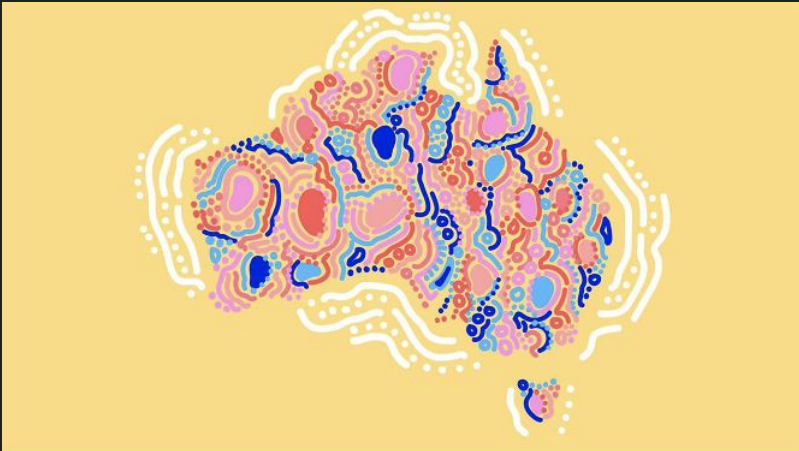


# That Random Raffle Demo!

Where randomness is too important to leave to chance and we are the keepers of our own destiny!



Womindjeka/  
Wominjeka  
Welcome



A close-up photograph of a wooden structure, possibly a part of a traditional dwelling or a ceremonial object. It features several vertical wooden poles. One pole is decorated with a black and white Aboriginal pattern, and another pole has a red and white pattern. The background is blurred, showing a blue sky and a brown ground.

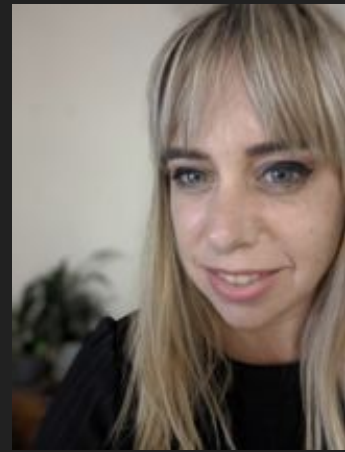
Hi 🖐️ I'm ....

## Alison Haire (Ally)

Developer Advocate @ IBM, Women in STEM ambassador, mechatronics engineer, developer, tech for good enthusiast & Open Saucerer, coder, dog lover, podcaster, global citizen, Lego Masters (AU) enthusiast , aspiring francophone, ex-coffee shop proprietor

Twitter: @developerally

LinkedIn: ally-haire



# The Project!


**Backend:** Solidity v0.6.7 + Chainlink

**FrontEnd:** NextJS + React

**Tooling:**

- Truffle (for blockchain contract deployment),
- Infura (for hosting our blockchain contracts on the Rinkeby testnet),
- Chainlink VRF, Keeper & Price Feed Oracles
- Web3 (for connecting our ethereum blockchain contract backend to React)

Connected



## Chainlink Testnet Raffle

**Contract Address** `0xbA22e59aB1bC776dA9B6a2EE078efD73a0Feb74c`

**Manager** `0x230115404c551Fcd0B6d447DE1DaD3afca230E07`


**Prize Pool** 0.02 ETH


**Round Number** 3

**Enter the raffle!**

Max Entrants before draw: 3


Entered Players

 `0xc7653D426F2EC8Bc33cdDE08b15F535E2EB2F523`

 `0x230115404c551Fcd0B6d447DE1DaD3afca230E07`

Click to Enter!

Entry Cost: 0.01 ETH




### Winners Board

Winner Address	Prize Pool	RoundID
<code>0xCAC120b29Bd386715011bf6927378C0bF75EeF4d</code>	0.03 ETH	1
<code>0xc7653D426F2EC8Bc33cdDE08b15F535E2EB2F523</code>	0.03 ETH	2

<

1


>

Powered by  Chainlink

ID \$7.6634

UNI / USD \$14.8731

XAU / USD \$1815.1100

DOGEY \$ 

ETH / USD \$1764.5544

BTC / USD \$29677.9056

LINK / USD \$13.7894

AUD / USD \$0.7318

AAVE / ETH \$0.1507

BNB / USD

LIVE: <https://chainlink-keepers.vercel.app/>

Code: <https://github.com/DeveloperAlly/Chainlink-Keeper>

# Let's BUIDL...

## Challenge 1:

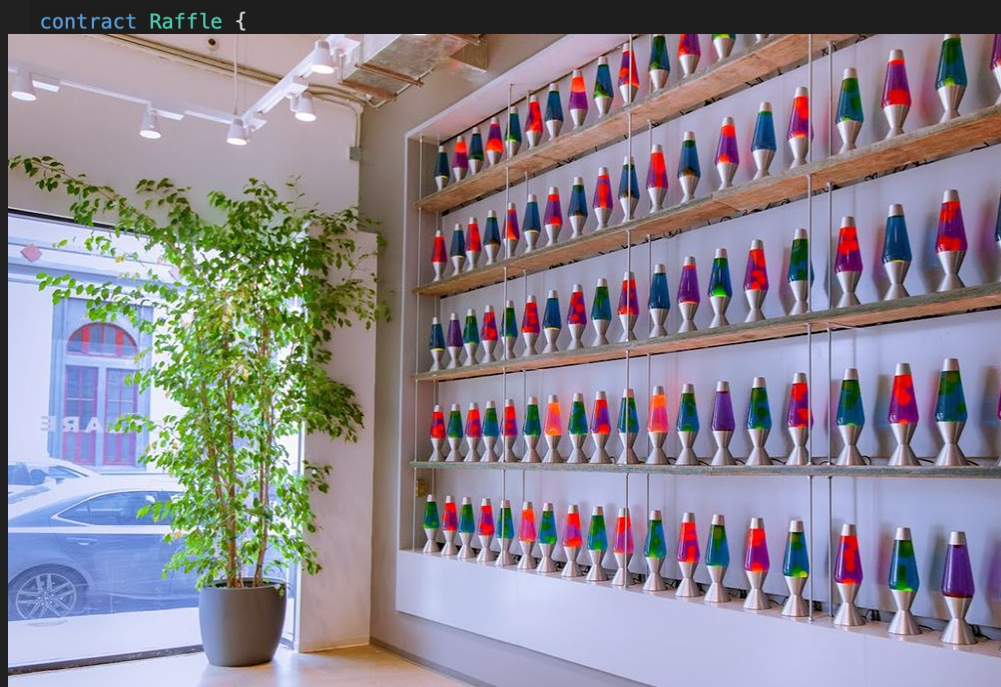
How do we generate good random numbers?

- Because randomness shouldn't be left to chance! XD

## Challenge 2:

How do we trigger this contract to automatically pick a winner?

- Because who's got time for the boring stuff!

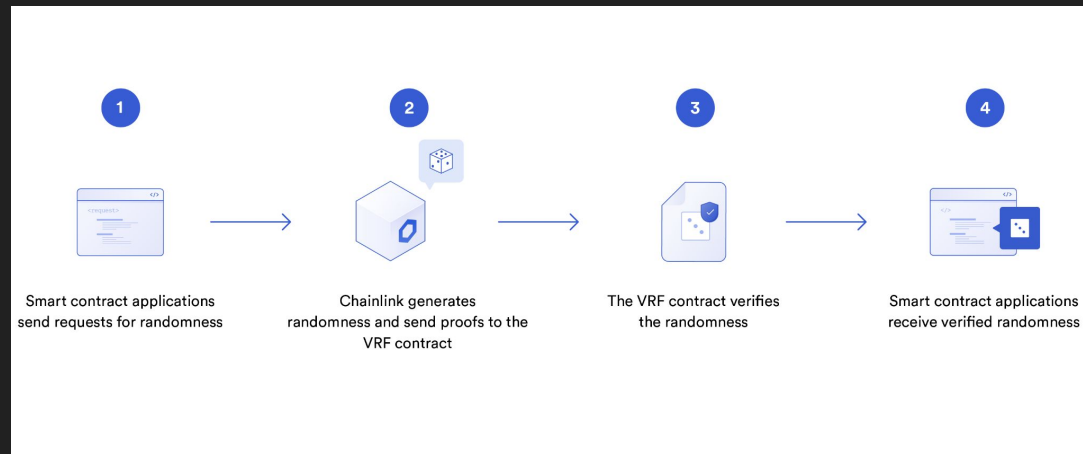


```
contract Raffle {  
  
    randomWinnerNumber = getRandomNumber() % players.length;  
    Winner memory newWinner = Winner({...  
    });  
    winnerRegistry.push(newWinner);  
    players[randomWinnerNumber].transfer(address(this).balance);  
    players = new address payable[](0);  
    prizePool = 0;  
    roundNumber = roundNumber+1;  
}
```

```
function getRandomNumber() private view returns (uint) {  
    return uint(keccak256(abi.encodePacked(now, block.difficulty, msg.sender)  
})
```

# Solutions!

- Chainlink Verifiable Random Number Function
- Chainlink Keeper Network - now you too can be the Keeper of your own destiny!

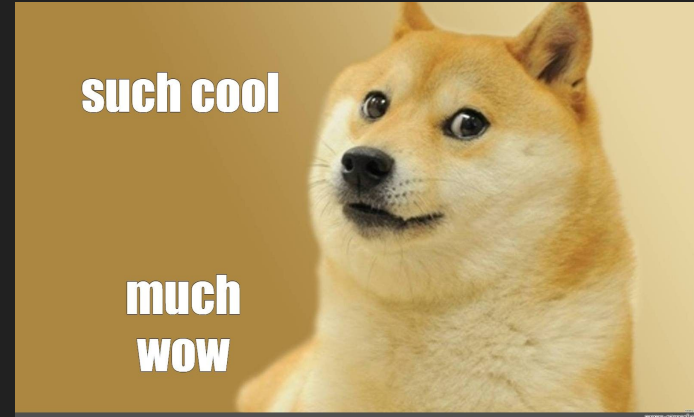
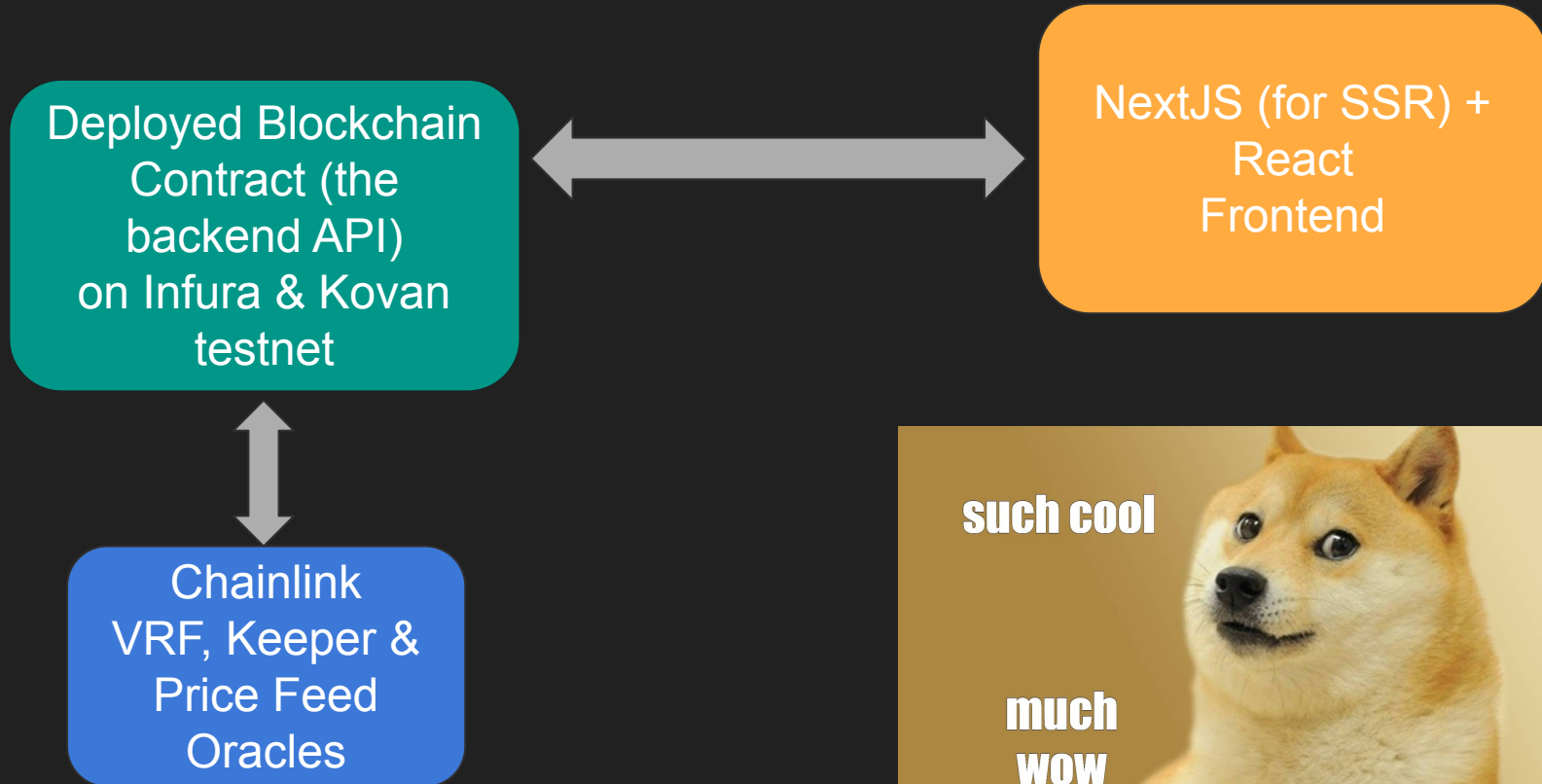


VRF: <https://chain.link/solutions/chainlink-vrf>

Keepers:  
<https://docs.chain.link/docs/chainlink-keepers/introduction/>



# Project Architecture - Final Contract



# Code Demo!

- Add VRF to the randomNumber Function
  - Add the VRF Interface & override function
  - Connect the VRF Interface in our constructor
  - Send LINK to our contract
  - Test it out!
- Add Chainlink Keepers to automatically trigger pick Winner!
  - Add the keeper interface & override functions
  - Add some variables & a trigger point
  - Change our architecture to use the keeper functions
  - Register the contract for Chainlink Keepers
  - Test it out!
- Interactive demo of our final contract! (I hope you have some kovan eth and metamask on your lappy!)
  - <https://chainlink-keepers.vercel.app/>





# What now?

## Possible contract additions..

- Add functionality to the keeper to check the Link balances and automatically top it up in both the contract (VRF use) and the Keeper Node
- Expand this to include multiple prizes, multiple winners or even NFT swapping.
- Other ideas: charity raffles, giving away NFT's, games and gaming (eg a game that uses VRF to set speeds for the characters to run then triggered the payout on completion, voting and achieving quorum, price prediction competitions using CL price feeds and QDT (from CL market), allthethings!

## Chainlink in Production Apps

- <https://blog.chain.link/44-ways-to-enhance-your-smart-contract-with-chainlink/>
  - VRF in production: <https://chain.link/solutions/chainlink-vrf>
  - Keepers in production: eg. Base Protocol
- <https://xangle.io/project/LINK/recent-disclosure/60cbf9bdd94651bc55d90072>



# Resources

Github: <https://github.com/DeveloperAlly/Chainlink-Keepers>

Chainlink VRF: <https://docs.chain.link/docs/chainlink-vrf/>

Chainlink Keepers: <https://docs.chain.link/docs/chainlink-keepers/introduction/>

Chainlink Price Feeds: <https://docs.chain.link/> (sneaky bonus in this!)

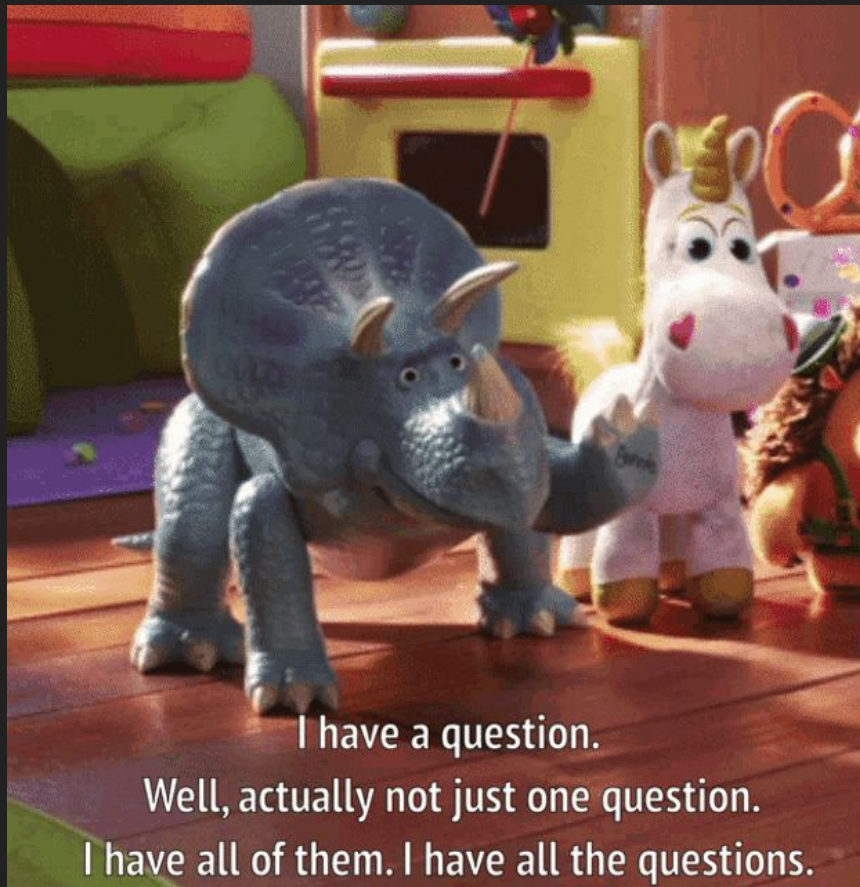
Trufflesuite: <https://www.trufflesuite.com/truffle>

Infura: <https://infura.io/>

NextJS: <https://nextjs.org/>

Web3: <https://web3js.readthedocs.io/>

Annddd... that's all folks....! Any Questions??



**Contact me:**

Twitter: @developerally

LinkedIn: ally-haire

Thanks for coming folks!

Go use those tech superpowers for good!

Want to learn more? -

Join a hackathon or keep

coming to meetups!



# Why Next-react?

I used Next in this project for 3 main reasons:

1. I got a cool free layout that worked really well as a price feed :P
2. It has inbuilt routing
3. I can fetch data before I render the page on the server side and return it in a nice way
4. Speaking of server-side - I can use next SSR to inject the web3 library even if I don't have an extension like metamask installed on my browser without getting CORS issues on my testnet address.
5. I can use it like a data cache so I don't need to React's useState or useEffect or use redux or graphql (graphql is cool though and learning how to use it with blockchain contracts is on my list!)