Blockchain Application Development Workshop

By Edson Alcala

About me

- Bachelor’s Degree in Electronics Engineering
- MSc Design Informatics
- 5 years of experience as “Full stack” developer
- Currently working as Research Associate in Blockchain + Location

Intro

- The workshop will run from 10am to 5pm
- We will have lunch break at 1pm
- Feel free to interrupt for questions at any time (specially if you don't understand me)
- I will say "it depends" so much
- Github repository (slides, code and resources)
- Slack channel https://goo.gl/Zoxu5L

Our Goal

- Develop distributed applications from idea to release

Why is important?

- Materialise our ideas
- Create MVPs
- Win a hackathon
- Land a job
- Earn money

$12 million USD in sales
in its first month after launch

Check more DApps in https://dappradar.com/

What to expect?

- Practical workshop
- Demos
- Recommendations
- Extra material (articles, books, notes, videos)

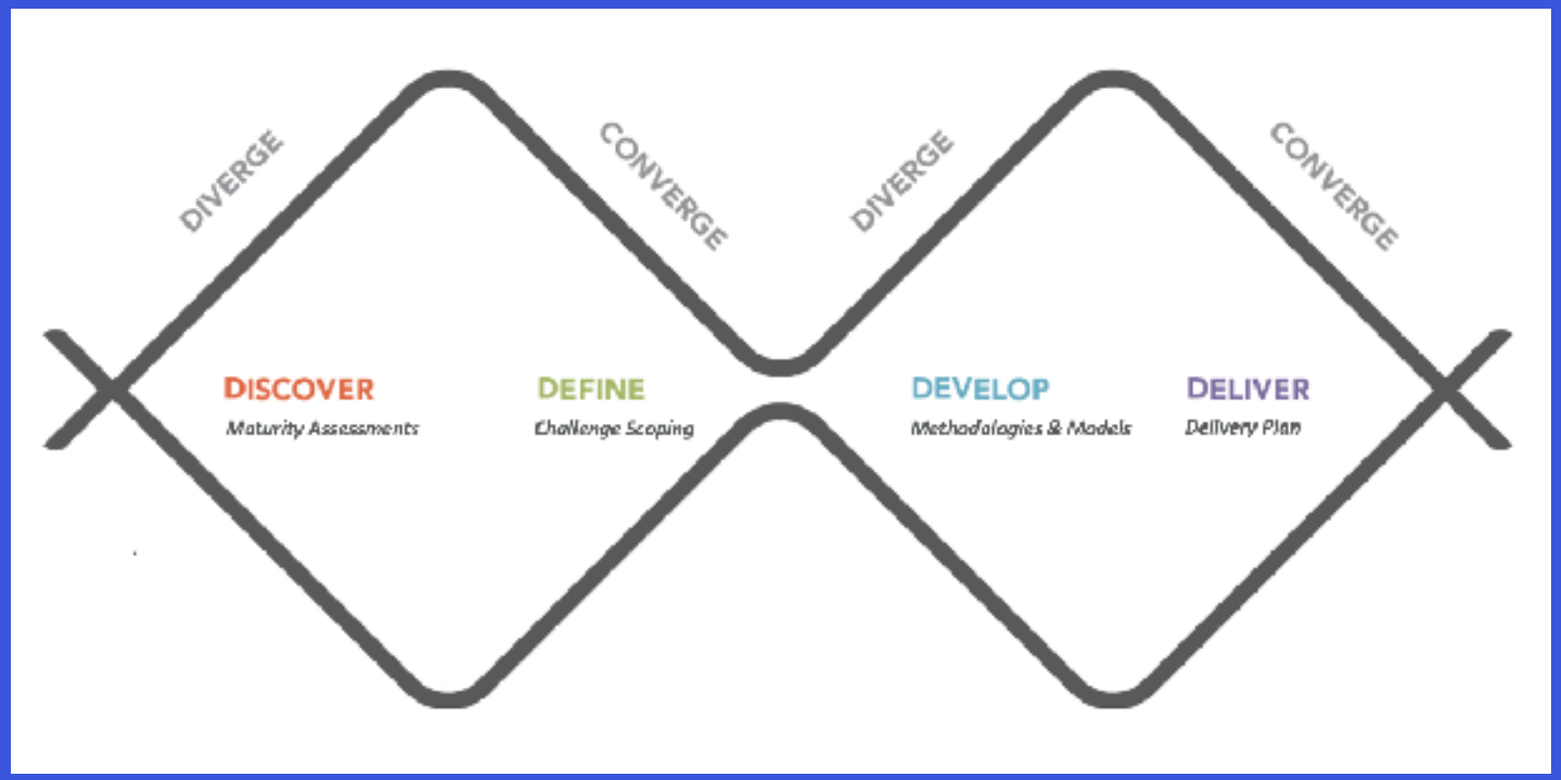
What to expect?

- Practical workshop
- Demos
- Recommendations
- Extra material (articles, books, notes, videos)

We will cover

- Design
- Front end
- Backend
- DevOps
- Smart contracts

The design process



https://www.designcouncil.org.uk/news-opinion/design-process-what-double-diamond

The design process

Basic Concepts

- What is a DAO
- A DAO in the Blockchain
- Introduction to self sovereign identity
- uPort demo

DAO

- Organisation whose decisions are made electronically through the vote of its members.
- Rules are explicitly written in code that define which actions an organisation will take.

Reference

DAO

Token holders

Holders of tokens (e.g. obtained via an ICO) can influence the actions of the DAO, draw profits from it, etc.

Coders

The authors of the DAO's code know its construction best but need not be involved in implementing the DAO.

Regulators

Public institutions, particularly in regulated industries, as well as tax authorities, may take an interest in the DAO's activity.

```
event Voted(uint ind
proposalID, bool
position, address
indexed voter); event
ProposalTallied(uint
indexed proposalID,
result, uint quorum);
event NewCurator(add
indexed _newCurator);
```

DAO

Contractors

Contractors are people cooperating with the DAO, e.g. providing certain services.

Oracles

Oracles provide various types of data and information from the outside world (beyond the blockchain).

Others

The platform (e.g. Ethereum) and its creators; curators (people who perform certain tasks for the DAO).

Examples: MakerDao

MKR is a utility token, governance token and recapitalisation resource of the Maker system.

“As a governance token, MKR is used by MKR holders to vote for the risk management and business logic of the Maker system”

“As a utility token, MKR is required for paying the fees in the Maker system”

https://makerdao.com/
https://medium.com/makerdao/what-is-mkr-e6915d5ca1b3

A DAO in the Blockchain

```
contract Congress is owned, tokenRecipient {
    // Contract Variables and events
    uint public minimumQuorum;
    uint public debatingPeriodInMinutes;
    int public majorityMargin;
    Proposal[] public proposals;
    uint public numProposals;
    mapping (address => uint) public memberId;
    Member[] public members;

    event ProposalAdded(uint proposalID, address recipient, uint amount, string description);
    event Voted(uint proposalID, bool position, address voter, string justification);
    event ProposalTallied(uint proposalID, int result, uint quorum, bool active);
    event MembershipChanged(address member, bool isMember);
    event ChangeOfRules(uint newMinimumQuorum, uint newDebatingPeriodInMinutes, int newMajorityMargin);

    struct Proposal {
        address recipient;
        uint amount;
        string description;
        uint votingDeadline;
        bool executed;
        bool proposalPassed;
        uint numberOfVotes;
        int currentResult;
        bytes32 proposalHash;
        Vote[] votes;
        mapping (address => bool) voted;
```

A DAO in the Blockchain

```
/**
 * Add member
 *
 * Make `targetMember` a member named `memberName`
 *
 * @param targetMember ethereum address to be added
 * @param memberName public name for that member
 */
function addMember(address targetMember, string memberName) onlyOwner public {
    uint id = memberId[targetMember];
    if (id == 0) {
        memberId[targetMember] = members.length;
        id = members.length++;
    }

    members[id] = Member({member: targetMember, memberSince: now, name: memberName});
    MembershipChanged(targetMember, true);
}
```

A DAO in the Blockchain

```
/**
 * Add Proposal
 *
 * Propose to send `weiAmount / 1e18` ether to `beneficiary` for `jobDescription`.
 *
 * @param beneficiary who to send the ether to
 * @param weiAmount amount of ether to send, in wei
 * @param jobDescription Description of job
 * @param transactionBytecode bytecode of transaction
 */
function newProposal(
    address beneficiary,
    uint weiAmount,
    string jobDescription,
    bytes transactionBytecode
)
    onlyMembers public
    returns (uint proposalID)
{
    proposalID = proposals.length++;
    Proposal storage p = proposals[proposalID];
    p.recipient = beneficiary;
    p.amount = weiAmount;
    p.description = jobDescription;
    p.proposalHash = keccak256(beneficiary, weiAmount, transactionBytecode);
    p.votingDeadline = now + debatingPeriodInMinutes * 1 minutes;
    p.executed = false;
    p.proposalPassed = false;
    p.numberOfVotes = 0;
```

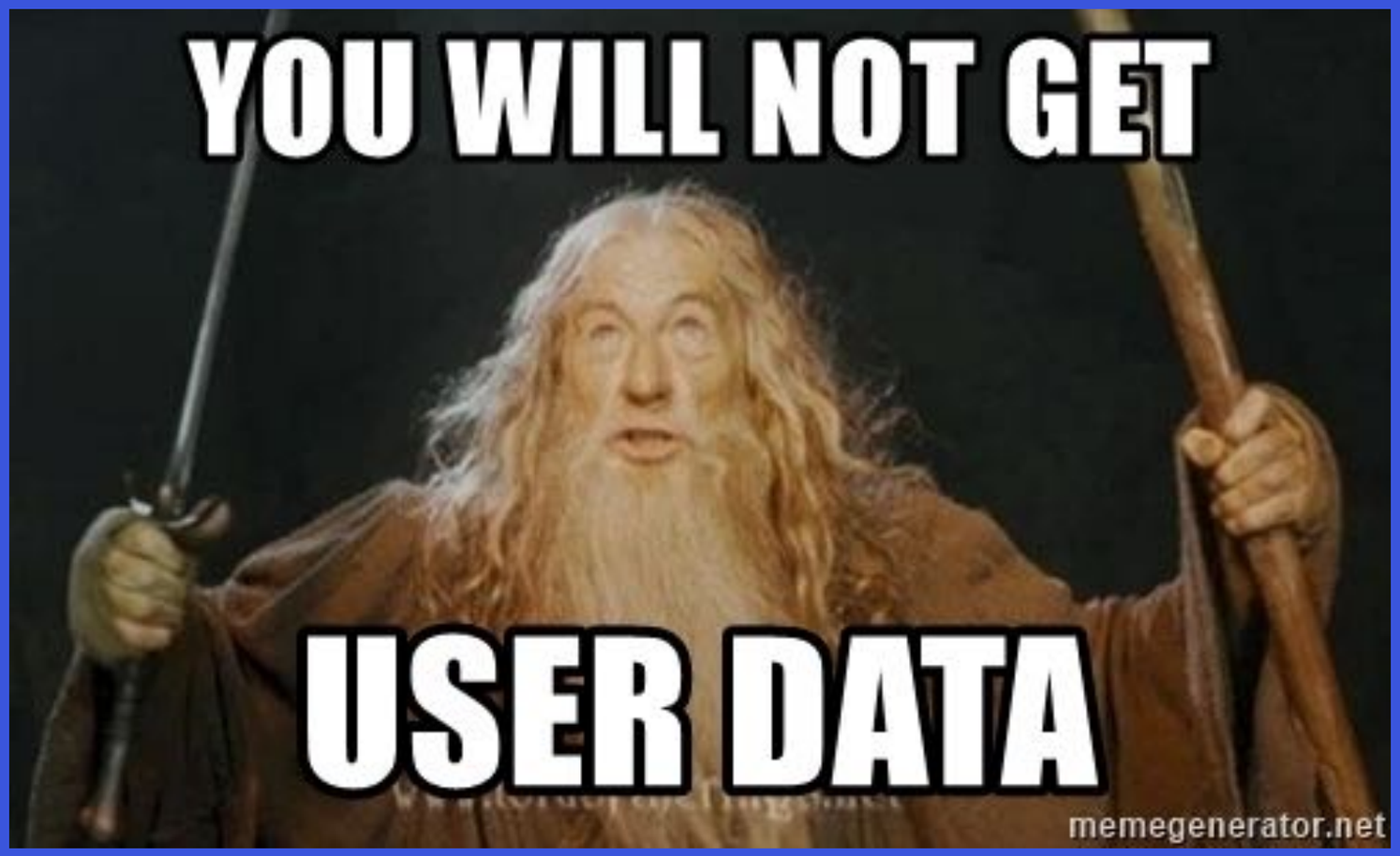
Self sovereign identity

- Users own their own data
- Users have control over their personal data and how, when, and to whom that personal data is revealed

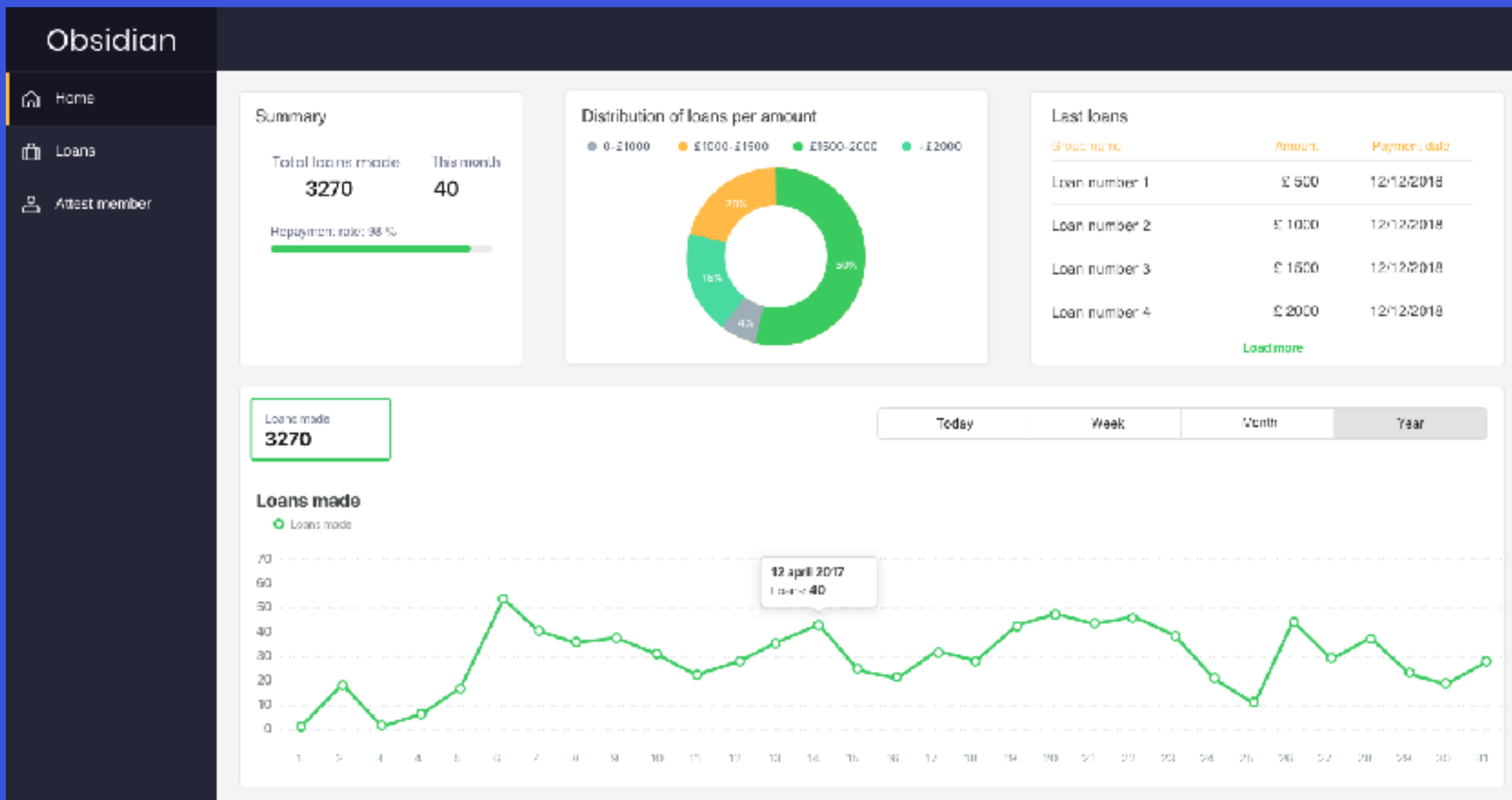
Self sovereign identity

- How could businesses work without people’s information?

Do your need user’s data?



Sample


Obsidian
Home
Loans
Attest member
Summary
Total loans made
3270
This month
40
Distribution of loans per amount
Last loans
Loan number 1
£ 500
12/12/2018
Loan number 2
Loan number 3
Loan number 4
£ 2000
Load more
Loans made
3270
Today
Week
Year
Loans made
12 april 2017

Self sovereign identity solutions

Sovrin

Veres one

uPort

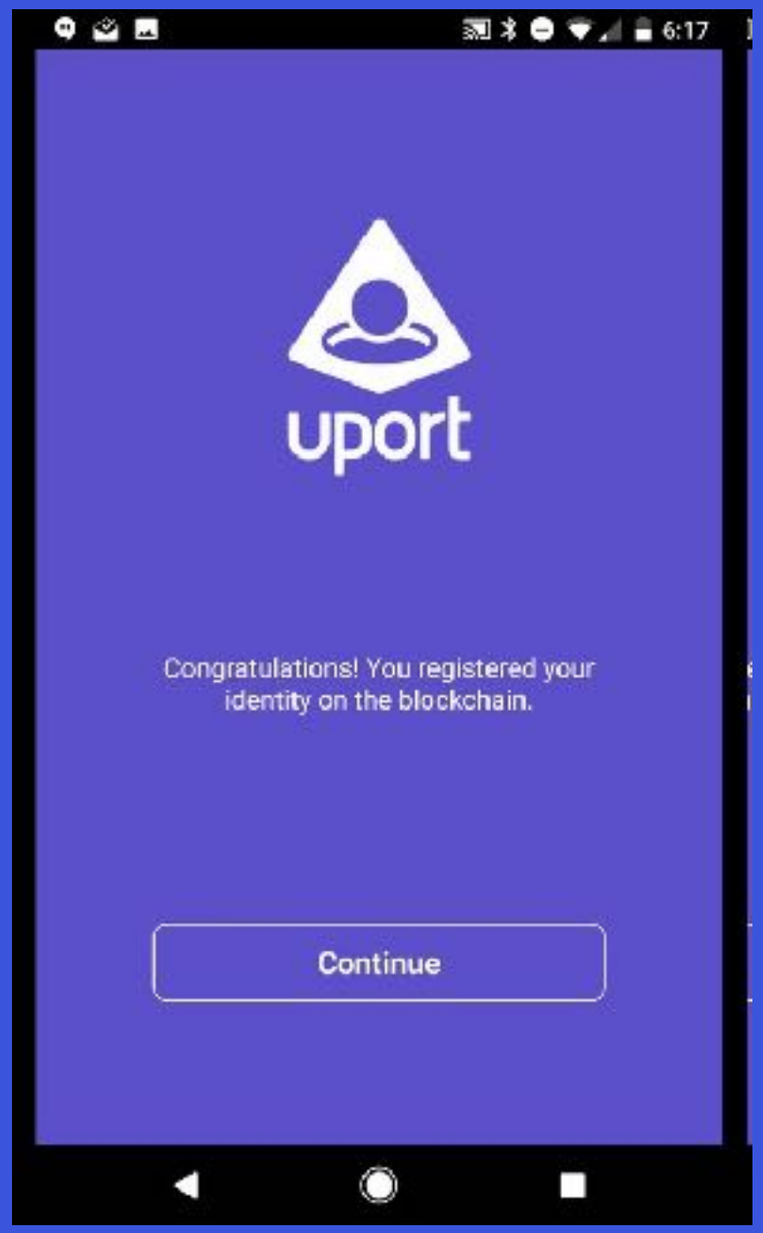
VERES ONE

sovrin

https://sovrin.org/wp-content/uploads/2017/06/The-Inevitable-Rise-of-Self-Sovereign-Identity.pdf

uPort

It doesn’t remove the need for trust in 3rd parties but users can choose the 3rd parties they want to trust.



uPort installation

- Only supported by iPhone and Android
- Go to https://www.uport.me (scroll down and look for your platform link)
- Or simply search in your App store for Mac or Google store for Android

Voting system

- Requirements
- The agile way
- Architecture
- Technology selection
- UI Design
- Proof of concepts
- Work planning
- Development

Requirements

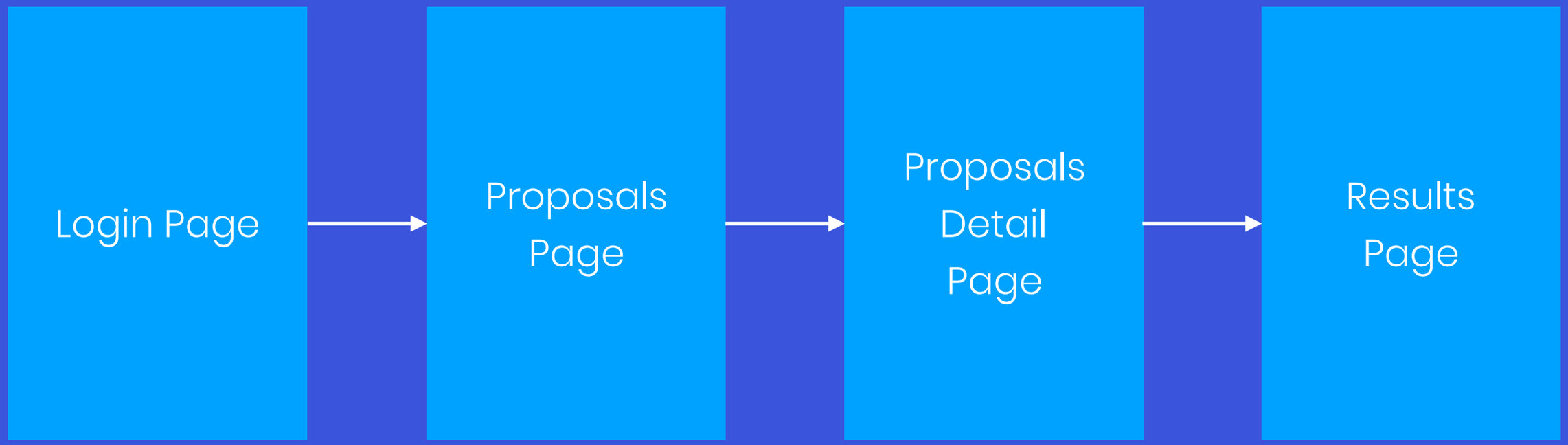
“Agile” way

“**As a** member of a DAO

I need to have a platform that allow me to vote for proposals

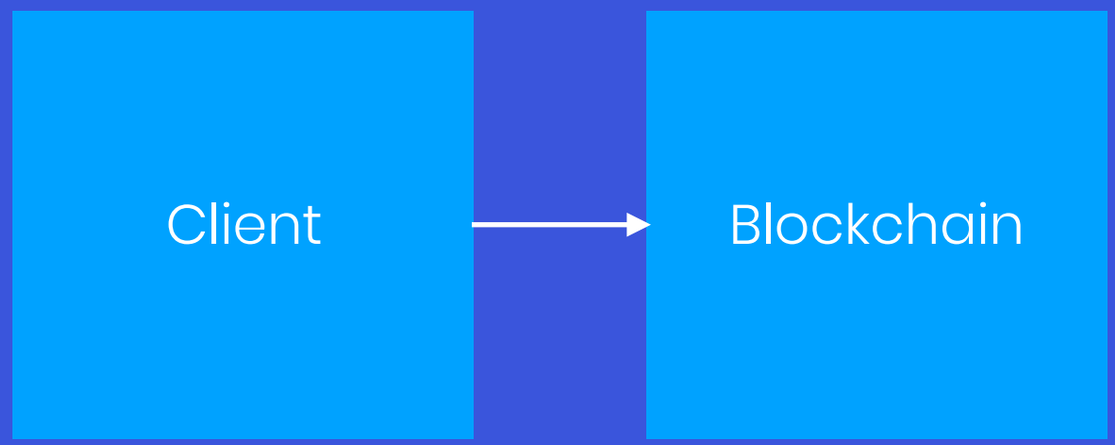
So that I can decide the actions of the organisation”

Our vision



1. Members of the DAO login to the platform by using uPort
2. Logged in members must be able to see the current proposals
3. Members should be able to vote for their favourite one (max 1 time)
4. Members can post their proposal.
5. Members should be able to see the winner proposal

Our started architecture



Technologies

- Web?
  - React, Angular, Vue, Knockout
- Mobile?
  - Xamarin, Ionic, React native
- Blockchain?
  - Private vs public
  - Ethereum?

Recommendations

- Do Proof of concepts
- Check the state of the dependencies and libraries
- Try to abstract which functionalities would be critical
- TODO: Include image that business can fail

React native limitations

ethereum / web3.js

Watch 310 | Unstar 4,819 | Fork 1,267

Code | Issues 387 | Pull requests 27 | Projects 1 | Wiki | Insights

web3 import in react-native #1022

New issue

Open imflavio opened this issue on 6 Sep 2017 · 21 comments

imflavio commented on 6 Sep 2017

```
// Versions:
"web3": "1.0.0-beta.18"
"react-native": "0.46.4"
```

```
// Error:
Bundling `index.android.js`  0.0% (0/1110), failed.
Error: EISDIR: illegal operation on a directory, read
    at Object.fs.readSync (fs.js:681:18)
    at Object.readSync (ProjectName/node_modules/graceful-fs/polyfills.js:138:28)
    at tryReadSync (fs.js:542:20)
    at Object.fs.readFileSync (fs.js:585:19)
    at Module._readSourceCode (ProjectName/node_modules/metro-bundler/build/node-haste/Modu
    at Module._getCacheProps (ProjectName/node_modules/metro-bundler/build/node-haste/Modul
    at Module._readFromTransformCache (ProjectName/node_modules/metro-bundler/build/node-ha
    at Module.readCached (ProjectName/node_modules/metro-bundler/build/node-haste/Module.js
    at ResolutionRequest.preprocessModule (ProjectName/node_modules/metro-bundler/build/nod
    at ResolutionRequest._preprocessPotentialDependen
```

Once I import web3 in my project I get this error, not sure h

👍 3

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Notifications

Subscribe

binhnd-socicom commented on 20 Sep 2017

dougbacelar commented on 25 Oct 2017

I created a step-by-step simple guide on how I set up web3.js with Create React Native App:

https://gist.github.com/dougbacelar/29e60920d8fa1982535247563eb63766

👍 4 | 👎 1

Infura limitations

INFURA / infura

Code | Issues 2

Watch events

Closed DTaste open

DTaste comme

Seems that wat
I'm trying to wa
`registerEvents`

It used to work
What happened

Thanks (and h

egalano comm

Hi **@DTaste** ha
authentication
feature on rops
function under
allowed this bu
nodes behind a
subsequent ge
payload even if
filter was actua

We're working

INFURA / infura

Watch 15 | Star 121 | Fork 8

Code | Issues 27 | Pull requests 0 | Projects 0 | Wiki | Insights

Stateful transaction support #10

New issue

Open egalano opened this issue on 3 Dec 2016 · 9 comments

egalano commented on 3 Dec 2016

@chejazi reported that his test page https://credsign.github.io/terminal/test.html was not working consistently.

After taking a quick look, it appears to be because the page is using newFilter and getFilterLog methods against INFURA. These transactions require that the getFilterLog request gets routed to the same node on which the Filter was installed. We need to come up with our solution for these kinds of stateful transaction methods and how we want to support them.

Some ideas include, installing all filters across all nodes by hooking the newFilter call at the ferryman level and only returning success after installation across the cluster is complete. Or a more traditional approach would be to enable some sort of session header and enable sticky sessions across ferryman and RPC nodes. The problem with this is how to handle RPC node failure or autoscaling events which would clear out the installed filters.

@wuehler @hermanjunge @maurycyp

chejazi commented on 3 Dec 2016

Another idea to mitigate the issue would be to support websocket connections once they're reintroduced into web3. I heard from the gitter channel they'll be included in the next major version.

Assignees

No one assigned

Labels

tracked

Projects

None yet

Milestone

No milestone

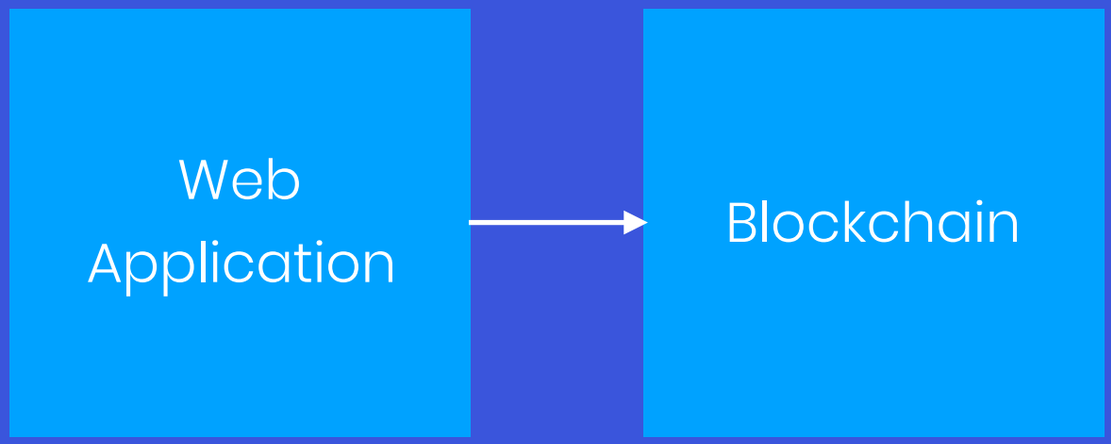
Notifications

Subscribe

You're not receiving notifications from this thread.

10 participants

Technologies selected



- React
- Ethereum
- uPort -> Rinkeby network

UI Design


DAMMIT, JIM, I'M A
PROGRAMMER
NOT A DESIGNER
memegenerator.net

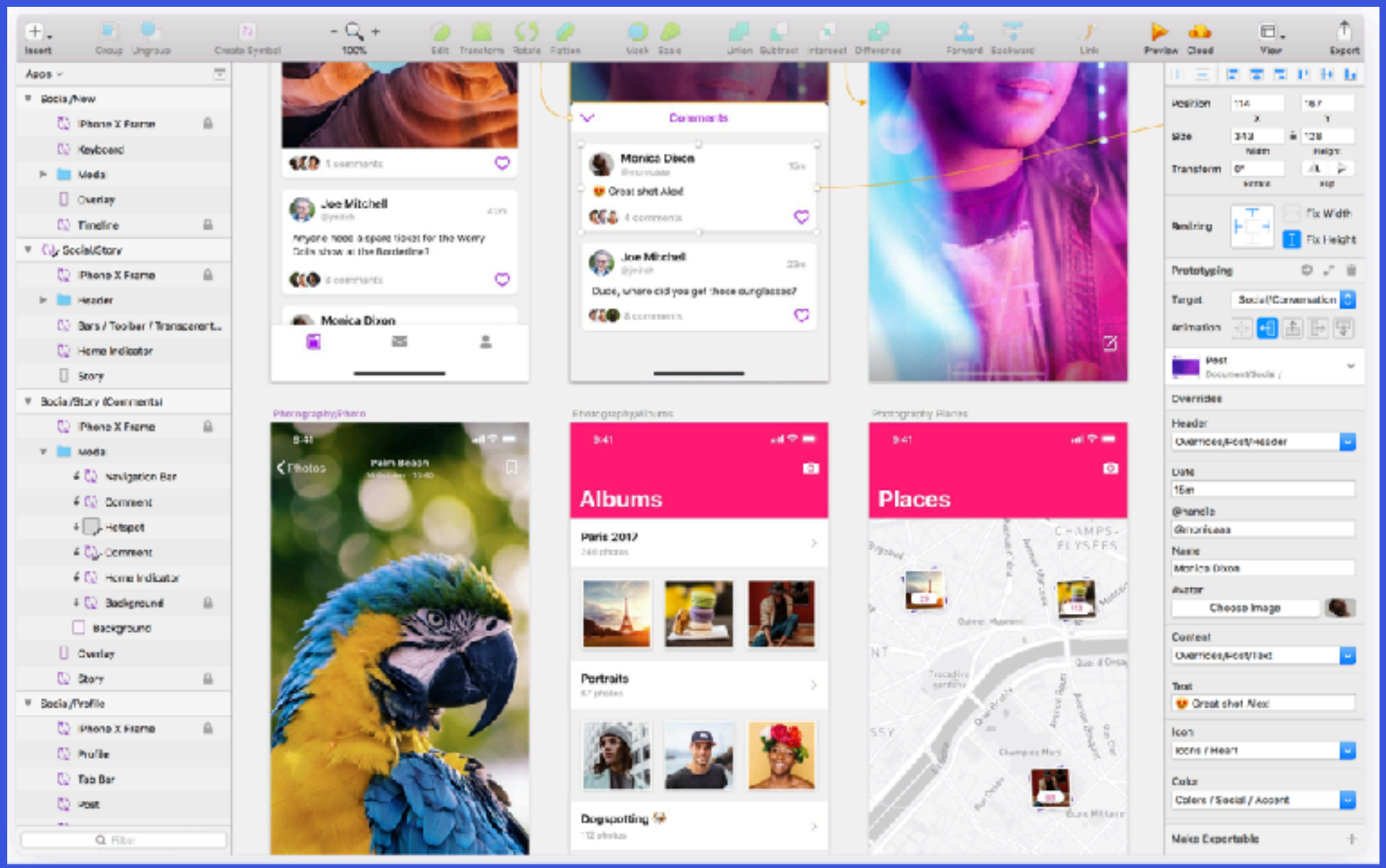
UI Design

- Get inspiration
    - Pinterest
    - Create a board
    - Search
    - Pin and repeat

Sketch

- Paper and pen first
- Sketch App
- Sketch App Resources

Sketch App

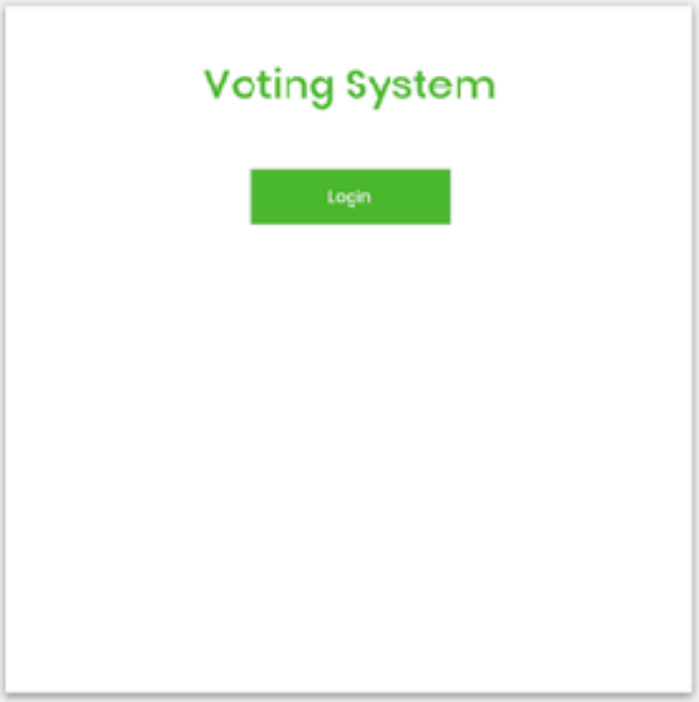

Insert
Group
Ungroup
Create Symbol
100%
Edit
Transform
Rotate
Flatten
Mask
Scale
Union
Subtract
Intersect
Difference
Forward
Backward
Link
Preview
Cloud
View
Export
Apps
Socia/New
Socia/Story
Socia/Story (Comments)
Socia/Profile
Comments
Monica Dixon
Great shot Alex!
Joe Mitchell
Dude, where did you get those sunglasses?
Photography/Photo
Photography/Albums
Photography/Places
Albums
Paris 2017
Portraits
Dogspotting
Places
Prototyping
Target
Animation
Overrides
Header
Date
Name
Avatar
Choose Image
Content
Text
Great shot Alex!
Icon
Color
Make Exportable

Recommendation

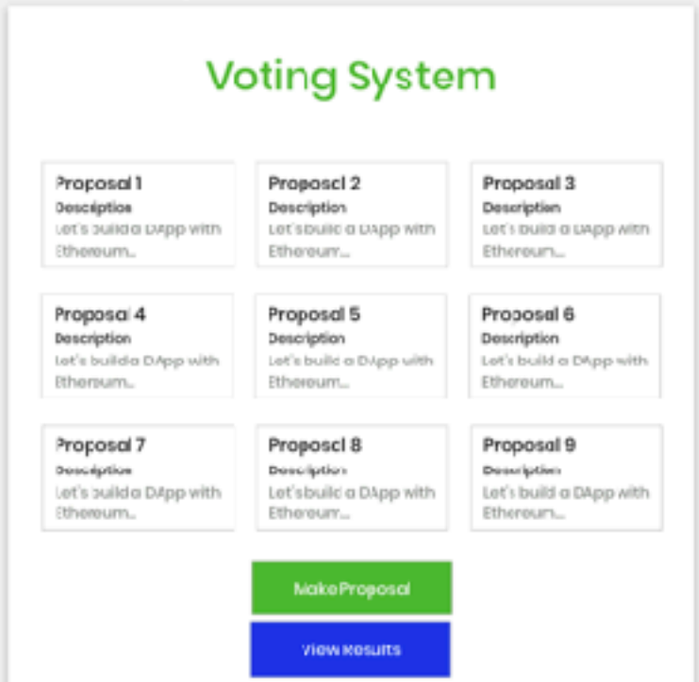
- Think in tasks, what do users want to accomplish?
- What is important for your users?

Voting System UI

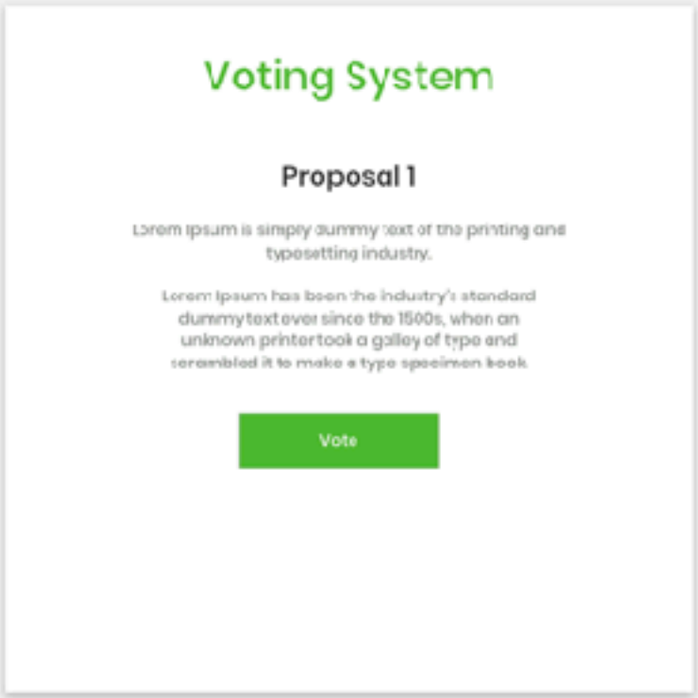
login-page



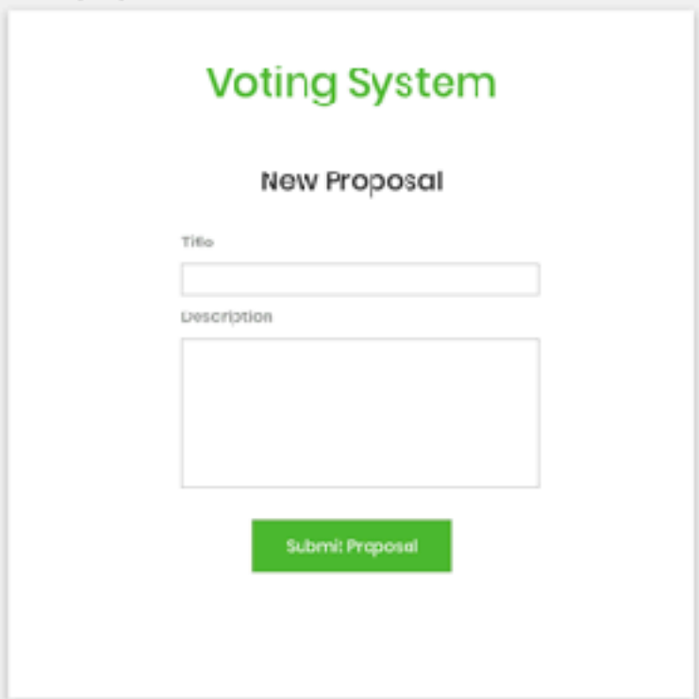
proposals-page



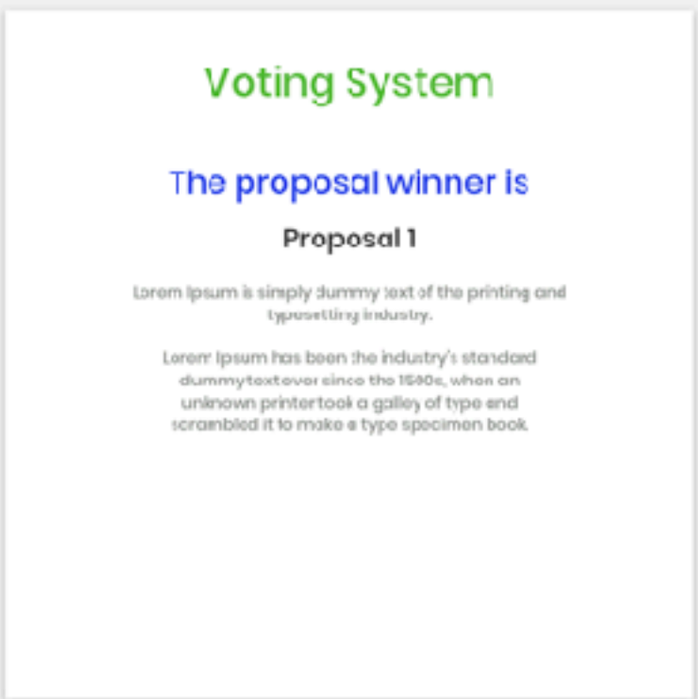
proposal-detail



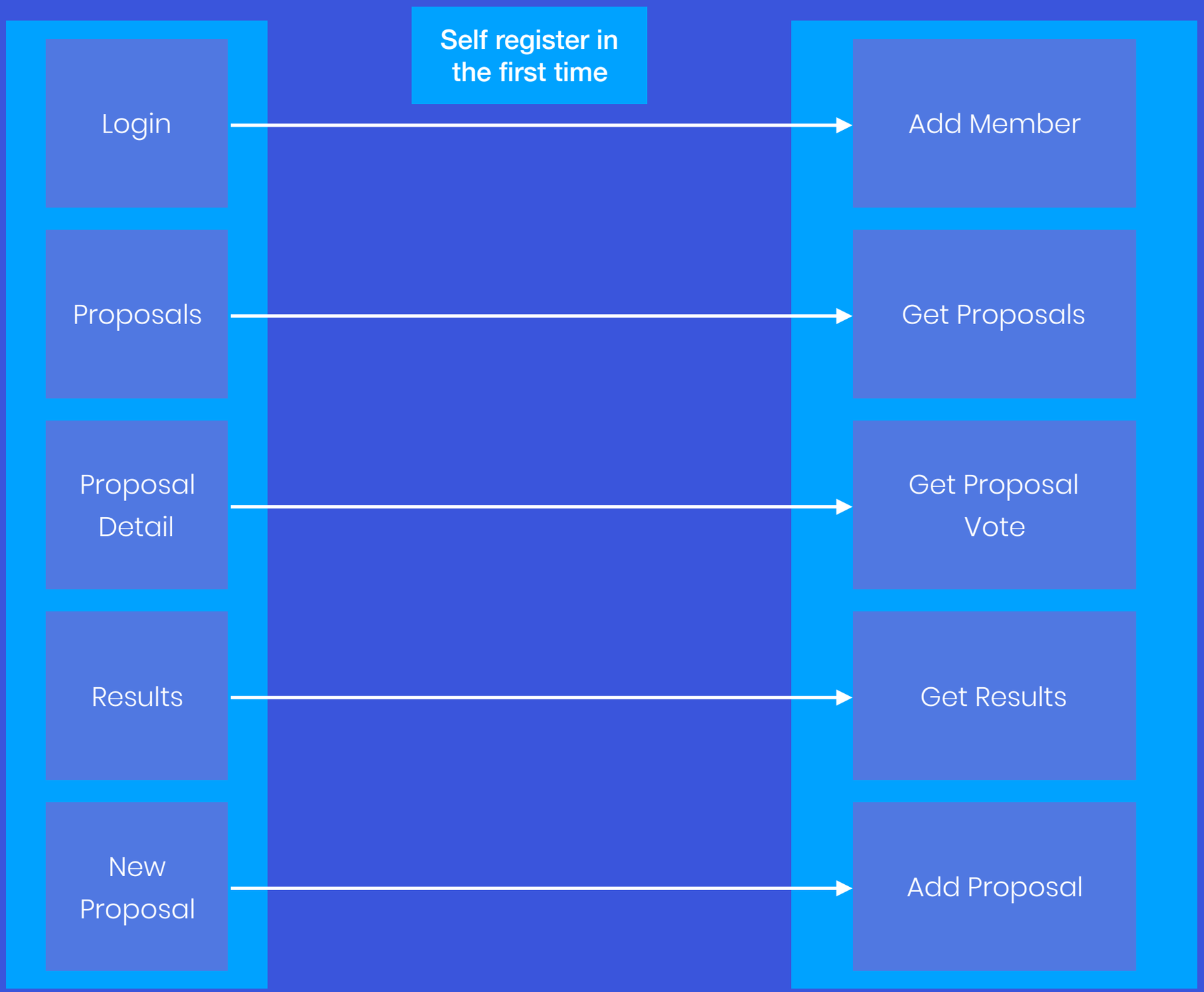
new-proposal



results



Components interaction

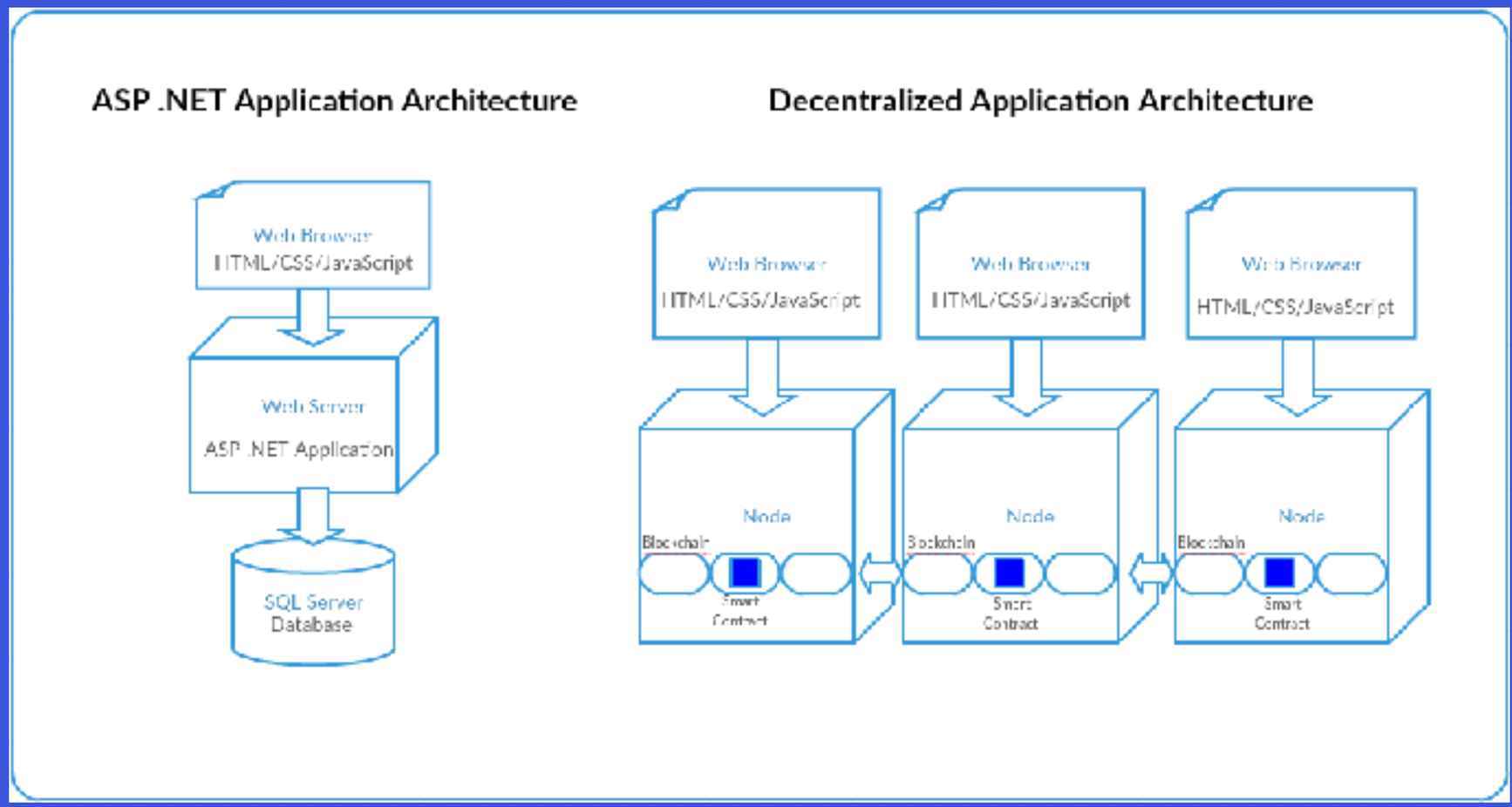


Smart contracts as services

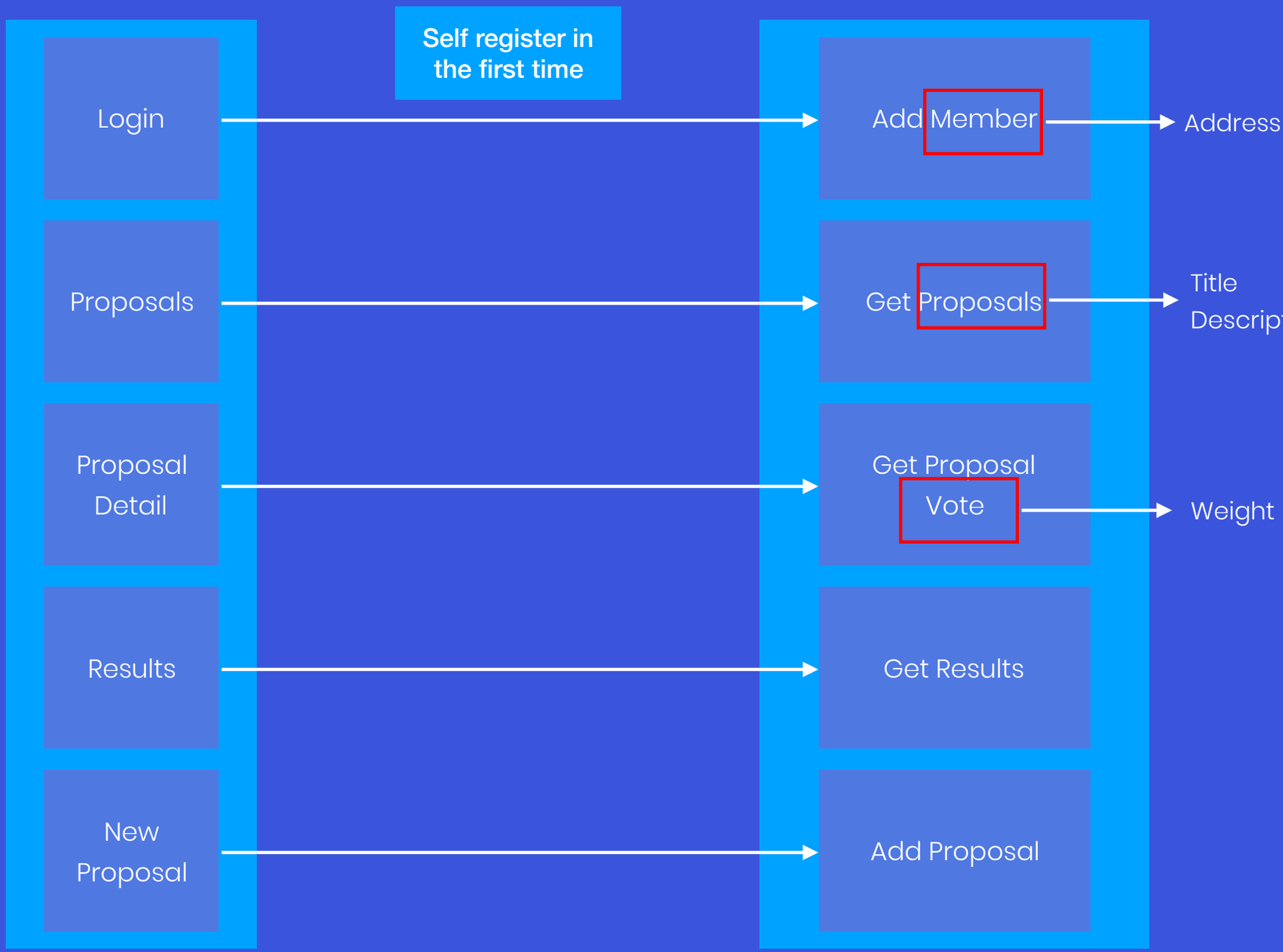
- a smart contract is a unit of functionality, the public functions exposed by their Solidity contracts is the API, and their public address is the identifier”

Monax guide

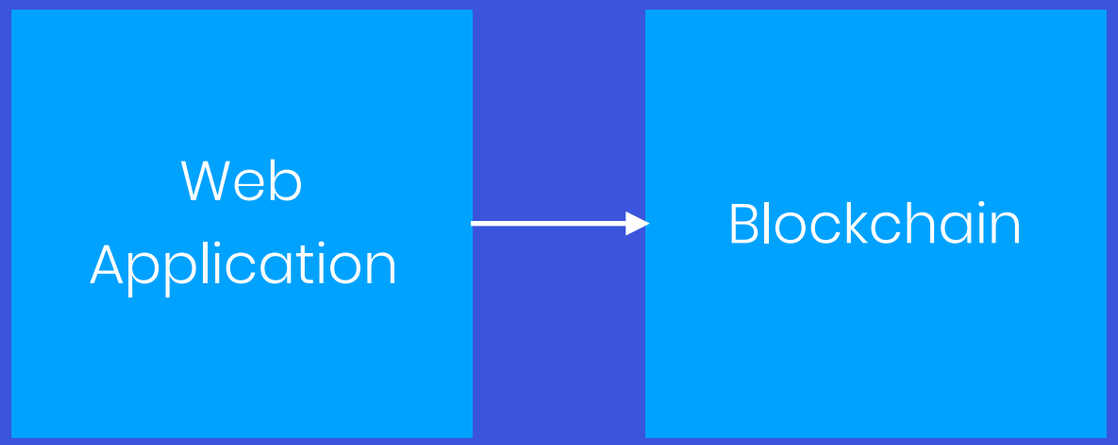
Apps vs DApps



A little abstraction



Components to develop



Setup Blockchain

- Ganache
- Create VM
- Docker
- Infura

Development setup

- Create a special folder in your equipment
- Open a terminal and move to that folder

Node.js Installation

- Go to https://nodejs.org/en/download/
- Download and install it
- Once it has finished test by running
- > node -v
- > npm -v
- In both cases we should see the version installed

Ganache-cli installation

- > npm install -g ganache-cli
- Now, let's validate if our installation was correct by running in our terminal
- > ganache-cli

http://truffleframework.com/docs/ganache/using

Testing Ganache with Remix

- > Open Remix
- Try this contract
- Change environment to http://localhost:8545
- Create contract
- Test the function “say”

Docker installation

- https://docs.docker.com/docker-for-mac/install/
- Now, let's validate if our installation was correct by running in our terminal
- > docker
- > docker-machine

Docker machine demo

- Go to your Azure account
- Get Azure subscription id
- > docker-machine create --driver azure --azure-subscription-id <subscription_id> <anyname>
- Example
- docker-machine create --driver azure --azure-subscription-id 82d36beb-b4d4-4b3c-bc5e-473b0197328a vmazure-test

Docker machine demo

- Check your Azure account
- Check docker-machine by running
- > docker-machine ls
- We can ssh to our machine by simply:
- > docker-machine ssh name-you-assigned-before
- Example:
- > docker-machine ssh vmazure-test

Docker machine demo

- Now, let's start ganache in the cloud
- > docker run -it -p 8545:8545 trufflesuite/ganache-cli:latest
- Configure DNS in Azure
- Open port 8545 (In network security group, use priority 1001)
- Change the environment address in Remix for the address of your machine, followed by the port :8545

Docker machine demo

- Test from your terminal with Curl
- > curl -X POST --data '{"jsonrpc":"2.0","method":"web3_clientVersion","params":[],"id":67}' vmazure-test.westus.cloudapp.azure.com:8545
-

Docker hub

- We just setup a docker host
- It means we can run any container on it
- Images are available in docker-hub to make deployments easy
- https://hub.docker.com/r/trufflesuite/ganache-cli/
- https://hub.docker.com/r/ethereum/client-go/

Infura

- Register in Infura
- How to connect
- https://infura.io/docs/gettingStarted/makeRequests

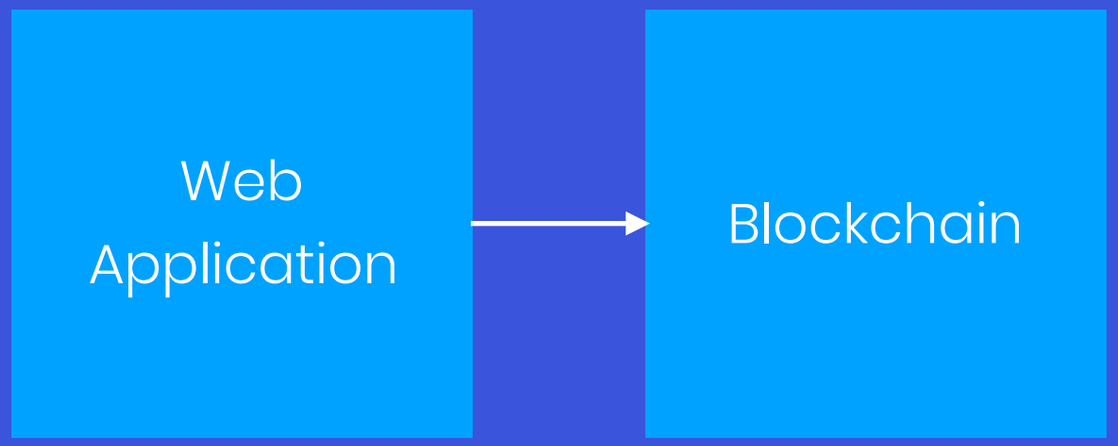
Install Metamask

- Get the Metamask extension
- Please save your seed words !

Setting up Rinkeby

- Change the network for Rinkeby
- Let’s get Ether from a Faucet
- https://faucet.rinkeby.io/

Components to develop



Design our smart contract

Truffle installation

- > npm install -g truffle
- Now, let's validate if our installation was correct by running in our terminal
- > truffle

http://truffleframework.com/docs/getting_started/installation

Truffle

- Truffle makes easy to develop smart contracts and follow a TDD approach.

- It uses Mocha behind the scenes

- It can be integrated easy to any continuos integration process

TDD

- Why TDD?
- The importance of TDD

Our smart contract

- Check our DAO contract
-

Testing our contract

- Start a truffle project
- > truffle init
- Create a organisation contract in our truffle project
- Create a test file named as in the form:
- contractName.test.js
- Let’s write our first test

Testing our contract

- Follow this boilerplate and the sample I use to test the method addMember.
- Do the same with:
  - getWinnerProposal
  - vote
  - addProposal
- In case you have problems with your configuration, use this truffle.js file https://gist.github.com/EdsonAlcala/27b610842980b2c60aec5260fcc6b2eb

Testing our contract

- We need to run
- > truffle test and see the results

Deploy the contract

- Deploy the contract to the Rinkeby network with Remix

Deploy the contract

Front end application

- Install Create React App
- npm install -g create-react-app
- Then create an app
- > create-react-app <name-of-your-app>
- > npm start

User interface

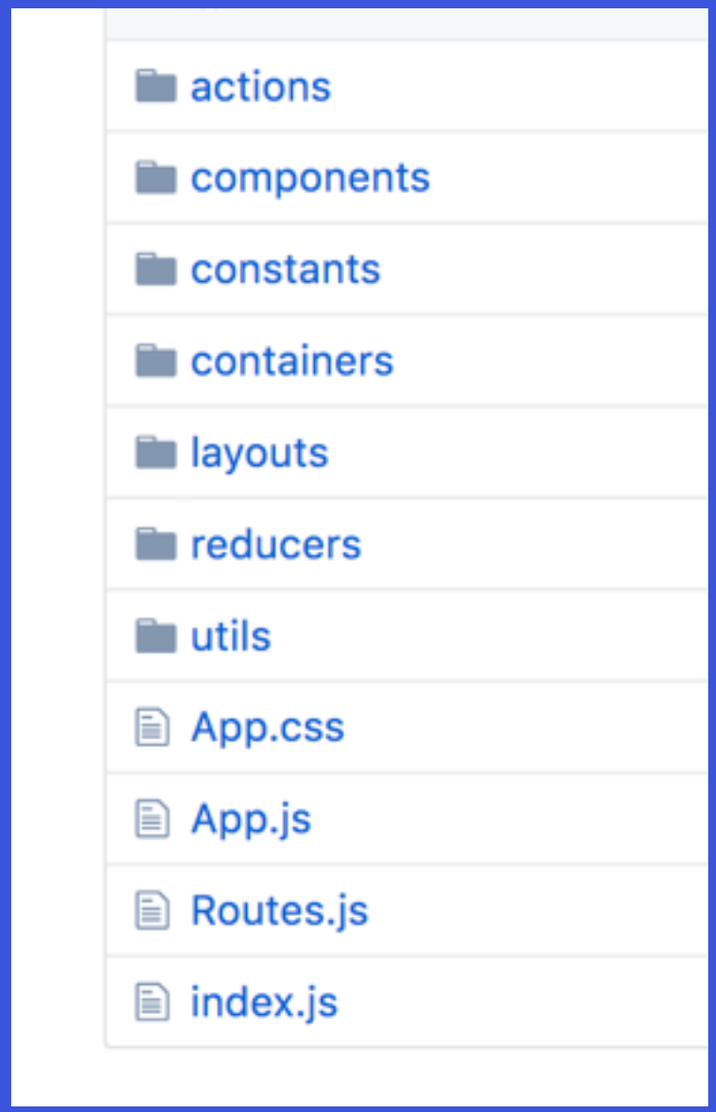
- We will use Semantic-UI
- Framework to develop apps and easy setup front end

Activity

- Explore the Semantic framework
- And for every screen/page of our application. Identity which components can we use?

Front end architecture

- Create folders
- Delete unused files



Navigation

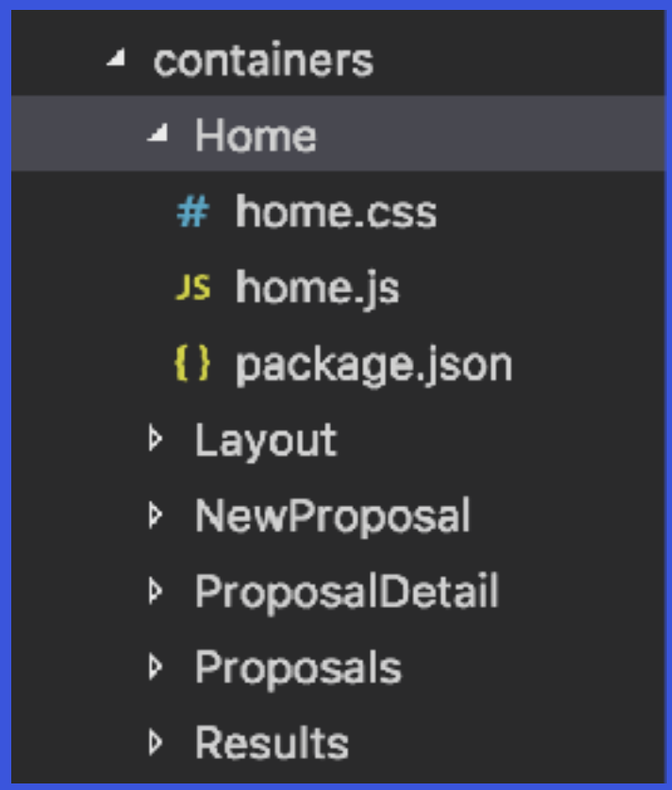
- Install the following components
- npm install —save redux redux-thunk react-router-redux react-router

Navigation

- Create containers
  - Home
  - Proposals
  - Proposal detail
  - Results
  - New Proposal

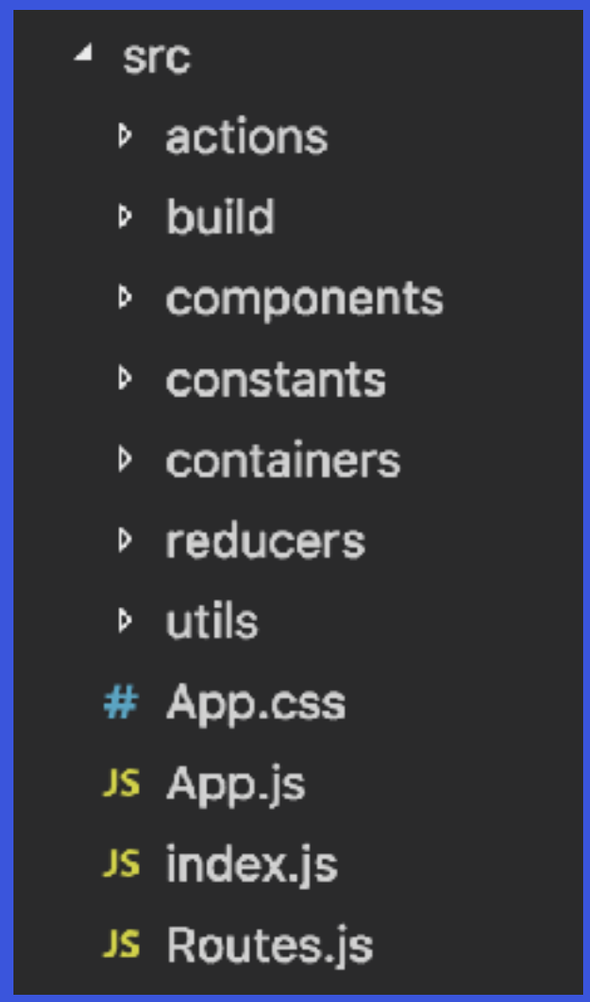
Navigation

- Create containers
  - Home
  - Proposals
  - Proposal detail
  - Results
  - New Proposal



Navigation

- Set Routes
- Check the Routes.js file
- Index.js
- App.js



Include semantic

- Install libraries
- > npm install —save semantic-ui-css semantic-ui-react
- In index.js
- import 'semantic-ui-css/semantic.min.css'

Create a Layout

- Create a Layout container
- Include this code in Layout.js
- Also Layout.css
- Modify Routes.js
- Add withLayout HOC

Login Page

- Define which components do we need
- Define your tasks
- Add static version

Proposals Page

- Define which components do we need
- Define your tasks
- Add static version

Proposal Detail Page

- Define which components do we need
- Define your tasks
- Add static version

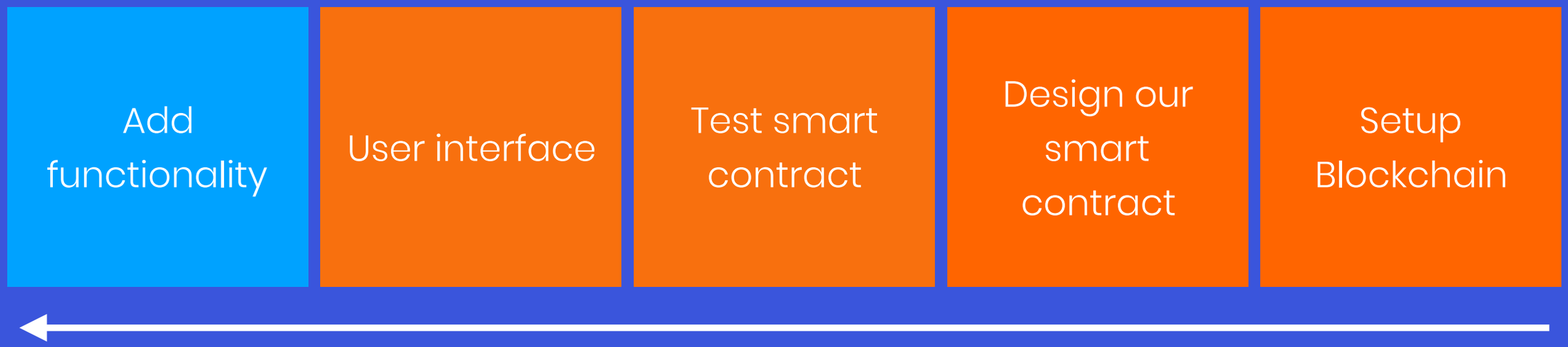
Results Page

- Define which components do we need
- Define your tasks
- Add static version

New Proposal Page

- Define which components do we need
- Define your tasks
- Add static version

Deploy the contract



Setup uPort

- Go to developer.uport.me
- (Need the app installed)
- Sign in
- Set App Name
- Then, expand (click here for you App Code)
- Copy the code and save it securely

Integrate uPort

- Add connector.js to utils
- Install dependencies
- Npm install truffle-contract uport-connect —save
- Modify home.js (gist)
- Test (use console.log to see information)
- npm install —save mind

Add member

- Actions
- Reducers
- Types

Get Proposals

- Actions
- Reducers
- Types

Proposal Detail

- Actions
- Reducers
- Types

Results

- Actions
- Reducers
- Types

New Proposal

- Actions
- Reducers
- Types

Put all pieces together

- We should have a voting system with uPort

Thanks

https://blockchain.topcoder.com/bsic-incubator

Extra resources

Testing tools

> espresso test/demo --verbose

Time: 8.577s

> npm test test/demo/*

Time: 14.666s

Hackathon list

- World virtual hack 450k in Prizes (December)
- BSIC Decentralised Impact Incubator 50k in Prizes (Running)
- Top coder
- DevPost