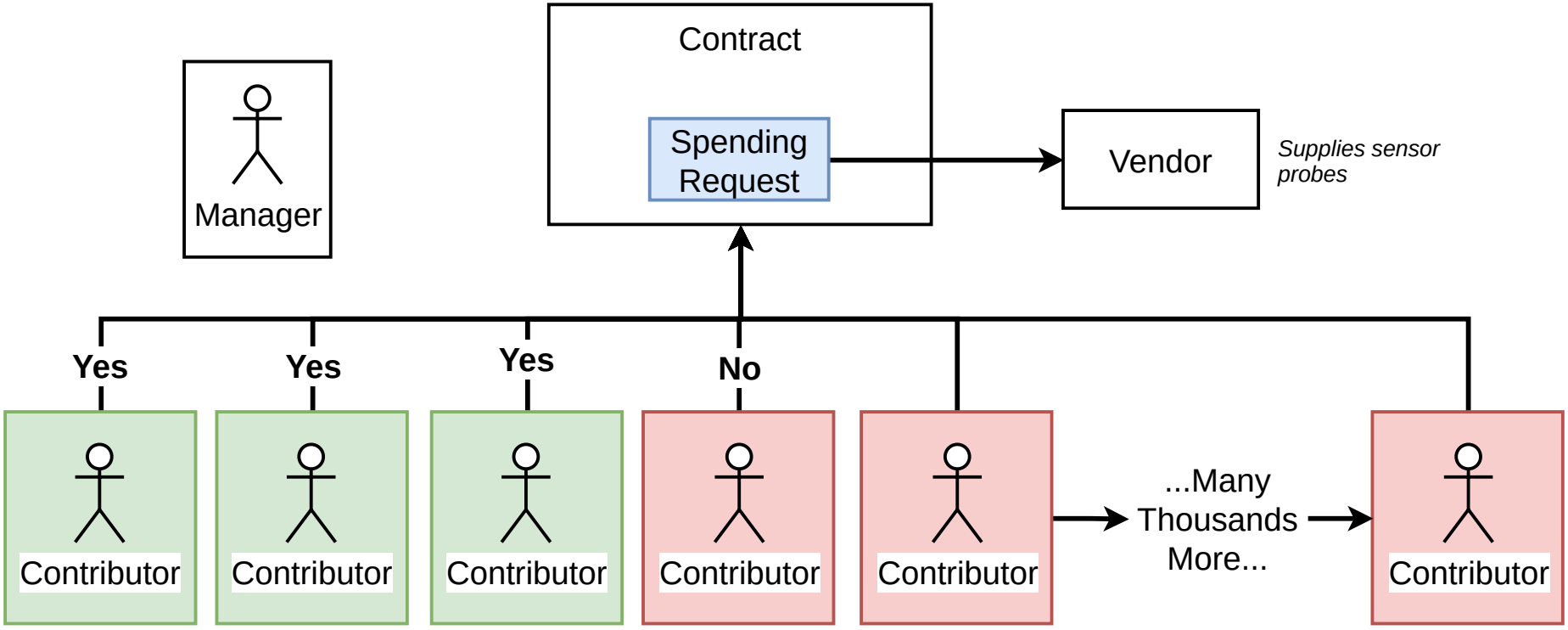

Memory
int[]
Storage
int[]
int[]
storage
myArray
=
numbers

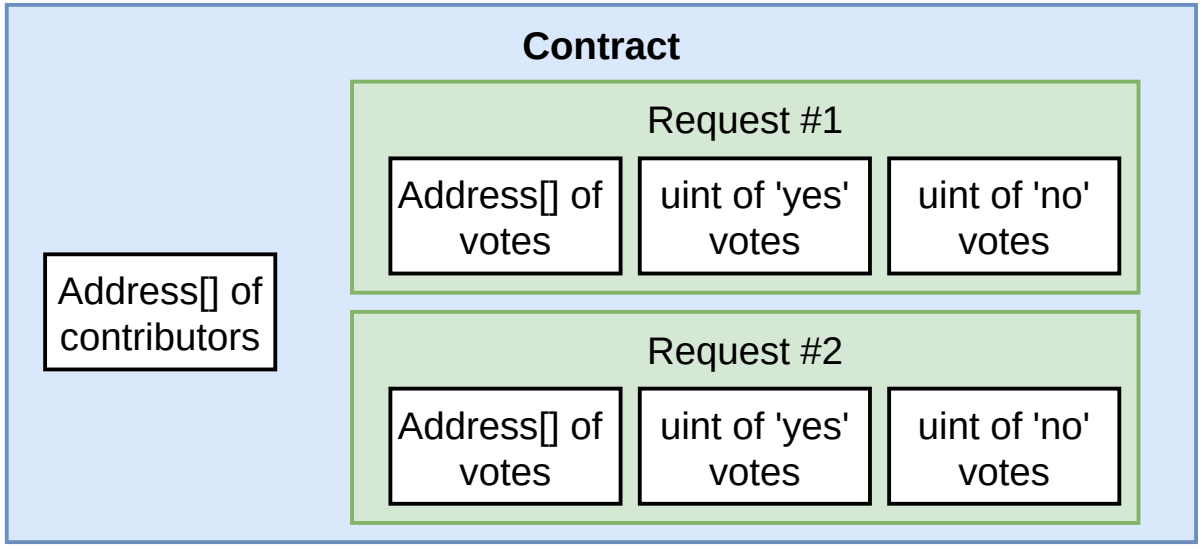

Blockchain
Contract
numbers
int[]
[20, 32]
Storage

Data Holding Places		
Storage	Holds data between function calls	*Pretty much like a computer's hard drive*
Memory	Temporary place to store data	*Pretty much like a computer's RAM*


Storage
Memory
Sometimes references where our contract stores data
Sometimes references how our solidity variables store values


Contract
Spending Request
Vendor
Supplies sensor probes
Manager
Yes
Yes
Yes
No
No
No
Bill
Jane
Alex
Amanda


Contract
Spending Request
Vendor
Supplies sensor probes
Manager
Yes
Yes
Yes
No
Contributor
Contributor
Contributor
Contributor
Contributor
...Many Thousands More...
Contributor


Contract
Request #1
Address[] of votes
uint of 'yes' votes
uint of 'no' votes
Address[] of contributors
Request #2
Address[] of votes
uint of 'yes' votes
uint of 'no' votes

Costs 10,000 gas per person

Costs 5,000 gas per person

```
function approveRequest(Request request) public {
    // Make sure person calling this function has donated
    bool isApprover = false;
    for (uint i = 0; i < approvers.length; i++) {
        if (approvers[i] == msg.sender) {
            isApprover = true;
        }
    }
    require(isApprover);

    // Make sure person calling this function hasn't voted before
    for (uint i = 0; i < request.approvers.length; i++) {
        require(approvers != msg.sender);
    }
}
```

For 1 Contributor

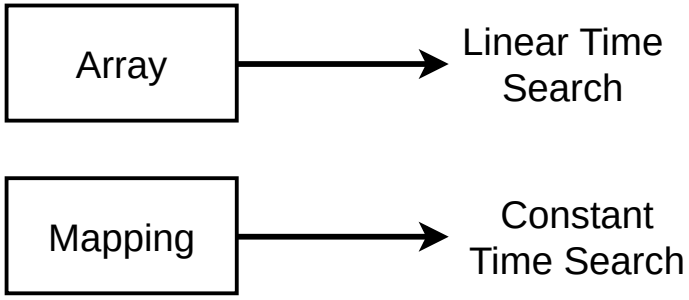
Checking is the caller is a member of our campaign		Checking if the caller has voted before	
10,000 gas * 1 contributor	+	5,000 gas * 1 contributor	**= 15,000 gas**

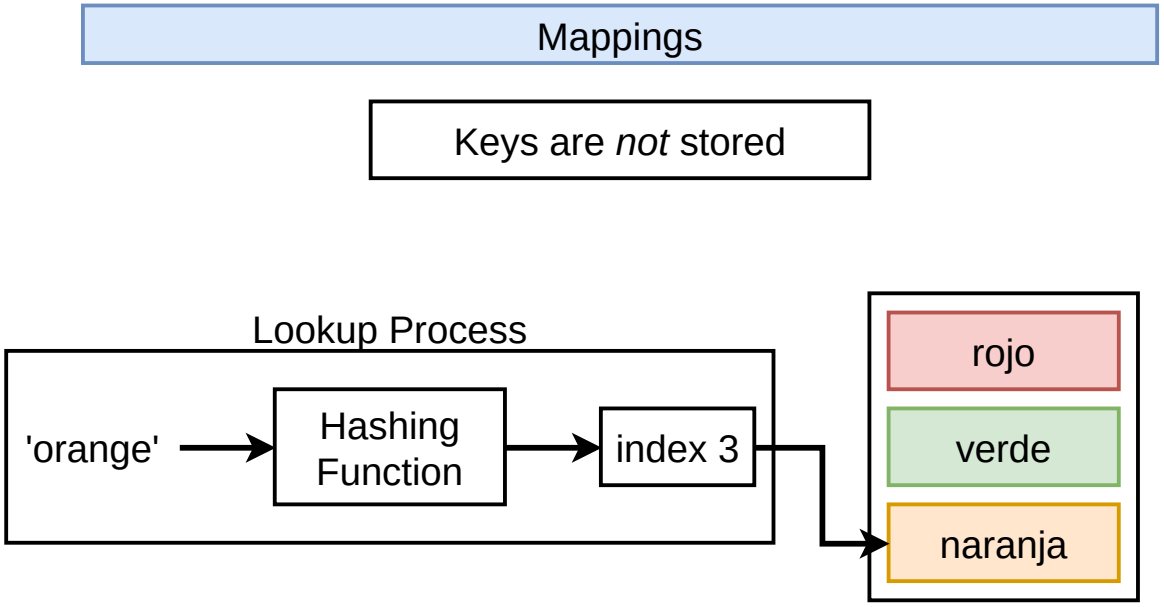
For 10,000 Contributors

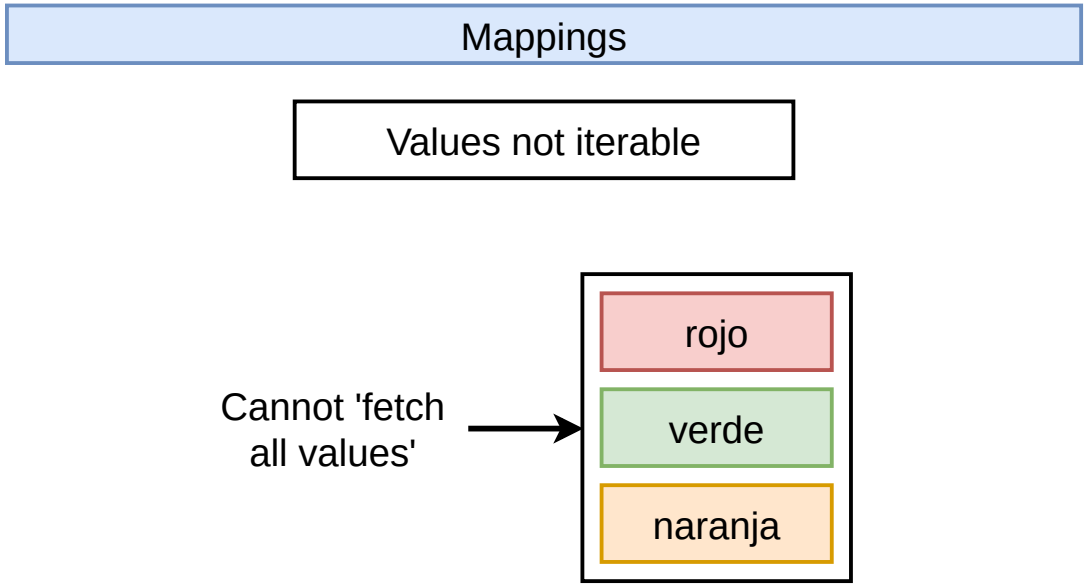
Checking is the caller is a member of our campaign	Checking if the caller has voted before

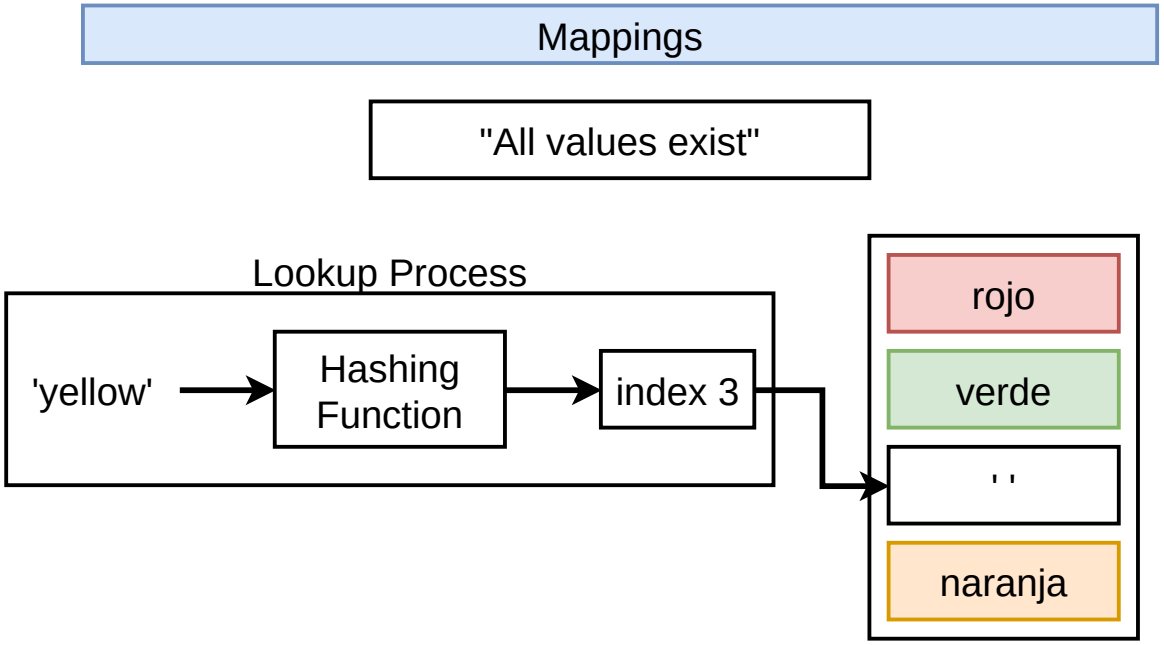
10,000 gas * 10,000 contributors + 5,000 gas * 10,000 contributor **= 100,015,000 gas**

Search Time




Mappings
Keys are *not* stored
Lookup Process
'orange'
Hashing Function
index 3
rojo
verde
naranja


Mappings
Values not iterable
Cannot 'fetch all values'
rojo
verde
naranja


Mappings
"All values exist"
Lookup Process
'yellow'
Hashing Function
index 3
rojo
verde
' '
naranja

Campaign Contract

Variables

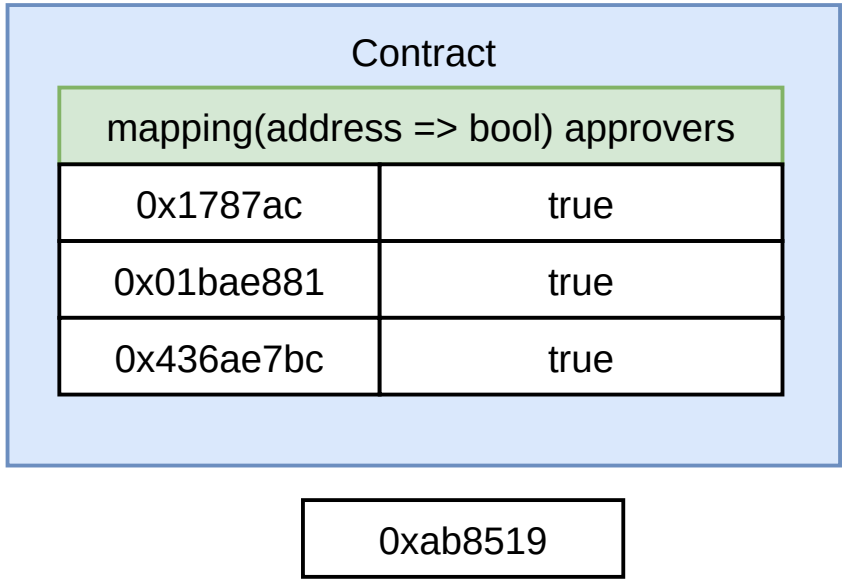
manager	address	address of the person who is managing this campaign
minimumContribution	uint	Minimum donation required to be considered a contributor or 'approver'
approvers	mapping	List of addresses for every person who has donated money
requests	Request[]	List of requests that the manager has created.

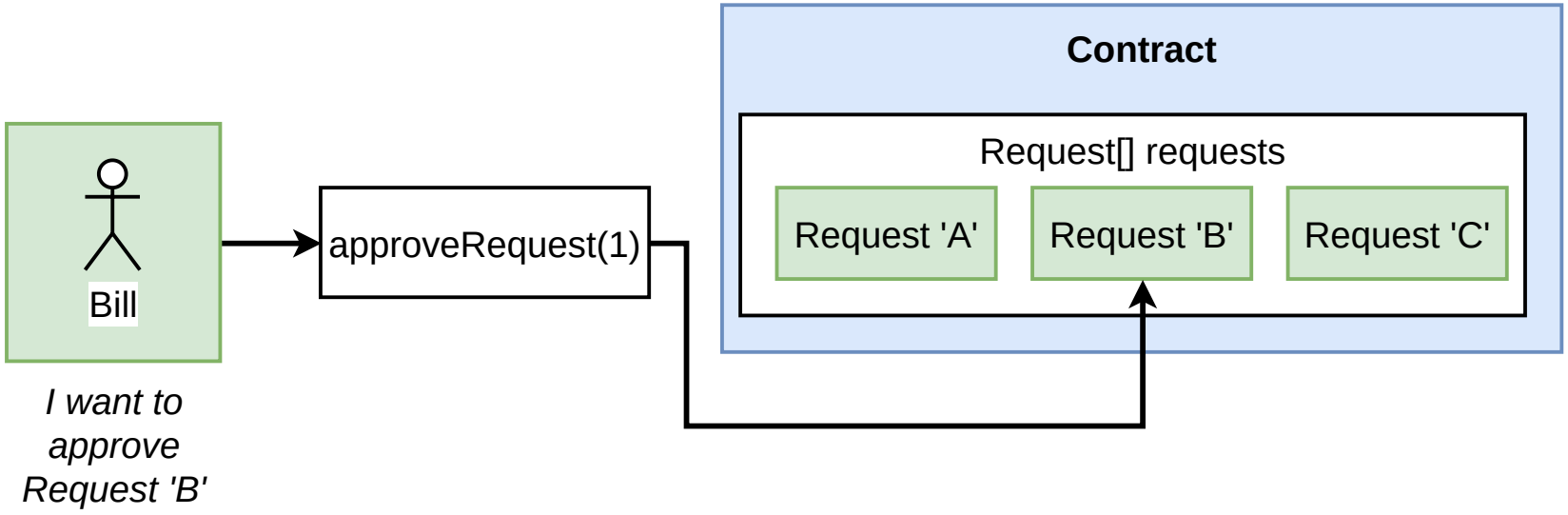
Functions

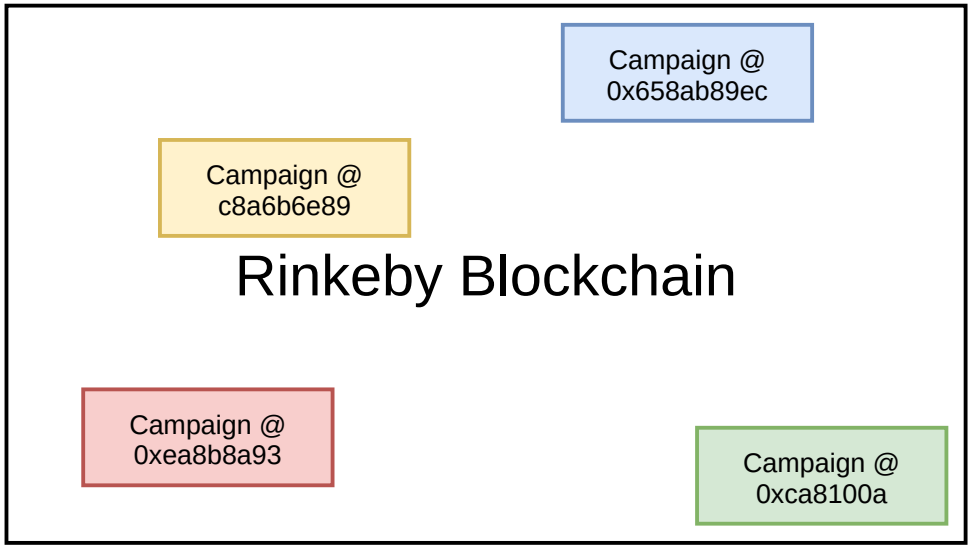
Campaign	Constructor function that sets the minimumContribution and the owner
contribute	Called when someone wants to donate money to the campaign and become an 'approver'
createRequest	Called by the manager to create a new 'spending request'
approveRequest	Called by each contributor to approve a spending request
finalizeRequest	After a request has gotten enough approvals, the manager can call this to get money sent to the vendor

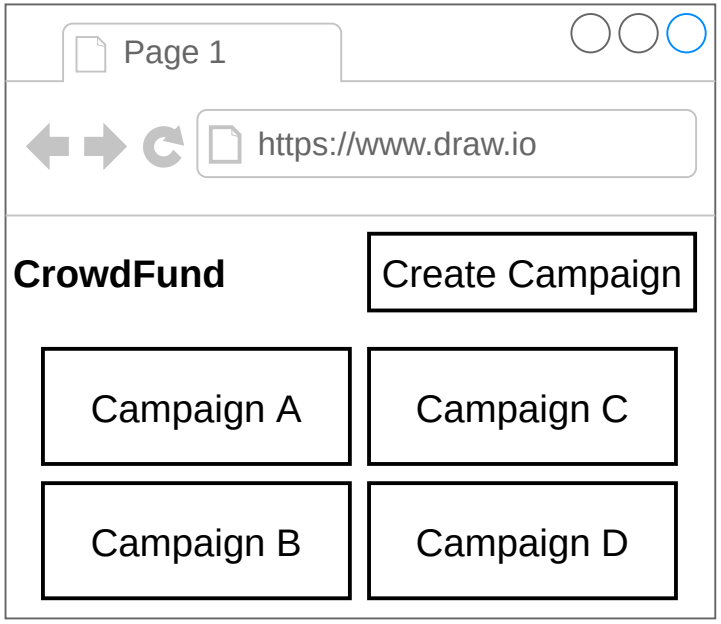
Request Struct

description	string	Purpose of request
amount	uint	Ether to transfer
recipient	address	Who gets the money
complete	bool	Whether the request is done
approvals	mapping	Track who has voted
approvalCount	uint	Track number of approvals

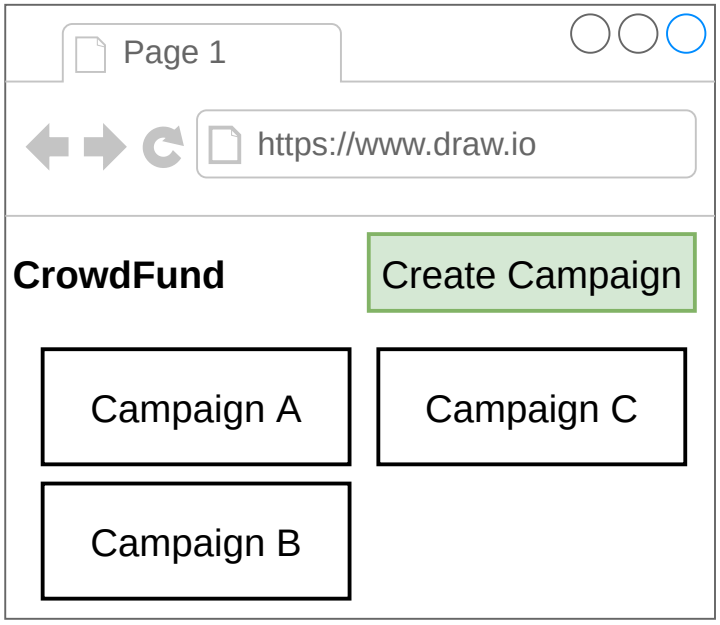

Contract
mapping(address => bool) approvers
0x1787ac
true
0x01bae881
true
0x436ae7bc
true
0xab8519

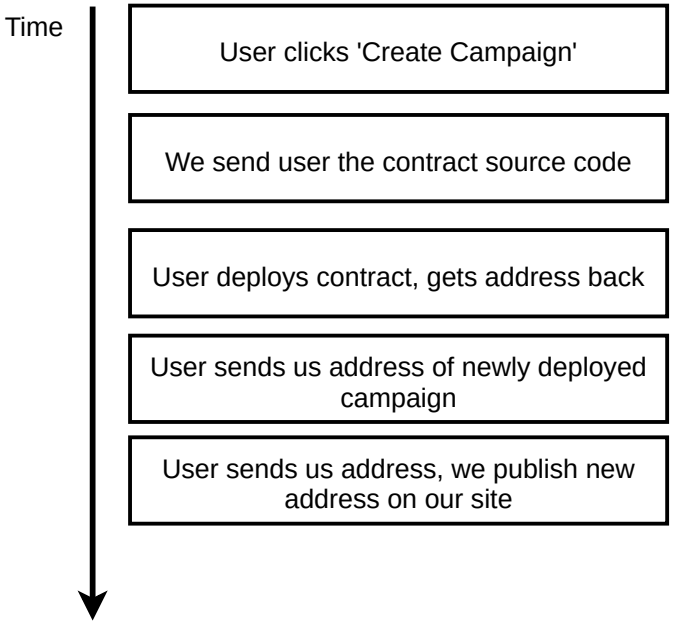

Contract
Request[] requests
Request 'A'
Request 'B'
Request 'C'
approveRequest(1)
Bill
I want to approve Request 'B'


Campaign @
0x658ab89ec
Campaign @
c8a6b6e89
Rinkeby Blockchain
Campaign @
0xea8b8a93
Campaign @
0xca8100a

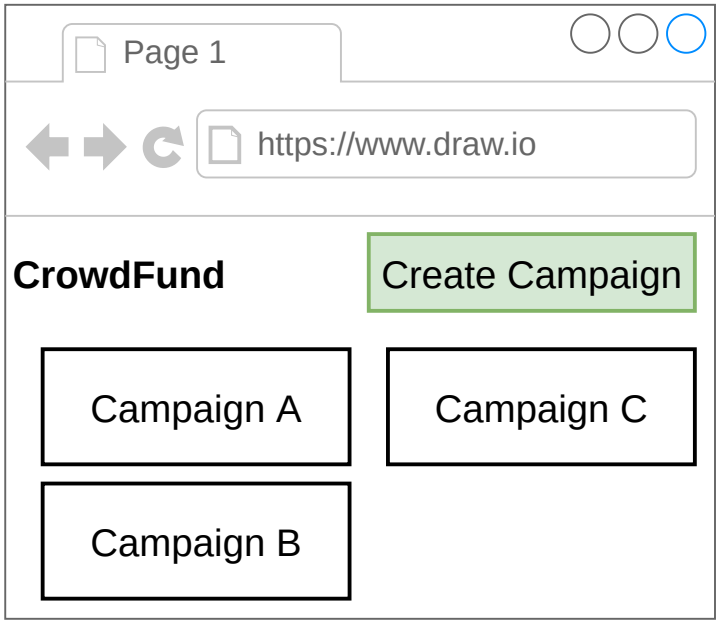

Page 1
https://www.draw.io
CrowdFund
Create Campaign
Campaign A
Campaign C
Campaign B
Campaign D

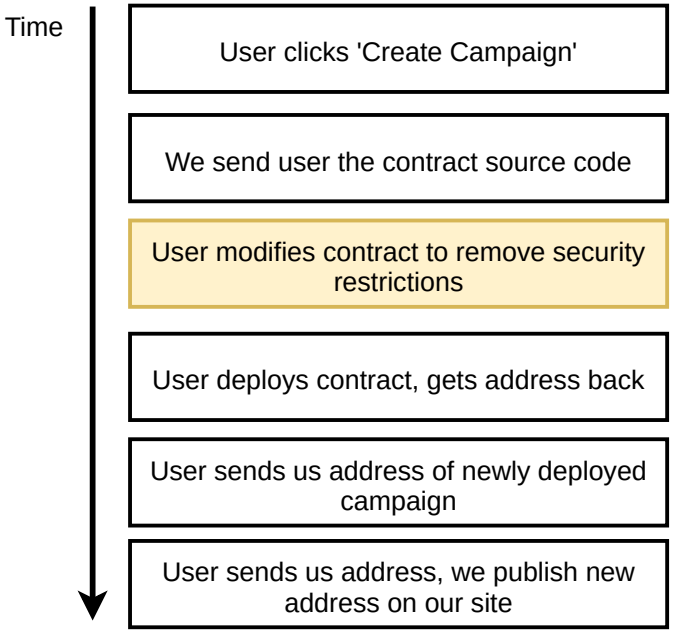
Solution #1


Page 1
https://www.draw.io
CrowdFund
Create Campaign
Campaign A
Campaign C
Campaign B

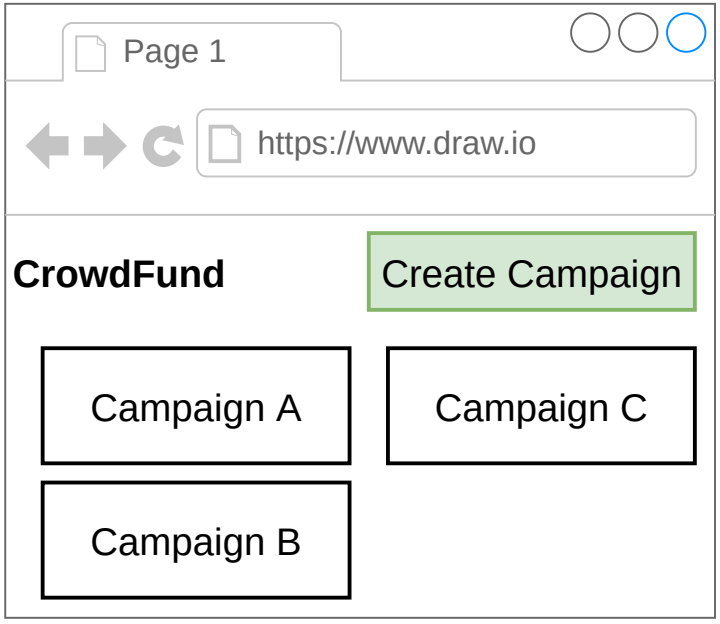

Time
User clicks 'Create Campaign'
We send user the contract source code
User deploys contract, gets address back
User sends us address of newly deployed campaign
User sends us address, we publish new address on our site

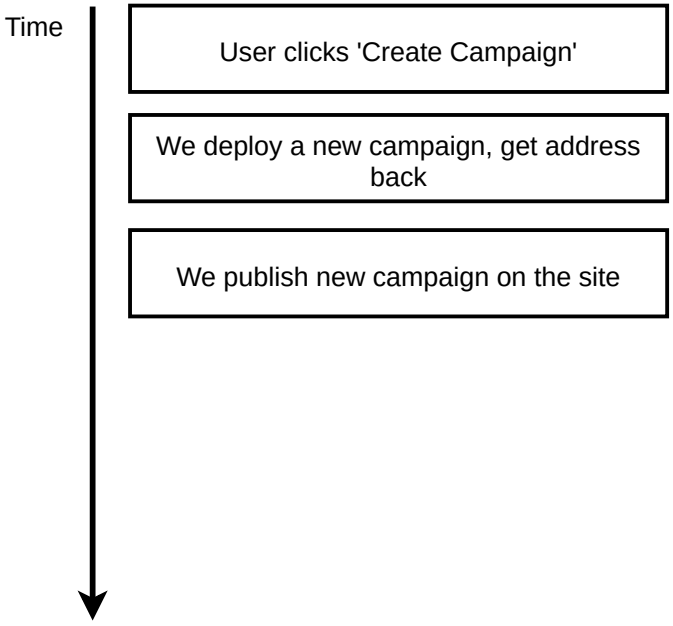
Solution #1


Page 1
https://www.draw.io
CrowdFund
Create Campaign
Campaign A
Campaign C
Campaign B

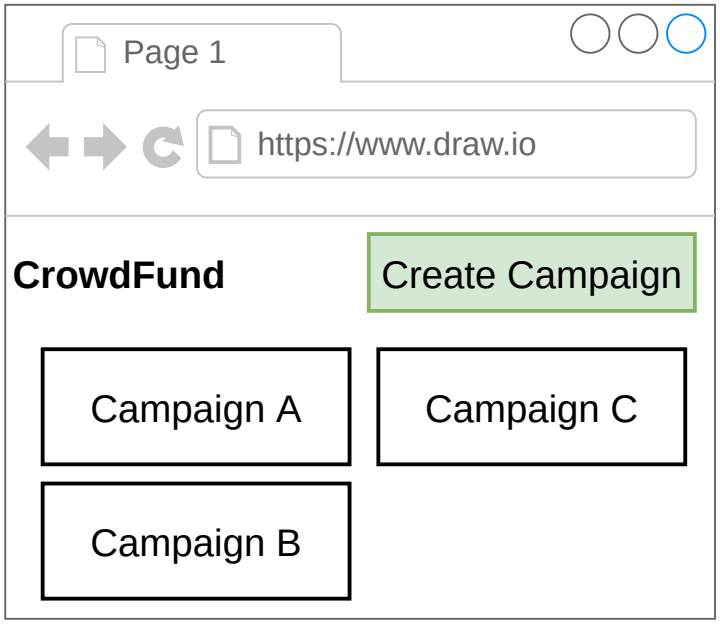

Time
User clicks 'Create Campaign'
We send user the contract source code
User modifies contract to remove security restrictions
User deploys contract, gets address back
User sends us address of newly deployed campaign
User sends us address, we publish new address on our site

Solution #2


Page 1
https://www.draw.io
CrowdFund
Create Campaign
Campaign A
Campaign C
Campaign B


Time
User clicks 'Create Campaign'
We deploy a new campaign, get address back
We publish new campaign on the site

Solution #3



Time

We create a 'factory' contract. It has a function to deploy a new instance of 'Campaign'

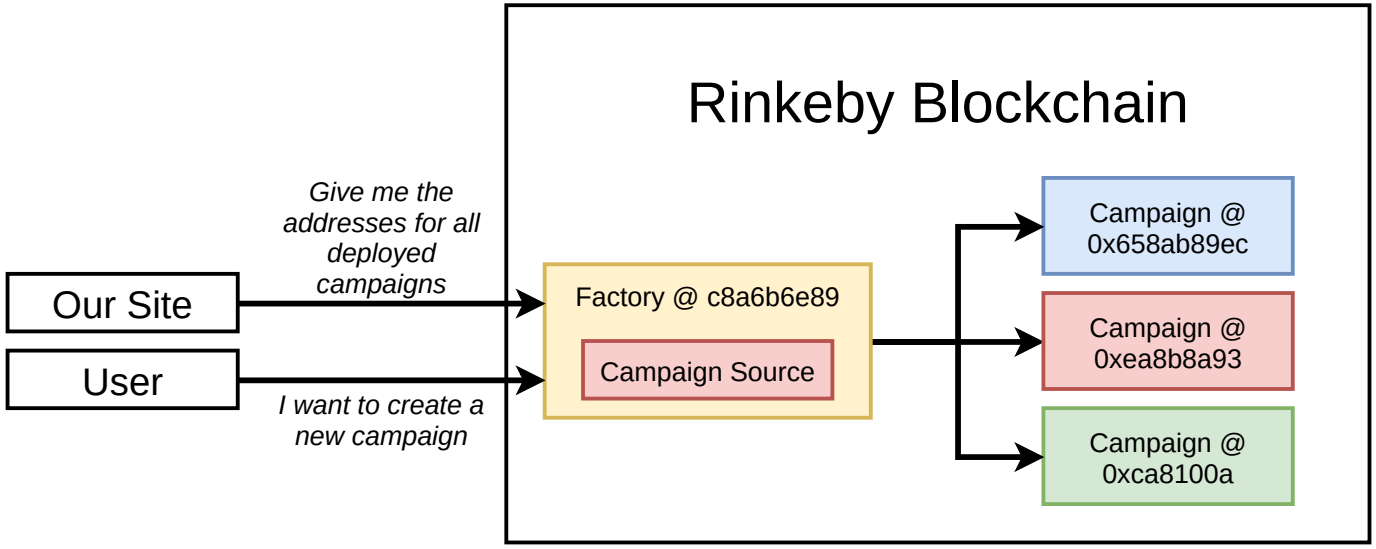
Time passes...

User clicks 'Create Campaign'

We instruct web3/metamask to show user a transaction that invokes 'Campaign Factory'

User pays deployment costs. Factory deploys a new copy of 'Campaign'.

We tell 'Campaign Factory' to give us a list of all deployed campaigns


Rinkeby Blockchain
Give me the addresses for all deployed campaigns
Our Site
User
I want to create a new campaign
Factory @ c8a6b6e89
Campaign Source
Campaign @ 0x658ab89ec
Campaign @ 0xea8b8a93
Campaign @ 0xca8100a

CampaignFactory Contract

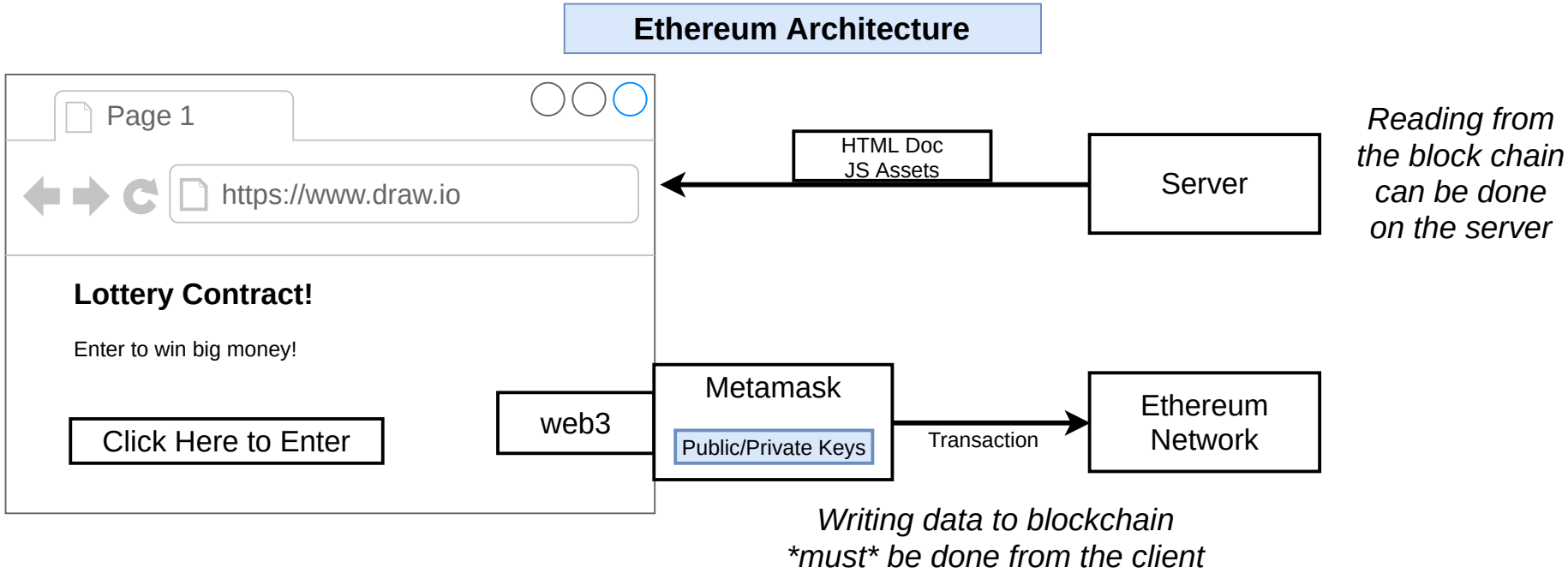
Variables

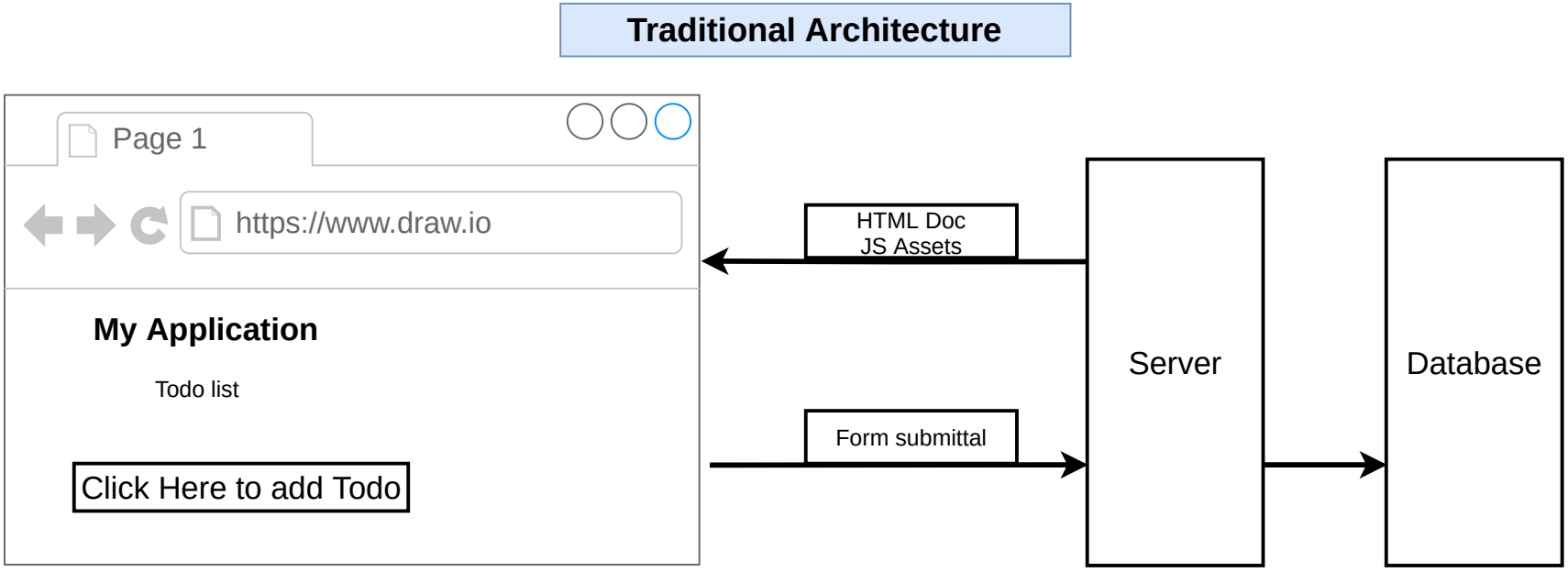
deployedCampaigns	address[]	Addresses of all deployed campaigns

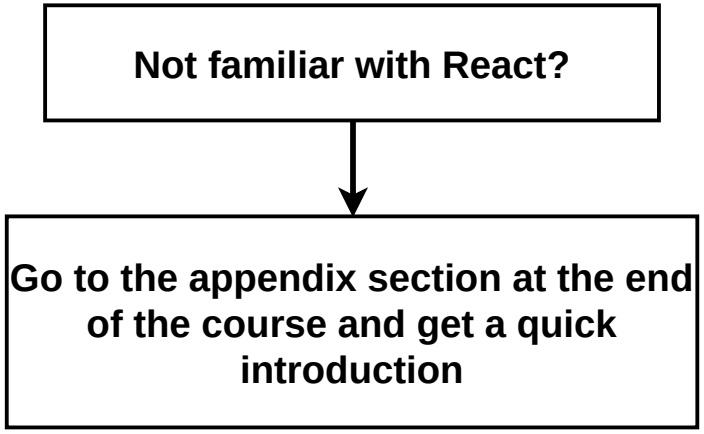
Functions

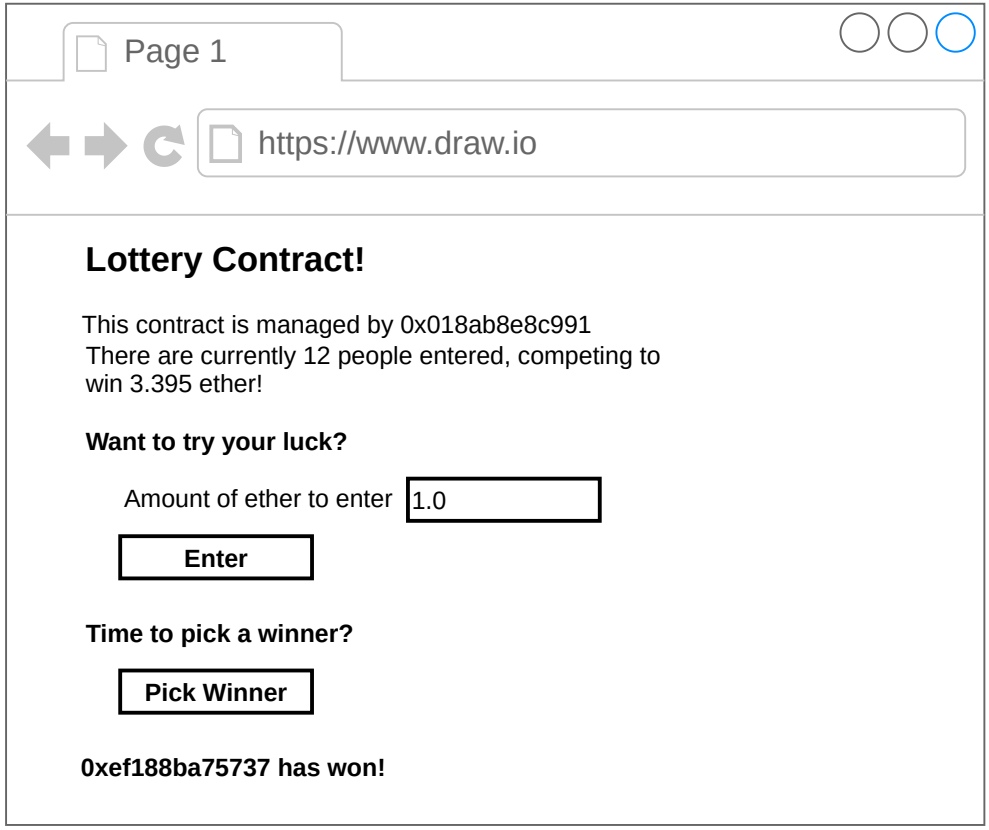
createCampaign	Deploys a new instance of a Campaign and stores the resulting address
getDeployedCampaigns	Returns a list of all deployed campaigns

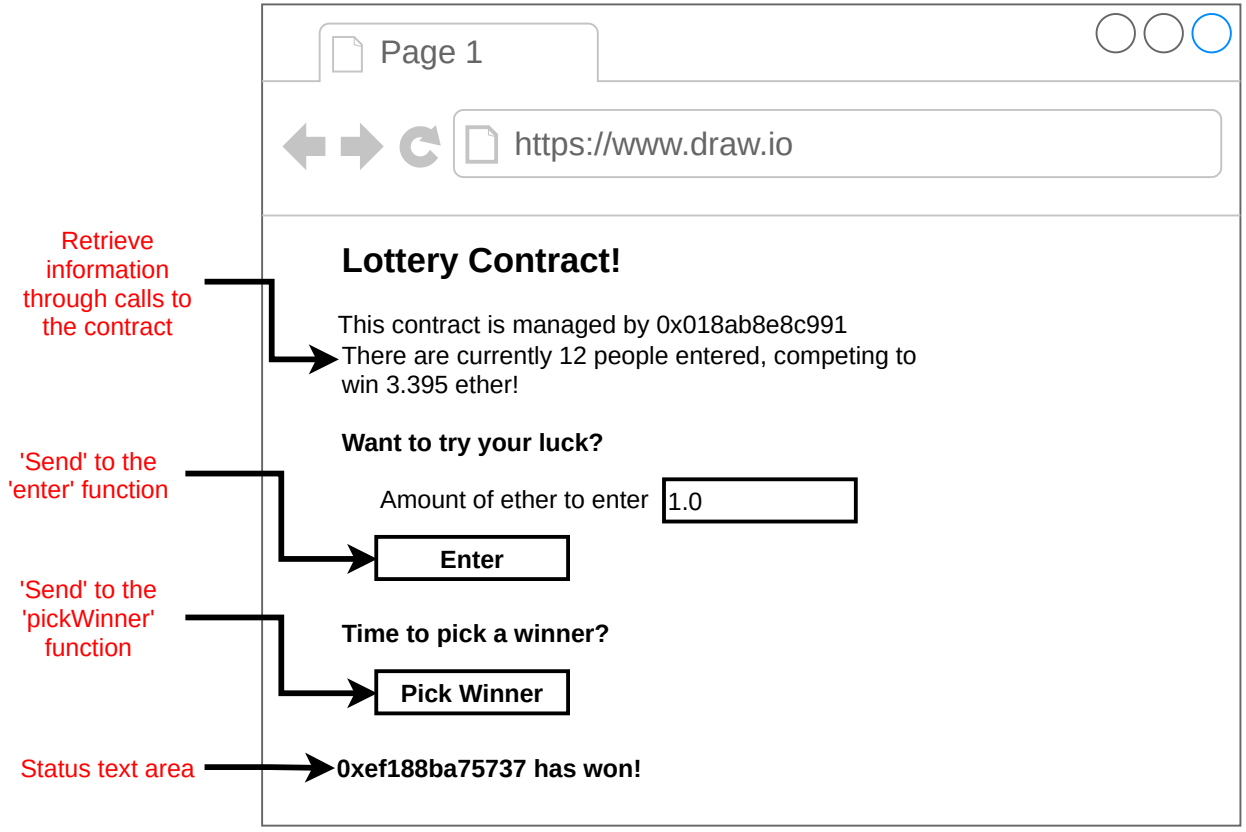
Why React?

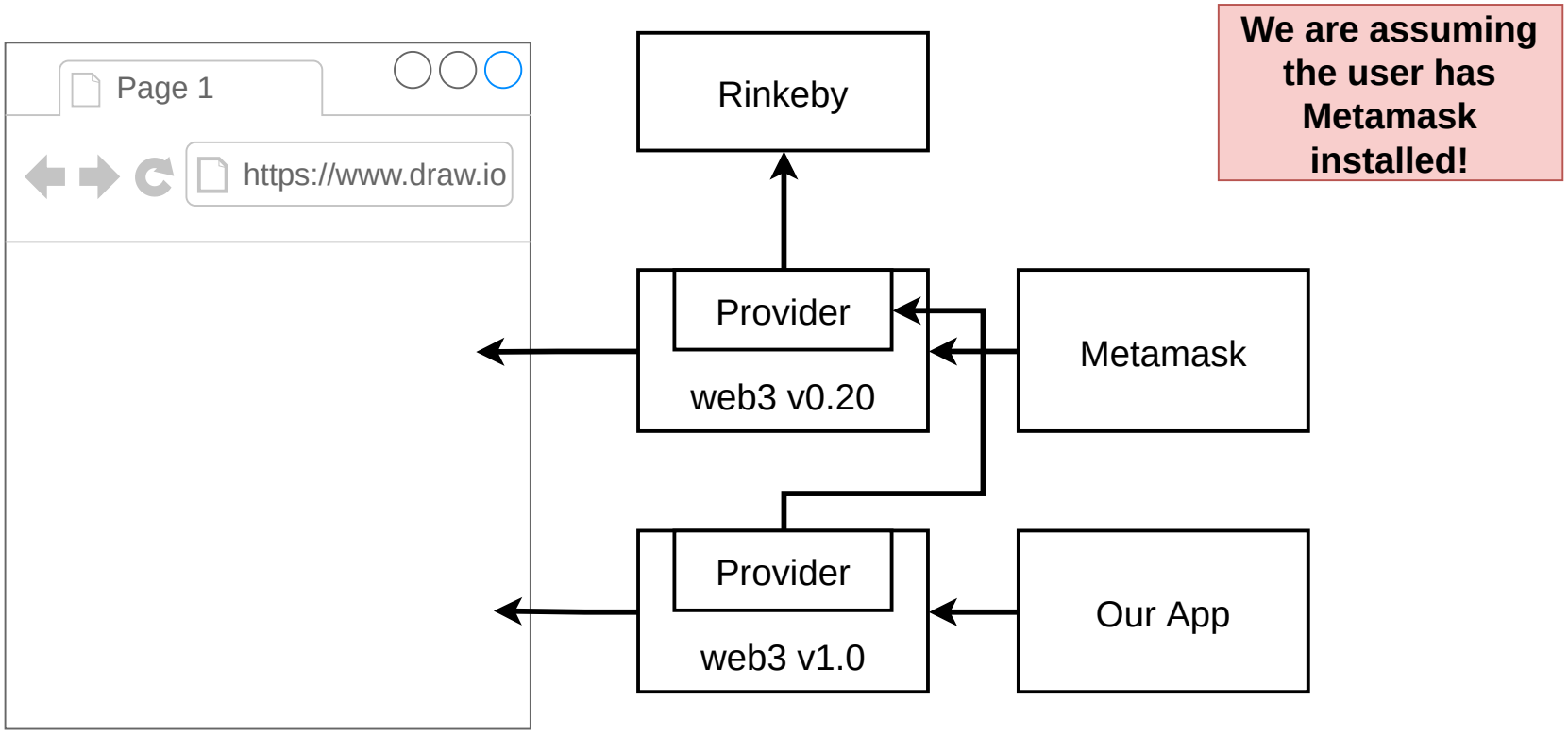

Ethereum Architecture
Page 1
https://www.draw.io
Lottery Contract!
Enter to win big money!
Click Here to Enter
web3
Metamask
Public/Private Keys
HTML Doc
JS Assets
Server
Transaction
Ethereum Network
Reading from the block chain can be done on the server
Writing data to blockchain *must* be done from the client

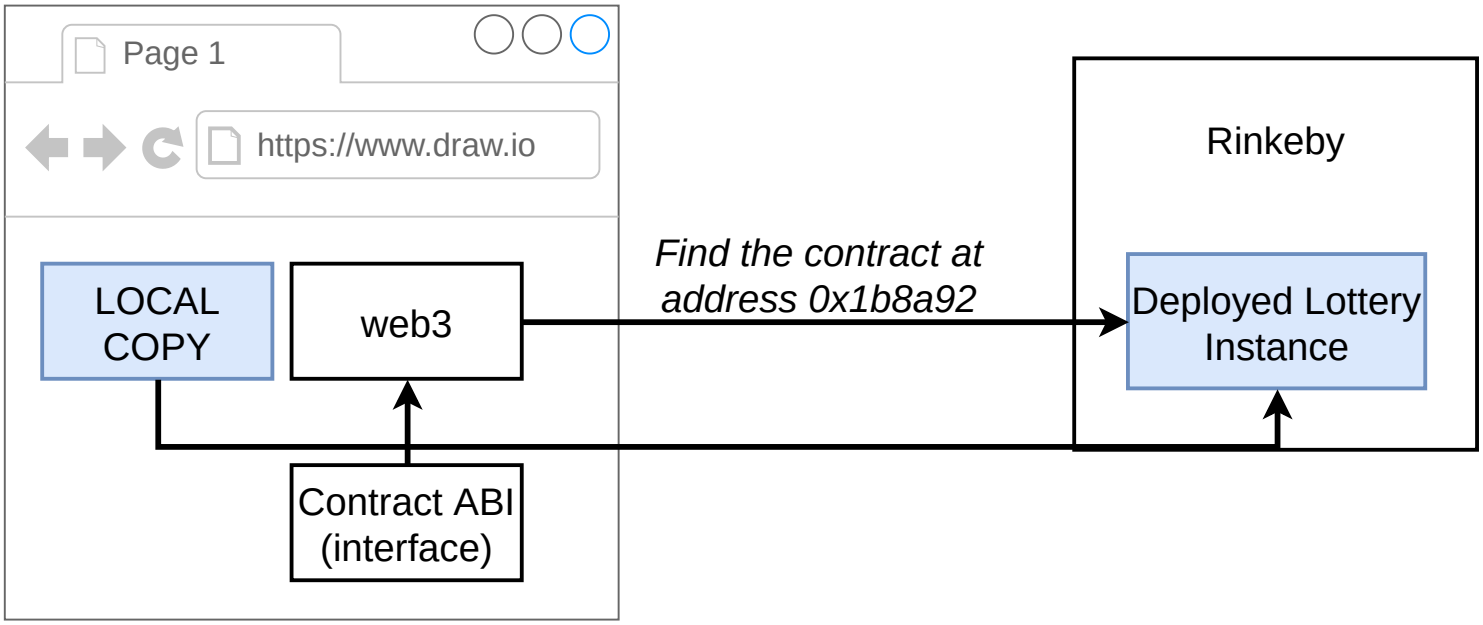

Traditional Architecture
Page 1
https://www.draw.io
My Application
Todo list
Click Here to add Todo
HTML Doc
JS Assets
Form submittal
Server
Database

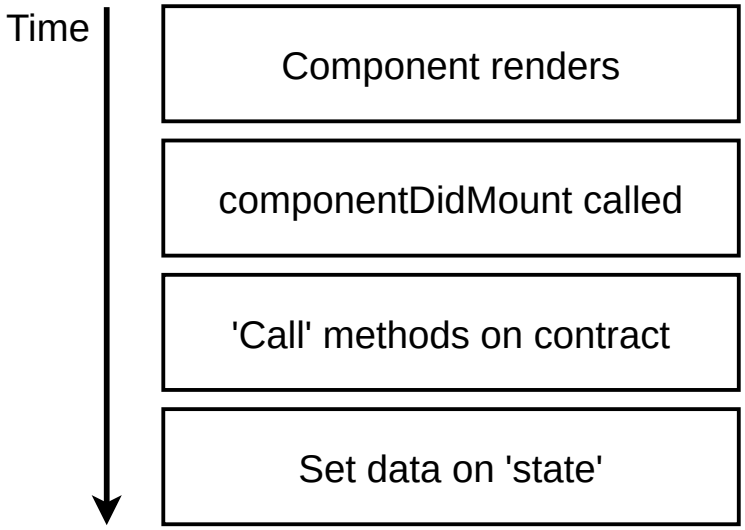

Not familiar with React?
Go to the appendix section at the end of the course and get a quick introduction

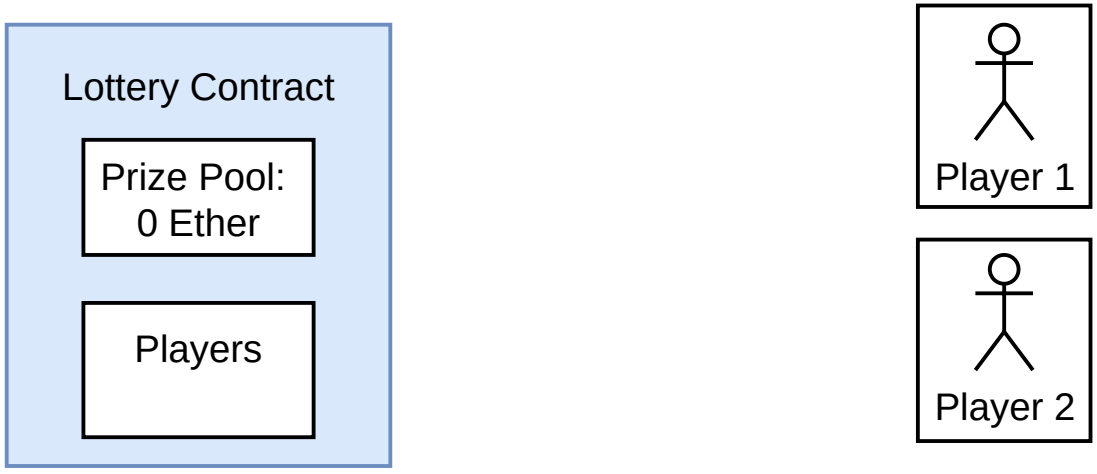

Page 1
https://www.draw.io
Lottery Contract!
This contract is managed by 0x018ab8e8c991
There are currently 12 people entered, competing to win 3.395 ether!
Want to try your luck?
Amount of ether to enter
1.0
Enter
Time to pick a winner?
Pick Winner
0xef188ba75737 has won!

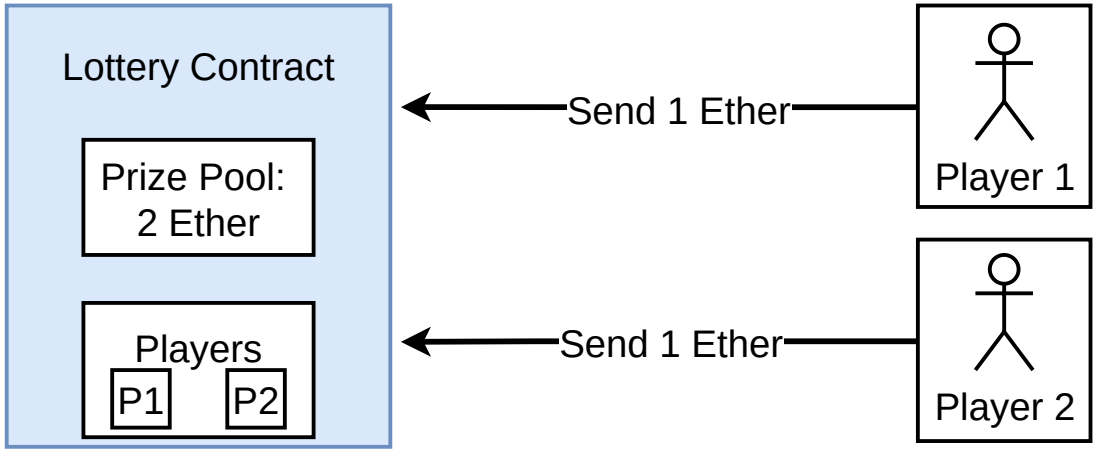

Page 1
https://www.draw.io
Retrieve information through calls to the contract
Lottery Contract!
This contract is managed by 0x018ab8e8c991
There are currently 12 people entered, competing to win 3.395 ether!
'Send' to the 'enter' function
Want to try your luck?
Amount of ether to enter
1.0
Enter
'Send' to the 'pickWinner' function
Time to pick a winner?
Pick Winner
Status text area
0xef188ba75737 has won!

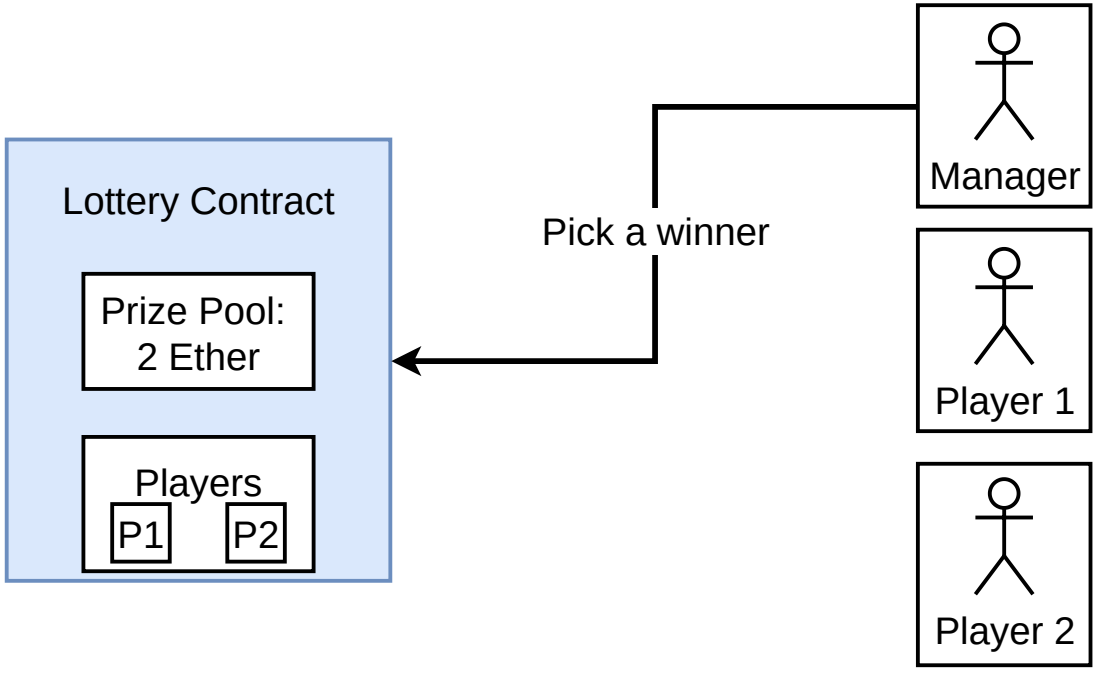

Page 1
https://www.draw.io
Rinkeby
We are assuming the user has Metamask installed!
Provider
web3 v0.20
Metamask
Provider
web3 v1.0
Our App

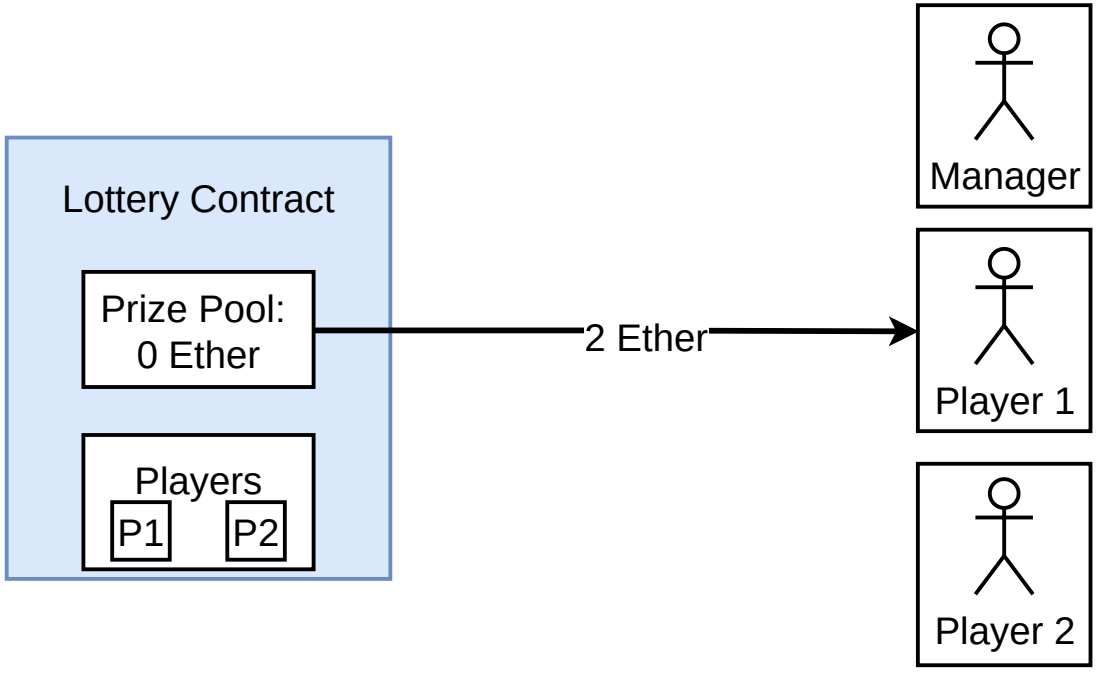

Page 1
https://www.draw.io
LOCAL COPY
web3
Contract ABI (interface)
Find the contract at address 0x1b8a92
Rinkeby
Deployed Lottery Instance


Time
Component renders
componentDidMount called
'Call' methods on contract
Set data on 'state'


Lottery Contract
Prize Pool:
0 Ether
Players
Player 1
Player 2


Lottery Contract
Prize Pool:
2 Ether
Players
P1
P2
Send 1 Ether
Send 1 Ether
Player 1
Player 2


Lottery Contract
Prize Pool:
2 Ether
Players
P1
P2
Pick a winner
Manager
Player 1
Player 2


Lottery Contract
Prize Pool:
0 Ether
Players
P1
P2
2 Ether
Manager
Player 1
Player 2

Lottery Contract

Variables

Name	Purpose
manager	Address of person who created the contract
players	Array of addresses of people who have entered

Functions

Name	Purpose
enter	Enters a player into the lottery
pickWinner	Randomly picks a winner and sends them the prize pool

Basic Types		
Name	Notes	Examples
string	Sequence of characters	"Hi there!" "Chocolate"
bool	Boolean value	true false
int	Integer, positive or negative. Has no decimal	0 -30000 59158
uint	'Unsigned' integer, positive number. Has no decimal	0 30000 999910
fixed/ufixed	'Fixed' point number. Number with a decimal after it	20.001 -42.4242 3.14
address	Has methods tied to it for sending money	0x18bae199c8dbae199c8d

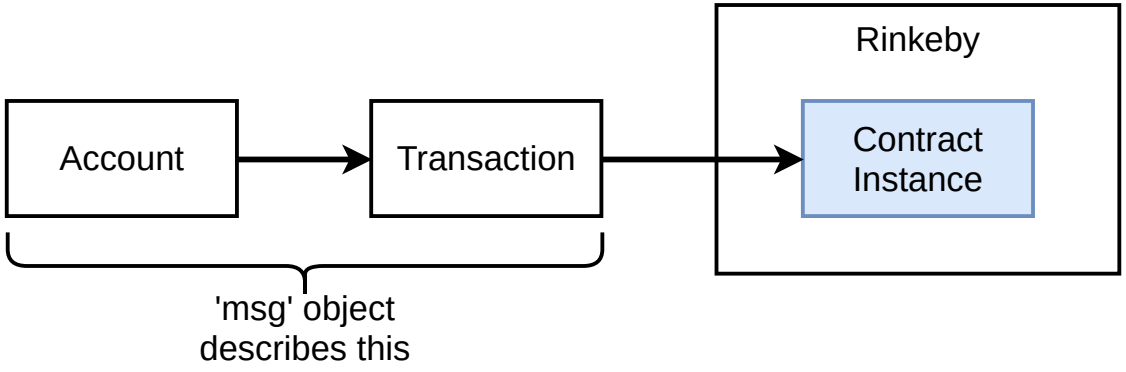
Integer Ranges		
Name	Lower Bound	Upper Bound
int8	-128	127
int16	-32,768	32,767
int32	-2,147,483,648	2,147,483,647
...	...	...
int256	Really, really negative	Really, really big

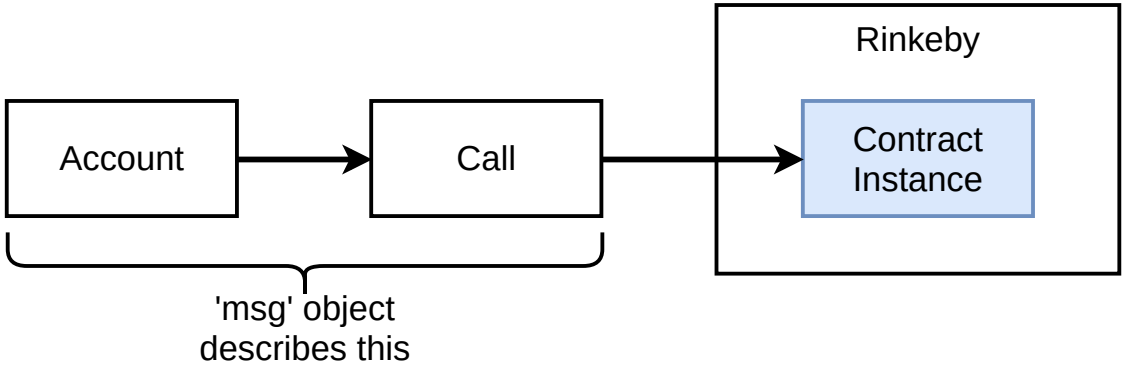
int == int256

Unsigned Integer Ranges		
Name	Lower Bound	Upper Bound
uint8	0	255
uint16	0	65,535
uint32	0	4,294,967,295
...	...	...
uint256	0	Really, really big

uint == uint256

The 'msg' Global Variable	
Property Name	Property Name
msg.data	'Data' field from the call or transaction that invoked the current function
msg.gas	Amount of gas the current function invocation has available
msg.sender	Address of the account that started the current function invocation
msg.value	Amount of ether (in wei) that was sent along with the function invocation


Rinkeby
Account
Transaction
Contract
Instance
'msg' object
describes this

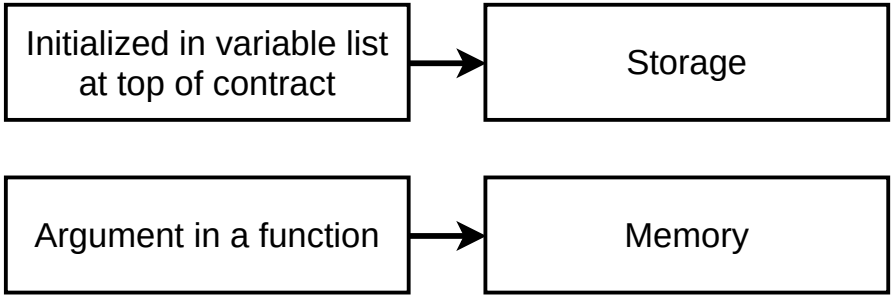

Rinkeby
Account
Call
Contract
Instance
'msg' object
describes this

Reference Types		
Name	Notes	Examples
fixed array	Array that contains a *single type* of element. Has an unchanging length	int[3] --> [1, 2, 3] bool[2] --> [true, false]
dynamic array	Array that contains a *single type* of element. Can change in size over time	int[] --> [1,2,3] bool[] --> [true, false]
mapping	Collection of key value pairs. Think of Javascript objects, Ruby hashes, or Python dictionary. All keys must be of the same type, and all values must be of the same type	mapping(string => string) mapping(int => bool)

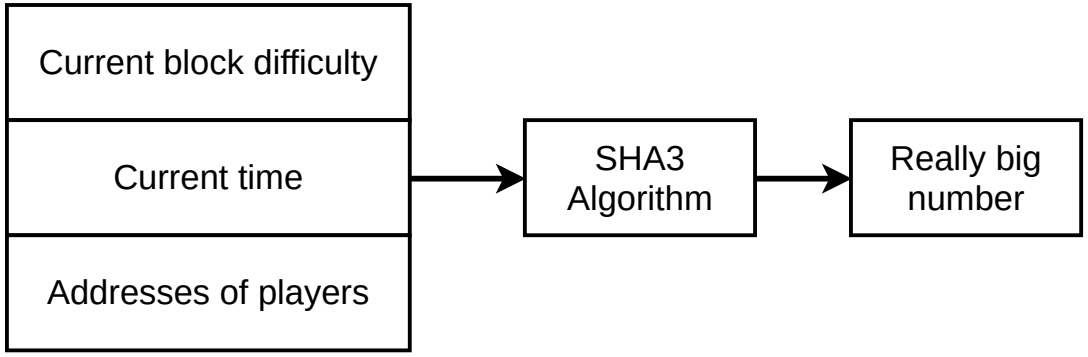
struct	Collection of key value pairs that can have different types.	struct Car {
		string make;
		string model;
		uint value;
		}

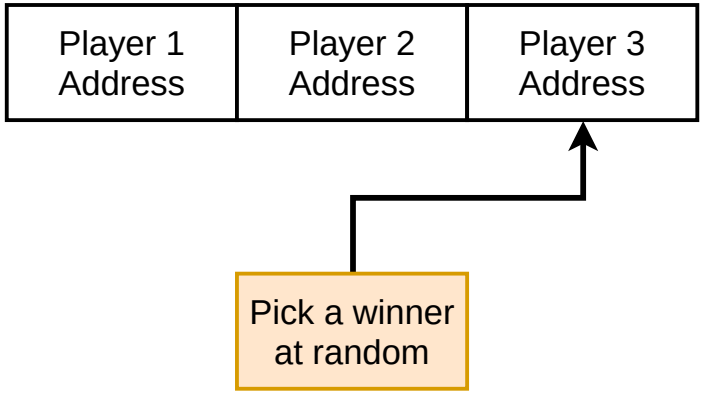
Where do Reference Types Exist?

Storage	Memory
The type is stored with the contract on the blockchain and can be accessed across multiple function calls	The type is created during a function invocation and then dumped forever


Initialized in variable list at top of contract
Storage
Argument in a function
Memory

Solidity World	ABI/JS/Web3 World
Nested Dynamic Arrays	Nested Dynamic Arrays *No Dice*


Current block difficulty
Current time
Addresses of players
SHA3 Algorithm
Really big number


Player 1 Address
Player 2 Address
Player 3 Address
Pick a winner at random

Lottery Contract

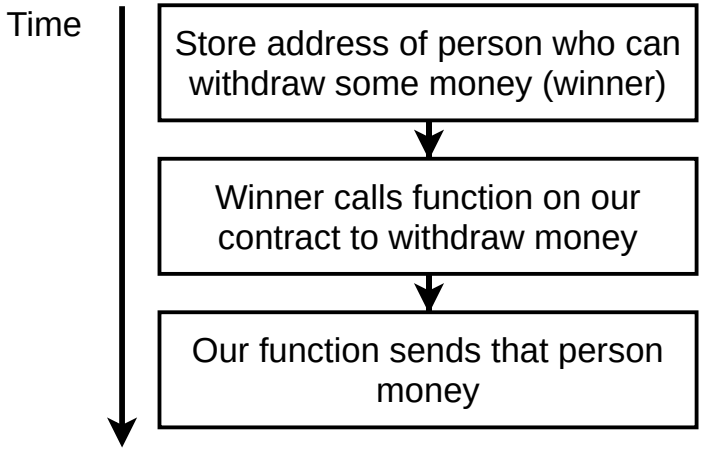
Variables

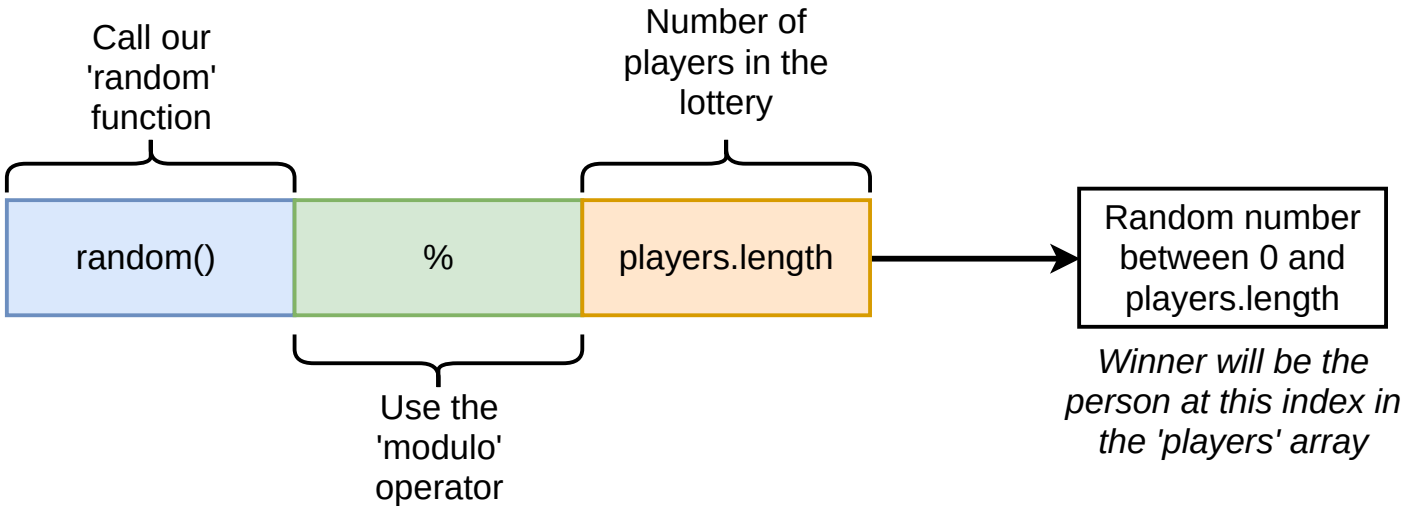
Name	Purpose
manager	Address of person who created the contract
players	Array of addresses of people who have entered
winner	Address of person who won the pick

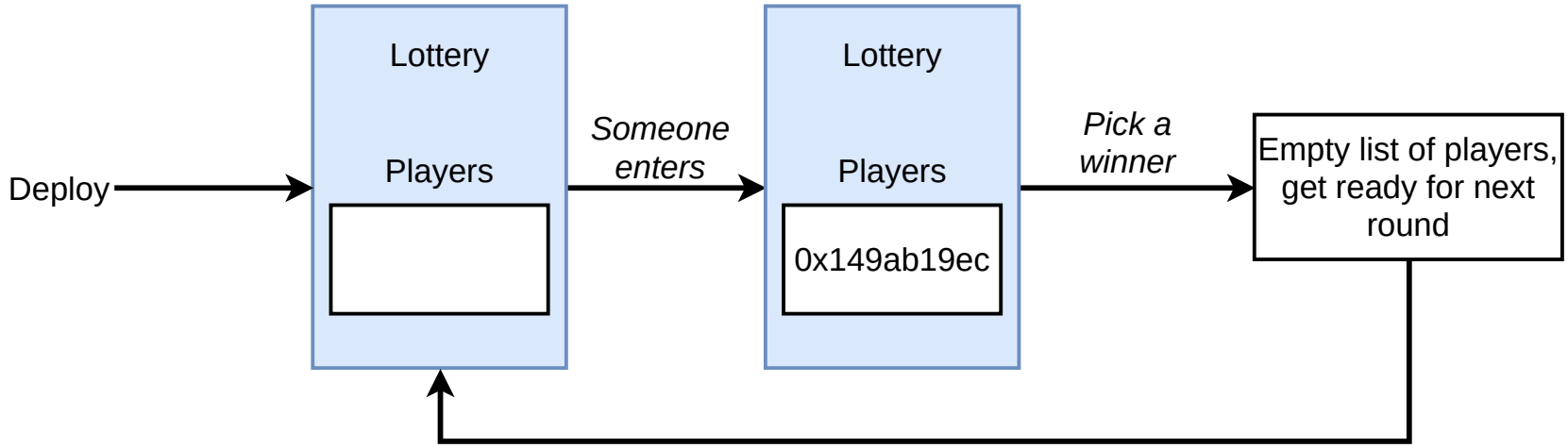
Functions

Name	Purpose
enter	Enters a player into the lottery
pickWinner	Randomly picks a winner

The 'Withdrawl' Pattern



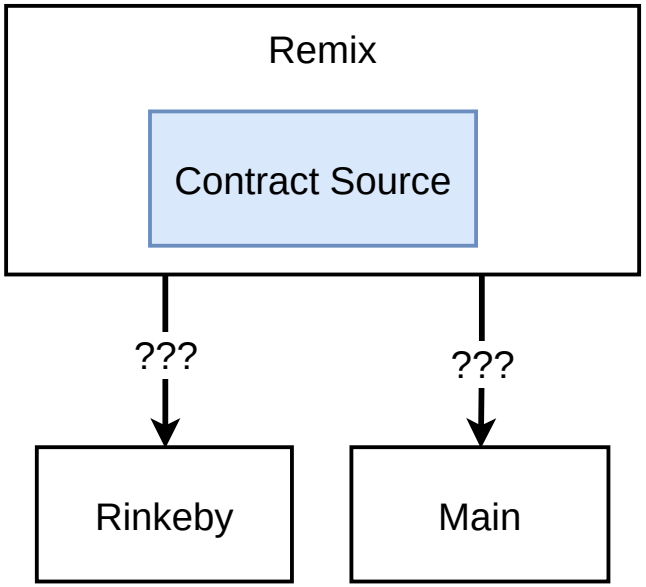

Call our 'random' function
Number of players in the lottery
random()
%
players.length
Random number between 0 and players.length
Use the 'modulo' operator
Winner will be the person at this index in the 'players' array

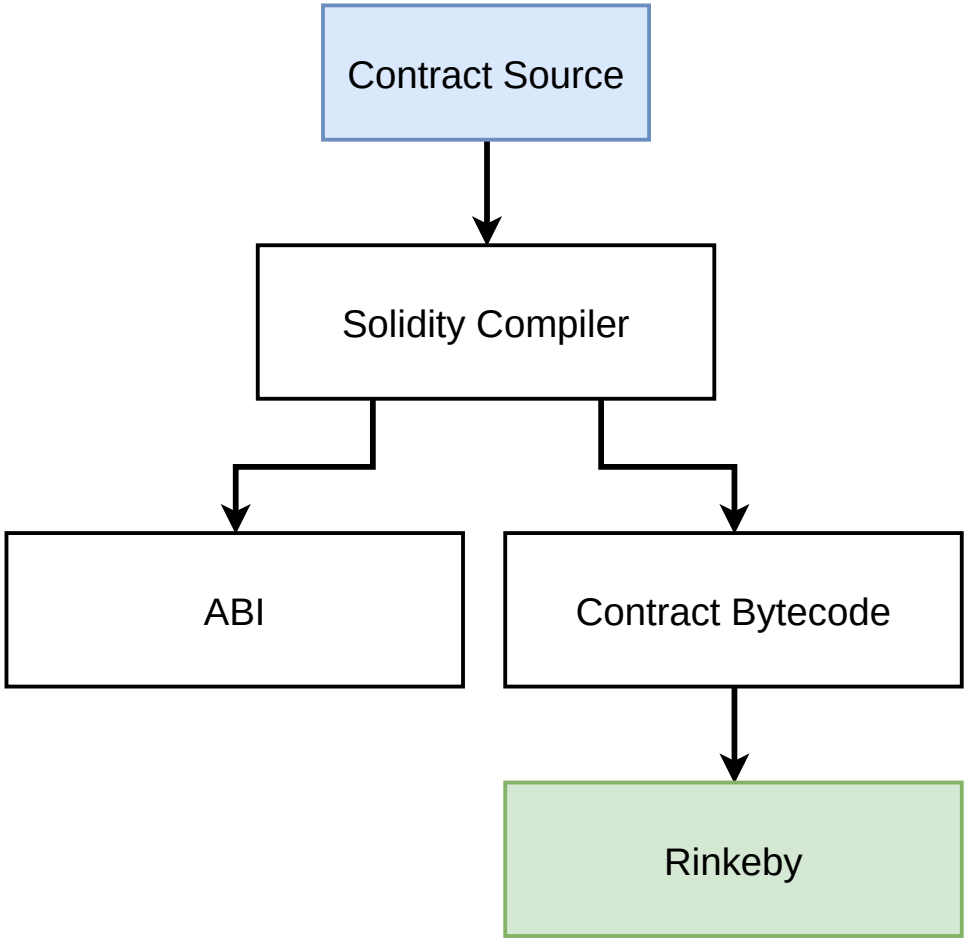

Deploy
Lottery
Players
Someone enters
Lottery
Players
0x149ab19ec
Pick a winner
Empty list of players, get ready for next round

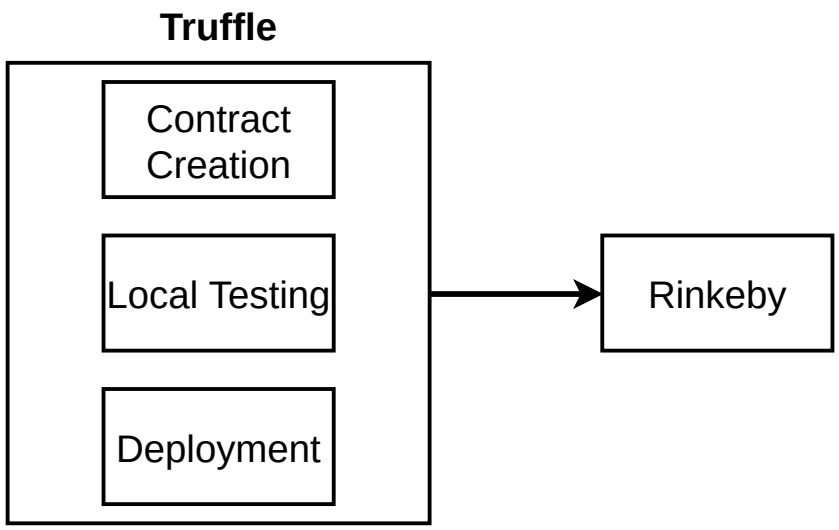
Our contract can only be used one time!
The 'pickWinner' function can be called by anyone
The 'pickWinner' function is not using the 'withdrawl' pattern, making it vulnerable to 're-entrancy' attacks

```
players[index].transfer(this.balance);
```

Could be vulnerable to a 're-entrancy' attack


Remix
Contract Source
???
???
Rinkeby
Main

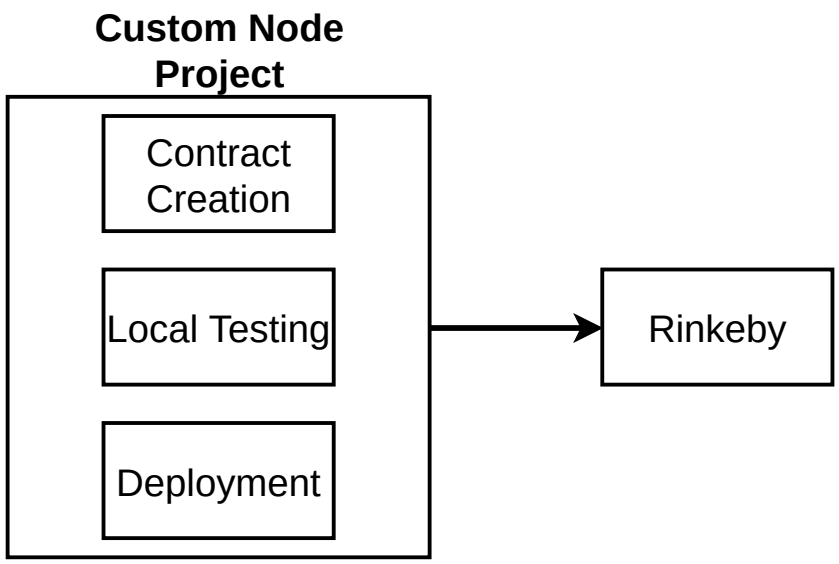

Contract Source
Solidity Compiler
ABI
Contract Bytecode
Rinkeby


Truffle
Contract Creation
Local Testing
Deployment
Rinkeby

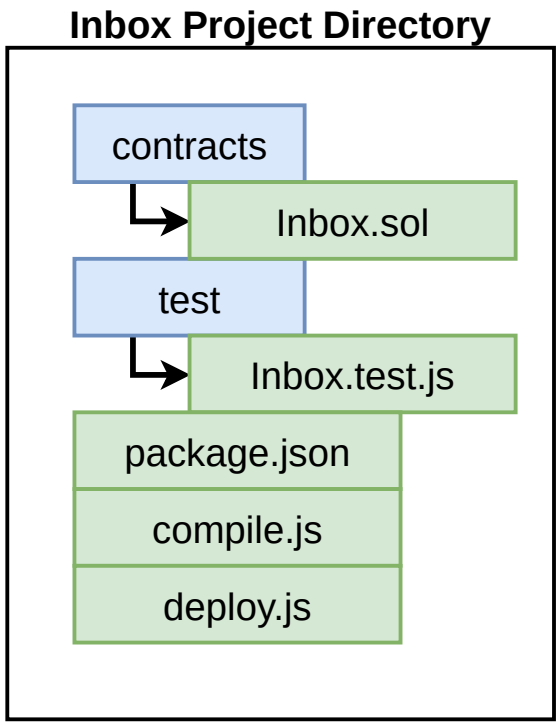
Truffle

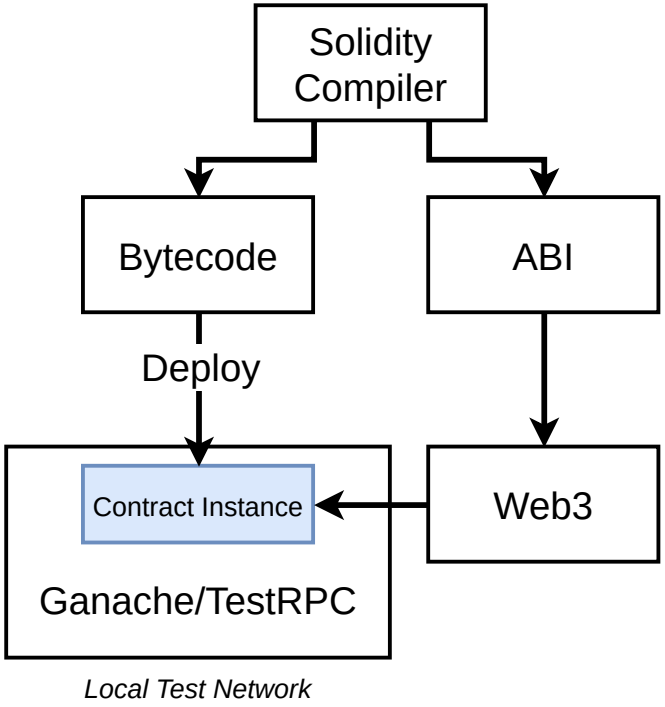
Undergoing rapid development
Some things don't work well
Some things don't work at all
Stuff breaks - patience is required.

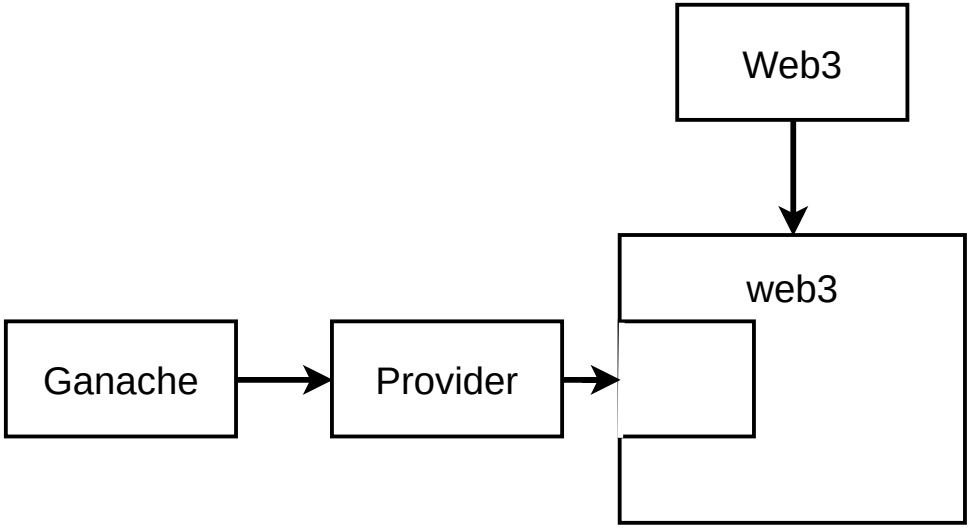
This is true of all current Ethereum tech


Custom Node Project
Contract Creation
Local Testing
Deployment
Rinkeby

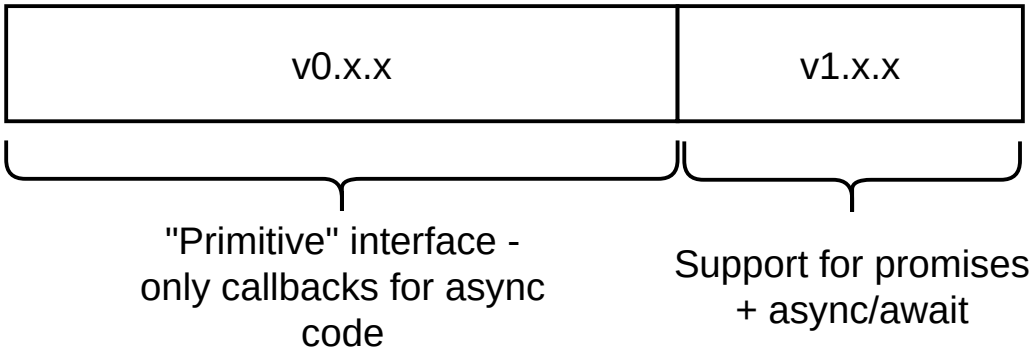
Boilerplate Design	
Issue	Solution
Need to be able to write Solidity code in a Javascript project	Set up the Solidity compiler to build our contracts
Need some way to rapidly test contracts without doing the manual testing we were doing with Remix	Set up a custom Mocha test runner that can somehow test Solidity code
Need some way to deploy our contract to public networks	Set up a deploy script to compile + deploy our contract


Inbox Project Directory
contracts
Inbox.sol
test
Inbox.test.js
package.json
compile.js
deploy.js

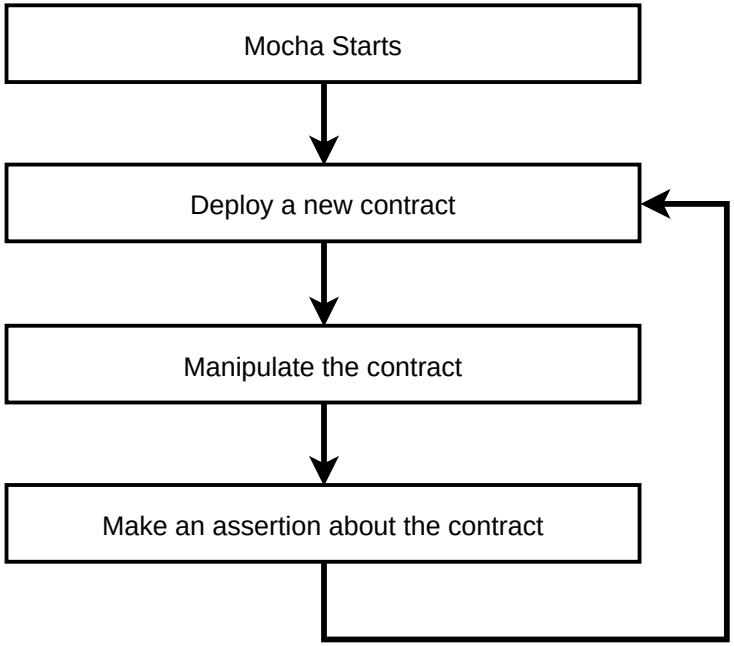

Solidity Compiler
Bytecode
ABI
Deploy
Contract Instance
Web3
Ganache/TestRPC
Local Test Network

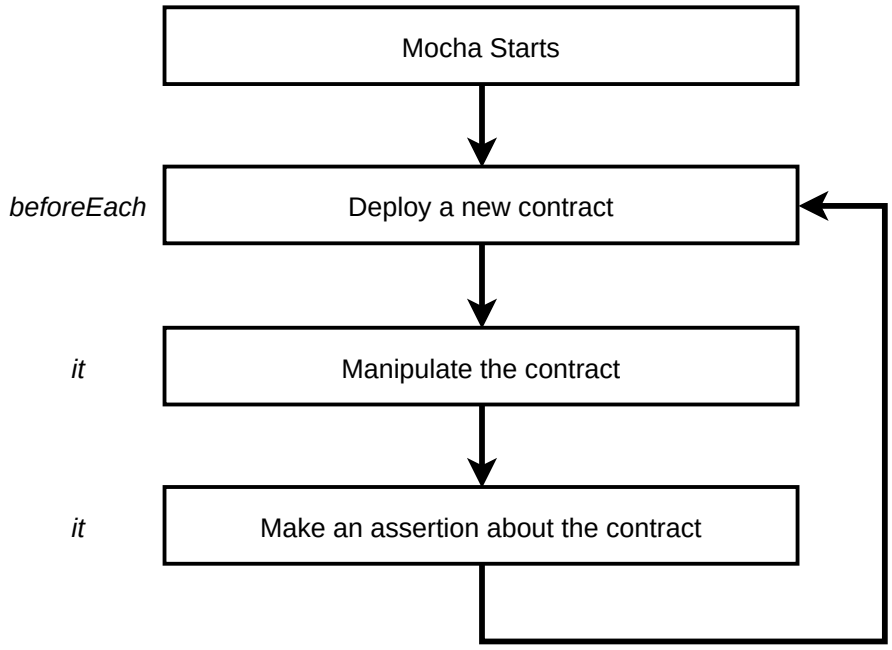

Web3
web3
Ganache
Provider

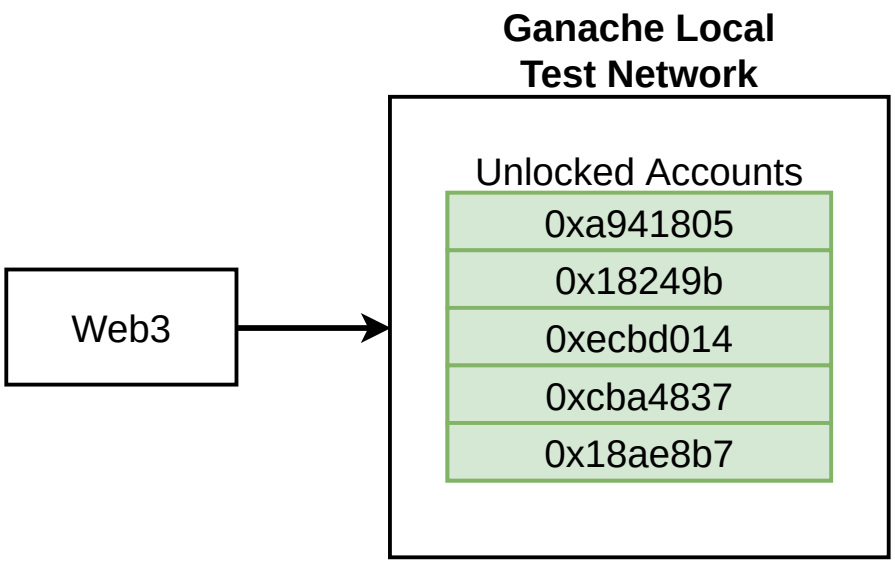
Web3 Versioning



Mocha Functions	
Function	Purpose
it	Run a test and make an assertion.
describe	Groups together 'it' functions.
beforeEach	Execute some general setup code.


Mocha Starts
Deploy a new contract
Manipulate the contract
Make an assertion about the contract


Mocha Starts
beforeEach
Deploy a new contract
it
Manipulate the contract
it
Make an assertion about the contract


Ganache Local
Test Network
Unlocked Accounts
0xa941805
0x18249b
0xecbd014
0xcba4837
0x18ae8b7
Web3

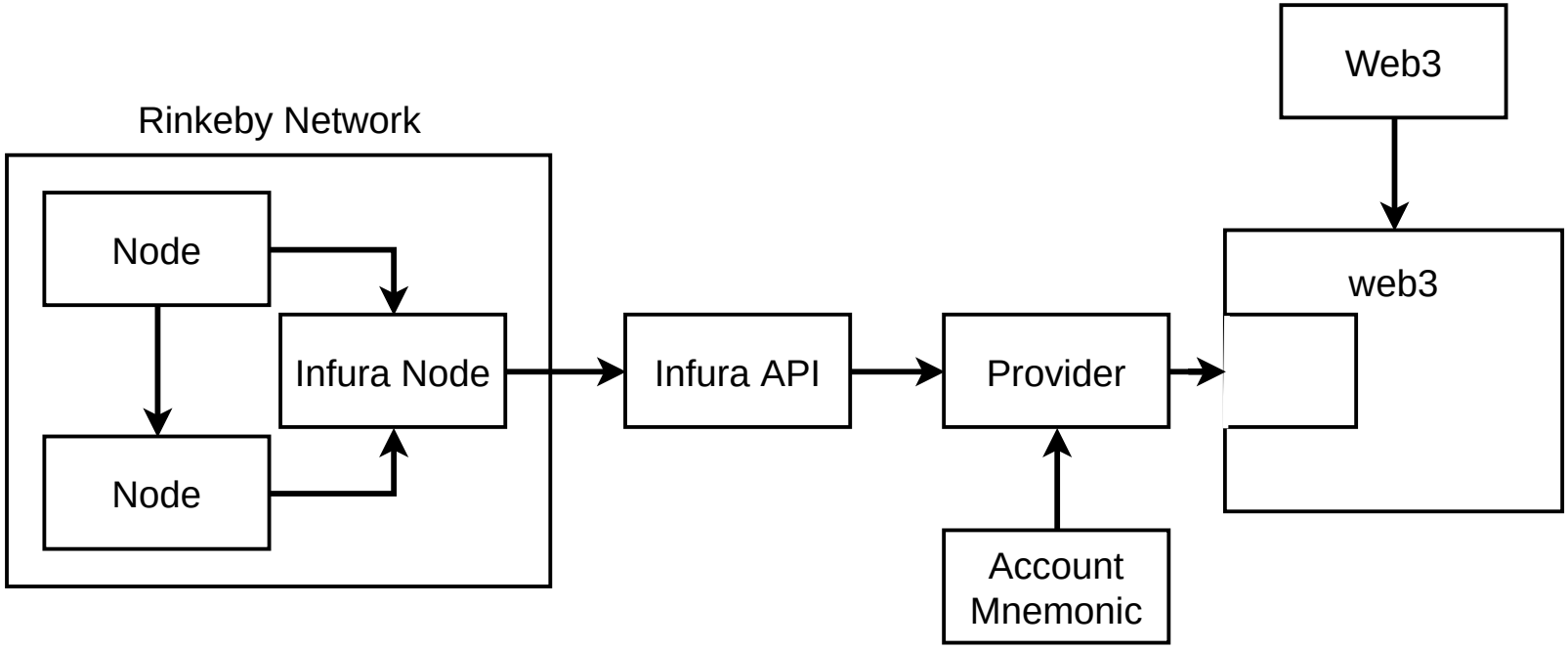
Teaches web3 about what methods an Inbox contract has

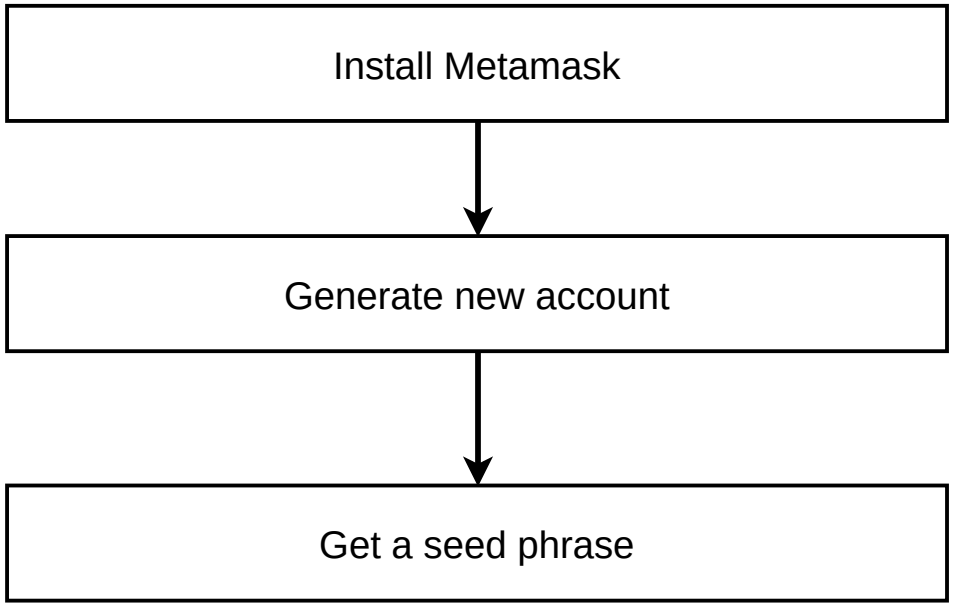
```
inbox = await new web3.eth.Contract(JSON.parse(interface))
  .deploy({ data: bytecode, arguments: ['Hi there!'] })
  .send({ from: accounts[0], gas: '1000000' });
```

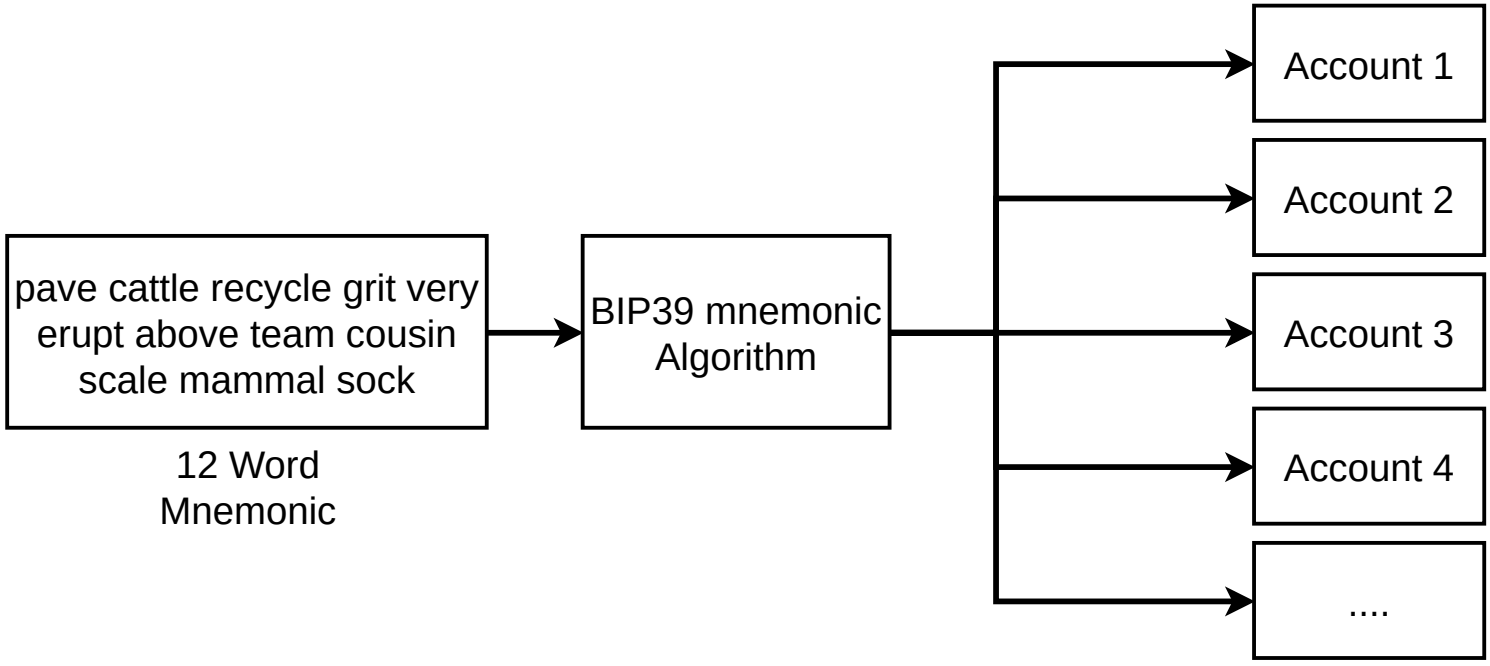
Tells web3 that we want to deploy a new copy of this contract

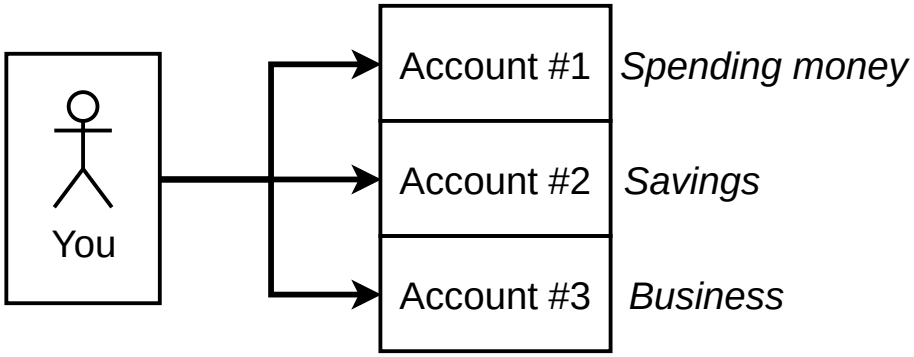
Instructs web3 to send out a transaction that creates this contract

Web3 With Contracts			
Goal	ABI	Bytecode	Address of deployed contract
Interact with deployed contract	✔	X	✔
Create a contract	✔	✔	X


Rinkeby Network
Node
Node
Infura Node
Infura API
Provider
Account
Mnemonic
Web3
web3


Install Metamask
Generate new account
Get a seed phrase


pave cattle recycle grit very erupt above team cousin scale mammal sock
12 Word Mnemonic
BIP39 mnemonic Algorithm
Account 1
Account 2
Account 3
Account 4
....


You
Account #1
Spending money
Account #2
Savings
Account #3
Business

Account #1

Address	0xcF01971DB0CAB2CBeE4A8C21BB7638aC1FA1c38c
Public	0xCA7442E17Dfe52542E50D6310b9aB64ca448DD13
Private	0xcF01971DB0CAB2CBeE4A8C21BB7638aC1FA1c38c

Account #2

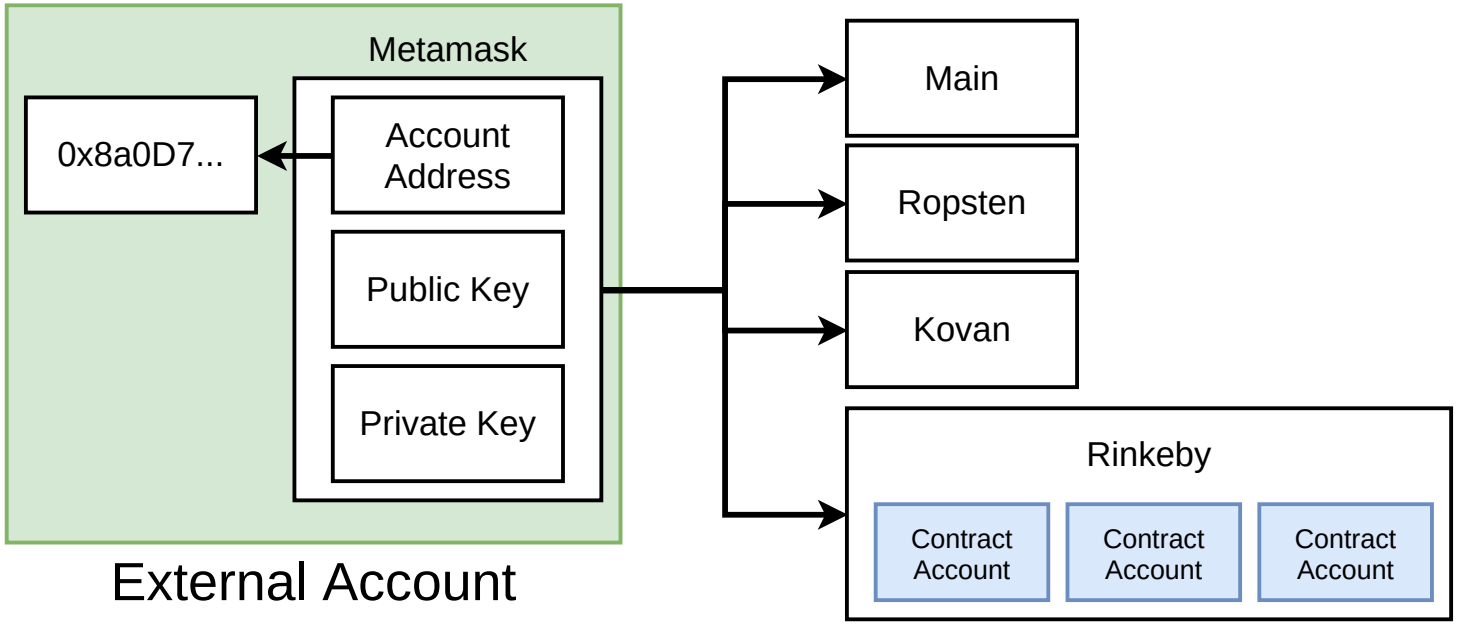
Address	0xf3Dd04C449669a89a7cF492c6fA8EF9aF388Ebd8
Public	0xd2A1faC633568cde440789D061Ef08D4f8a7054a
Private	0xf3Dd04C449669a89a7cF492c6fA8EF9aF388Ebd8

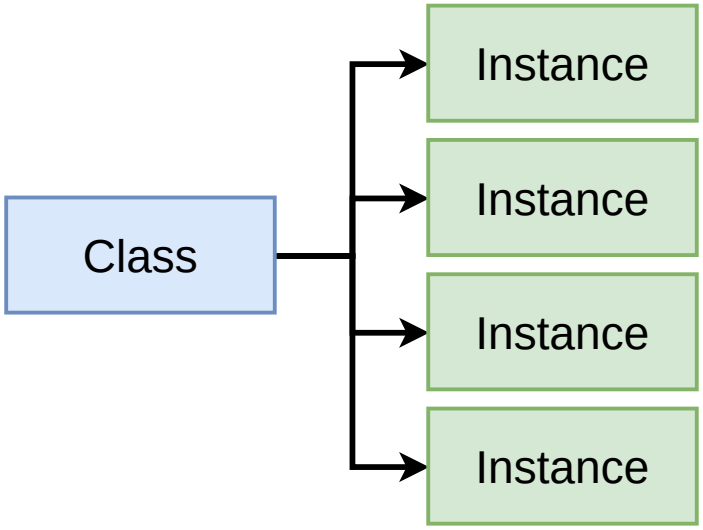
Account #3

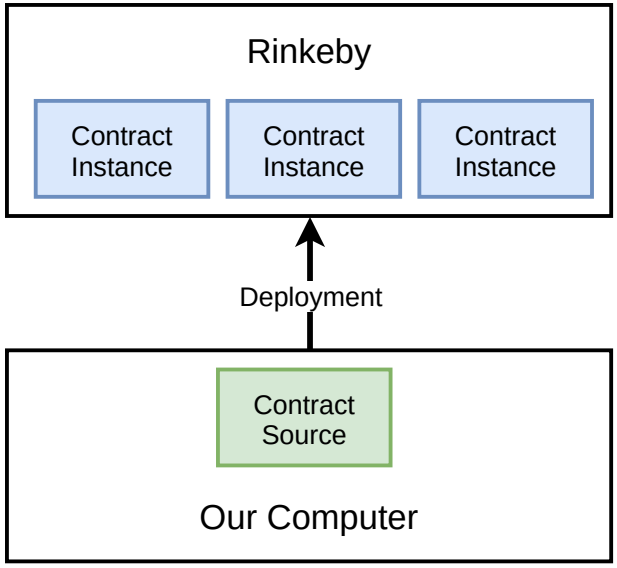
Address	0xcF01971DB0CAB2CBeE4A8C21BB7638aC1FA1c38c
Public	0xC1b6c46e1fc19B8b990AdF4601db9B0842e22C9F
Private	0xcF01971DB0CAB2CBeE4A8C21BB7638aC1FA1c38c

infura.io

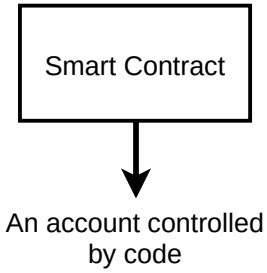
Infura API signup

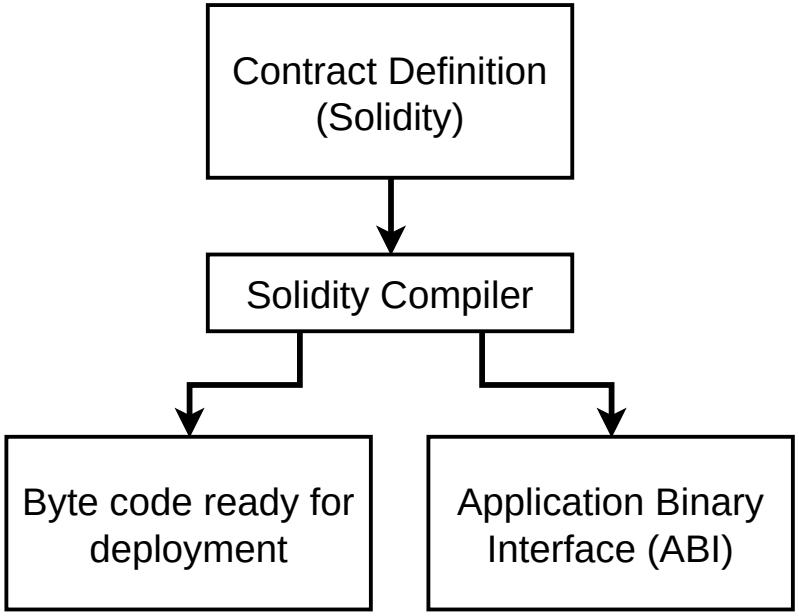

Metamask
0x8a0D7...
Account Address
Public Key
Private Key
External Account
Main
Ropsten
Kovan
Rinkeby
Contract Account
Contract Account
Contract Account

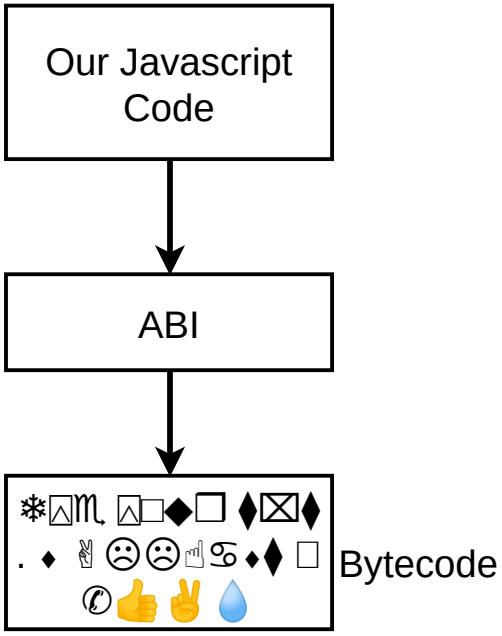

Class
Instance
Instance
Instance
Instance


Rinkeby
Contract Instance
Contract Instance
Contract Instance
Deployment
Contract Source
Our Computer

Contract Account	
Field	Description
balance	Amount of ether this account owns
storage	Data storage for this contract
code	Raw machine code for this contract


Smart Contract
An account controlled by code


Contract Definition (Solidity)
Solidity Compiler
Byte code ready for deployment
Application Binary Interface (ABI)


Our Javascript Code
ABI
Bytecode

Solidity Programming Language

Written in .sol files

Strongly typed

Similar to Javascript

Has several huge, gigantic 'gotchas'

Specifies the version of Solidity that our code is written with

Defines a new contract (remember classes!) that will have some number of methods and variables

Declares all of the instance variables (and their types) that will exist in this contract

Defines different functions that will be members of this contract

ContractDefinition **Inbox** 0

```
pragma solidity ^0.4.17;

contract Inbox {
    string public message;

    function Inbox(string initialMessage) public {
        message = initialMessage;
    }

    function setMessage(string newMessage) public {
        message = newMessage;
    }

    function getMessage() public view returns (string) {
        return message;
    }
}
```

Function name

Function type

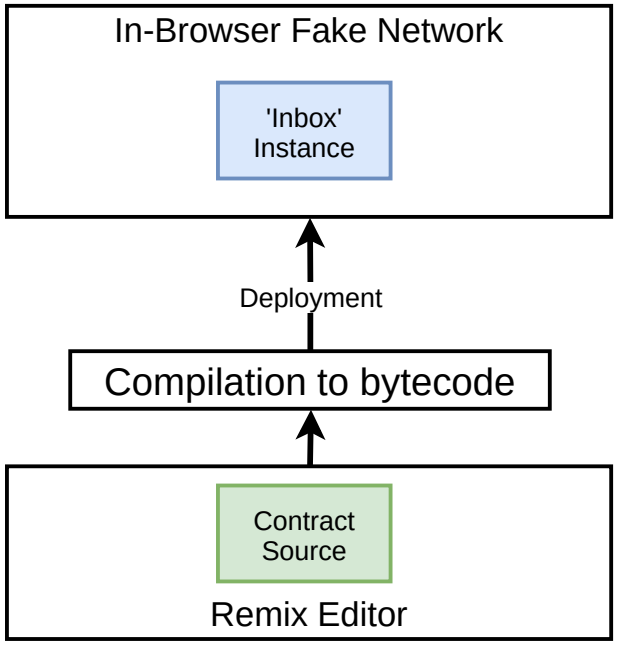
Return types

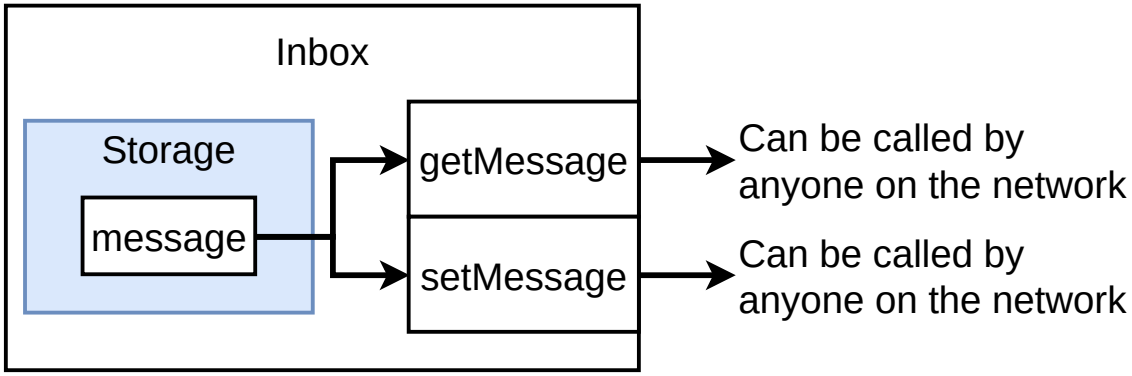
```
function getMessage() public view returns (string) {
    return message;
}
```

Can only use one per function

They mean the same thing

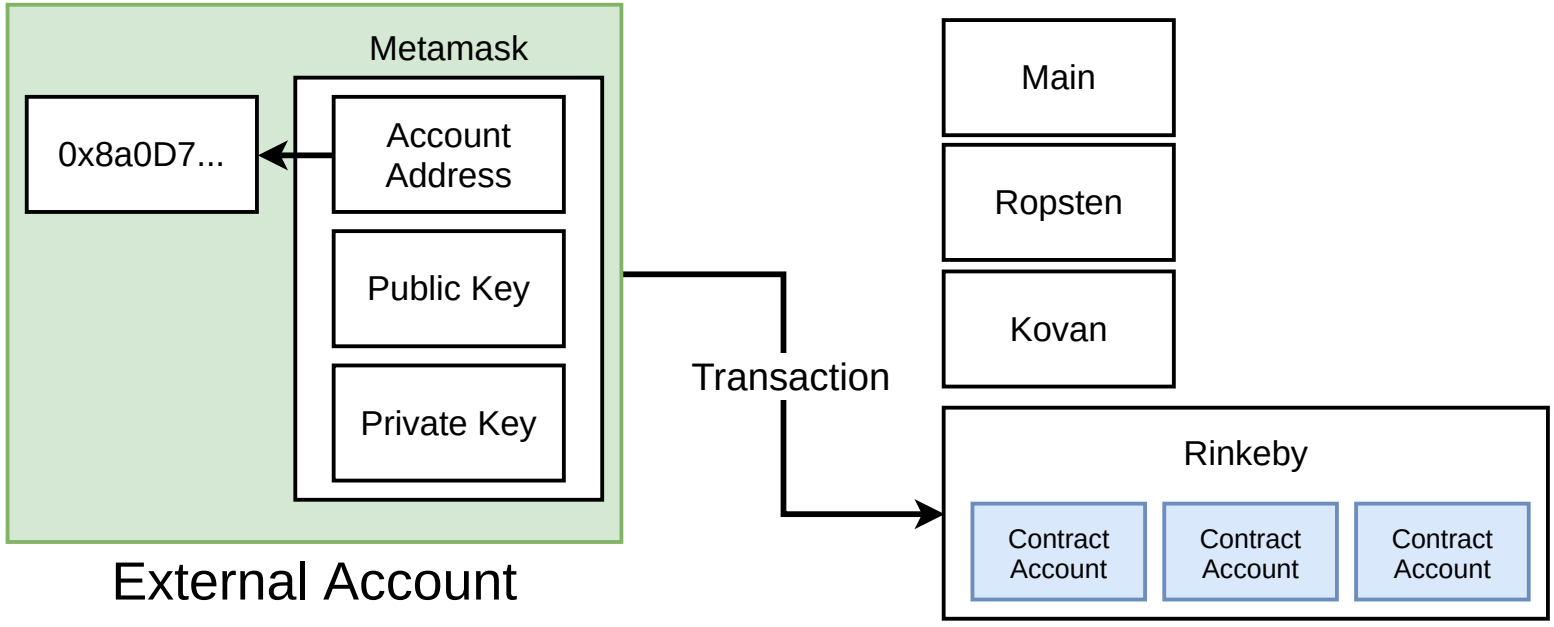
Common Function Types	
public	Anyone can call this function
private	Only this contract can call this function.
view	This function returns data and does *not* modify the contract's data
constant	This function returns data and does *not* modify the contract's data
pure	Function will not modify or even *read* the contract's data
payable	When someone call this function they might send ether along


In-Browser Fake Network
'Inbox' Instance
Deployment
Compilation to bytecode
Contract Source
Remix Editor


Inbox
Storage
message
getMessage
setMessage
Can be called by anyone on the network
Can be called by anyone on the network

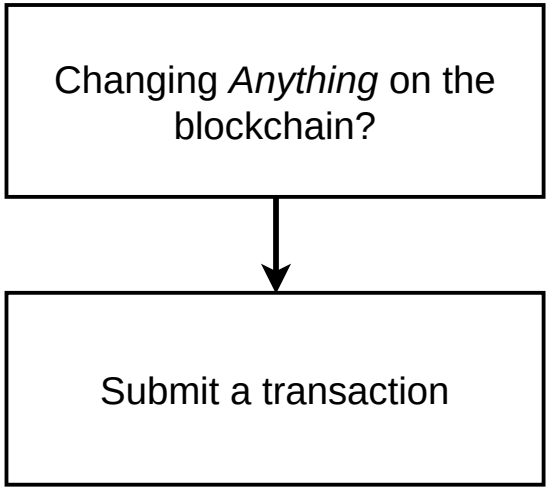
External to External Account Transaction

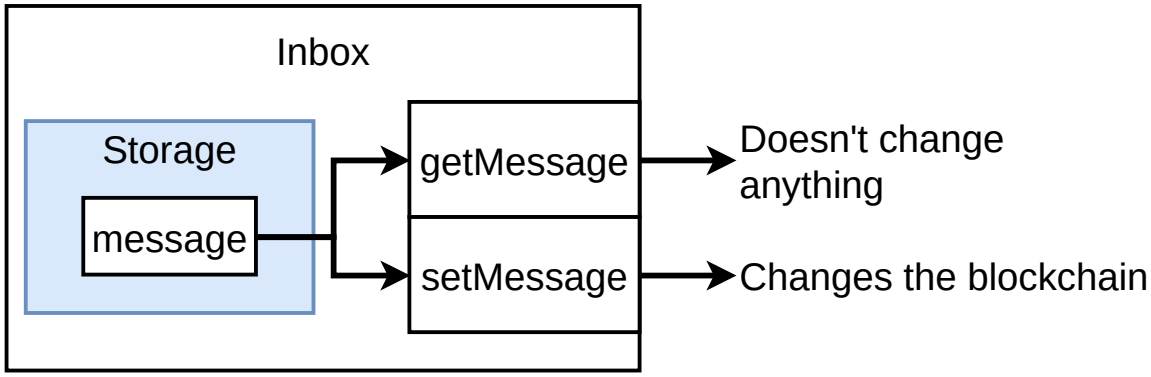
nonce	How many times the sender has sent a transaction
to	Address of account this money is going to
value	Amount of 'Wei' to send to the target address
gasPrice	Amount of Wei the sender is willing to pay per unit gas to get this transaction processed
startGas/gasLimit	Units of gas that this transaction can consume
v	Cryptographic pieces of data that can be used to generate the senders account address. Generated from the *sender's* private key.
r	
s	


Metamask
0x8a0D7...
Account Address
Public Key
Private Key
External Account
Transaction
Main
Ropsten
Kovan
Rinkeby
Contract Account
Contract Account
Contract Account

External Account to Create Contract Transaction

nonce	How many times the sender has sent a transaction
to	-
data	Compiled bytecode of the contract
value	Amount of 'Wei' to send to the target address
gasPrice	Amount of Wei the sender is willing to pay per unit gas to get this transaction processed
startGas/gasLimit	Units of gas that this transaction can consume
v	Cryptographic pieces of data that can be used to generate the senders account address. Generated from the *sender's* private key.
r	
s	


Changing *Anything* on the blockchain?
Submit a transaction


Inbox
Storage
message
getMessage
setMessage
Doesn't change anything
Changes the blockchain

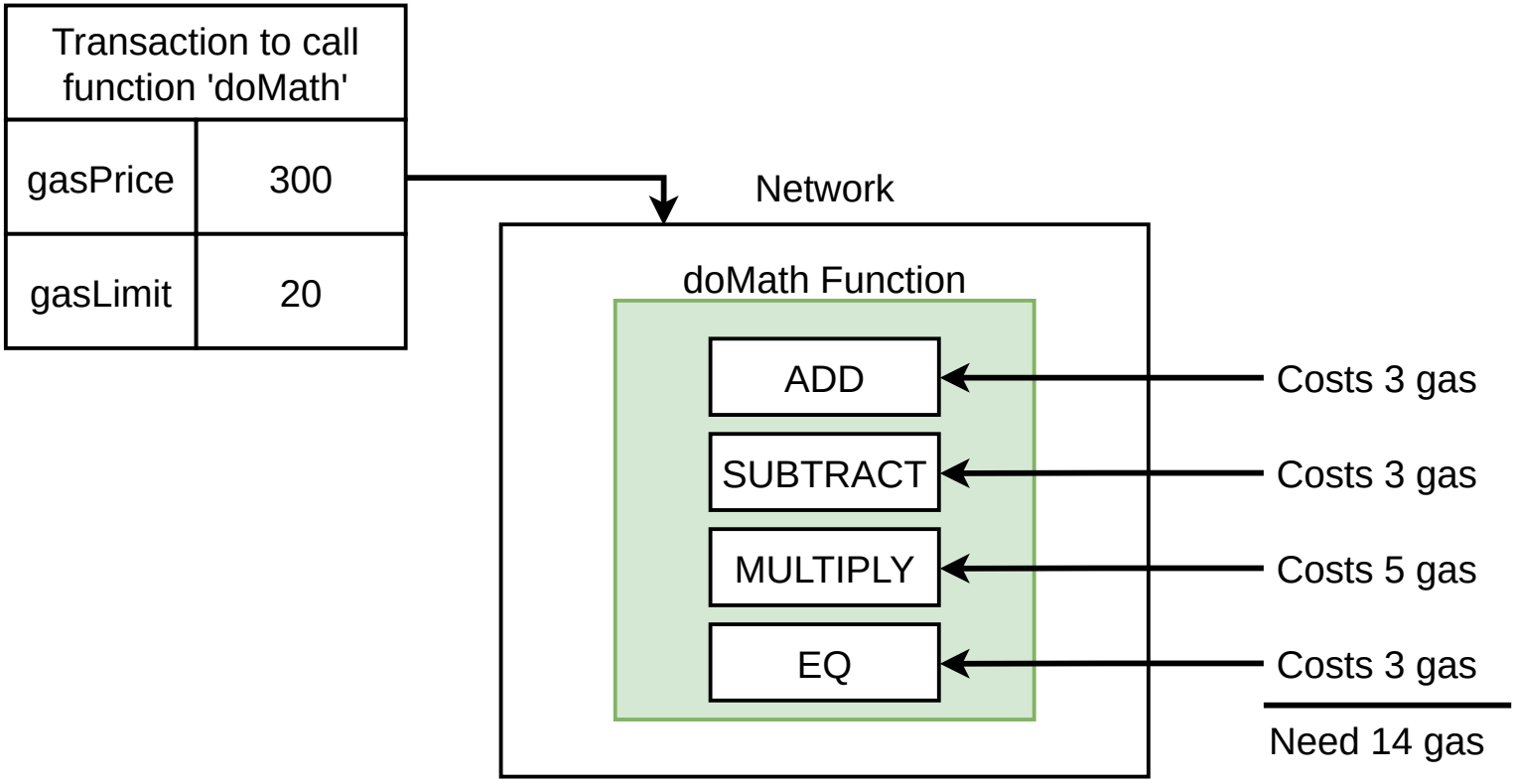
Running Contract Functions	
'Calling' a Function	Sending a Transaction to a Function
Cannot modify the contract's data	Can modify a contract's data
Can return data	Takes time to execute!
Runs instantly	Returns the transaction hash
Free to do!	Costs money!

1 Dollar == 100 Cents

== 

1 Ether	==	1,000,000,000,000,000,000 Wei

gasPrice	Amount of Wei the sender is willing to pay per unit gas to get this transaction processed
startGas/gasLimit	Units of gas that this transaction can consume


Transaction to call function 'doMath'
gasPrice
300
gasLimit
20
Network
doMath Function
ADD
SUBTRACT
MULTIPLY
EQ
Costs 3 gas
Costs 3 gas
Costs 5 gas
Costs 3 gas
Need 14 gas

gasPrice	300

Used 14 gas

$$\text{Total cost} = 300\ \frac{\text{wei}}{\text{gas}} \times 14\ \text{gas} = 4{,}200\ \text{wei}$$