LESSON 10

NextJS Smart Contract Lottery Full Stack / Front End 16:34:08

https://nextjs.org/ React tabanlı bir frameworktür. React full stack uygulama ve front end uygulama geliştirmek için kullanılna bir frameworktür. Next JS en üstünde bulunur.

React JS blockchainde de popüler olarak kullanılmaktadır. Uniswap ve Avi gibi uygulamarda da kullanılmıştır. Next JS, React ile çalışmayı kolaylaştırmaktadır.

https://www.freecodecamp.org/news/why-use-react-for-web-development

NEXTJS SETUP 16.40.36

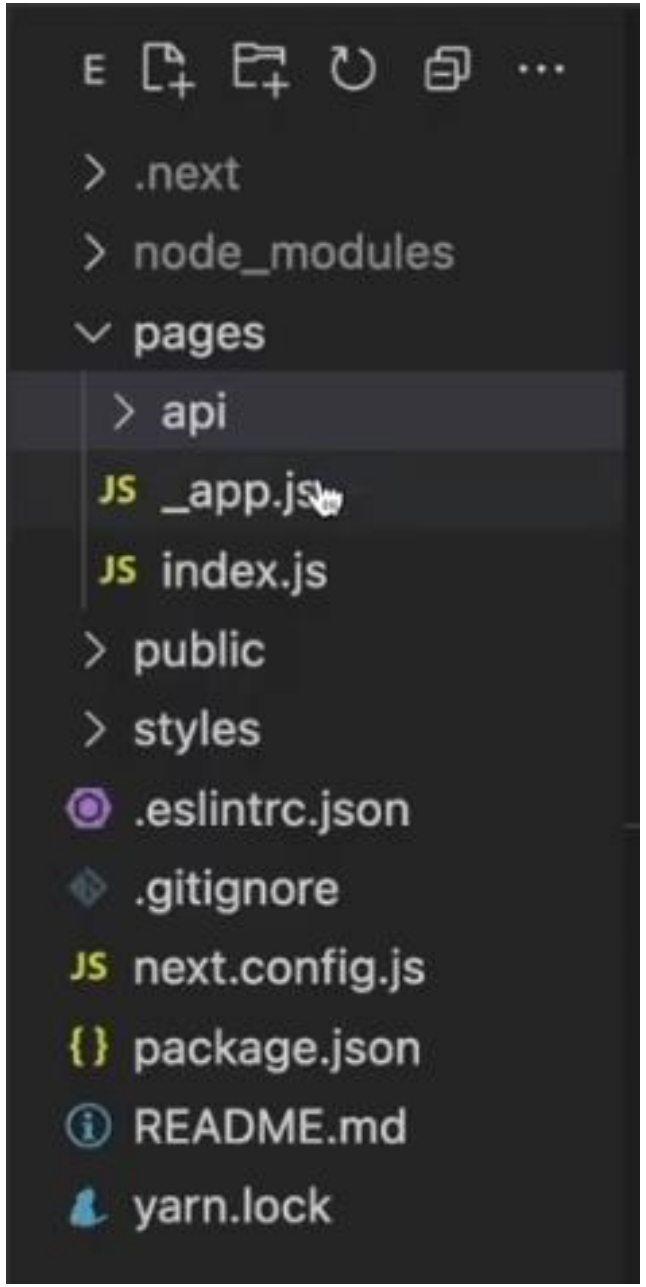
We will create front-end Project outside of hardhat Project folder.

Folder name:
nextjs-smartcontract-lottery-fcc

Start a new Project;

yarn create next-app .

Dot(.) makes folder as Project folder which you are inside. Writing a folder name (yarn cerate next-app myfrontendproject) makes a folder for project



Pages→ diferent pages on our site.

_app.js → to start app. It's entry point for everything.

yarn run dev → starts a web server, opens page in browser.

İndex.js → default page in site

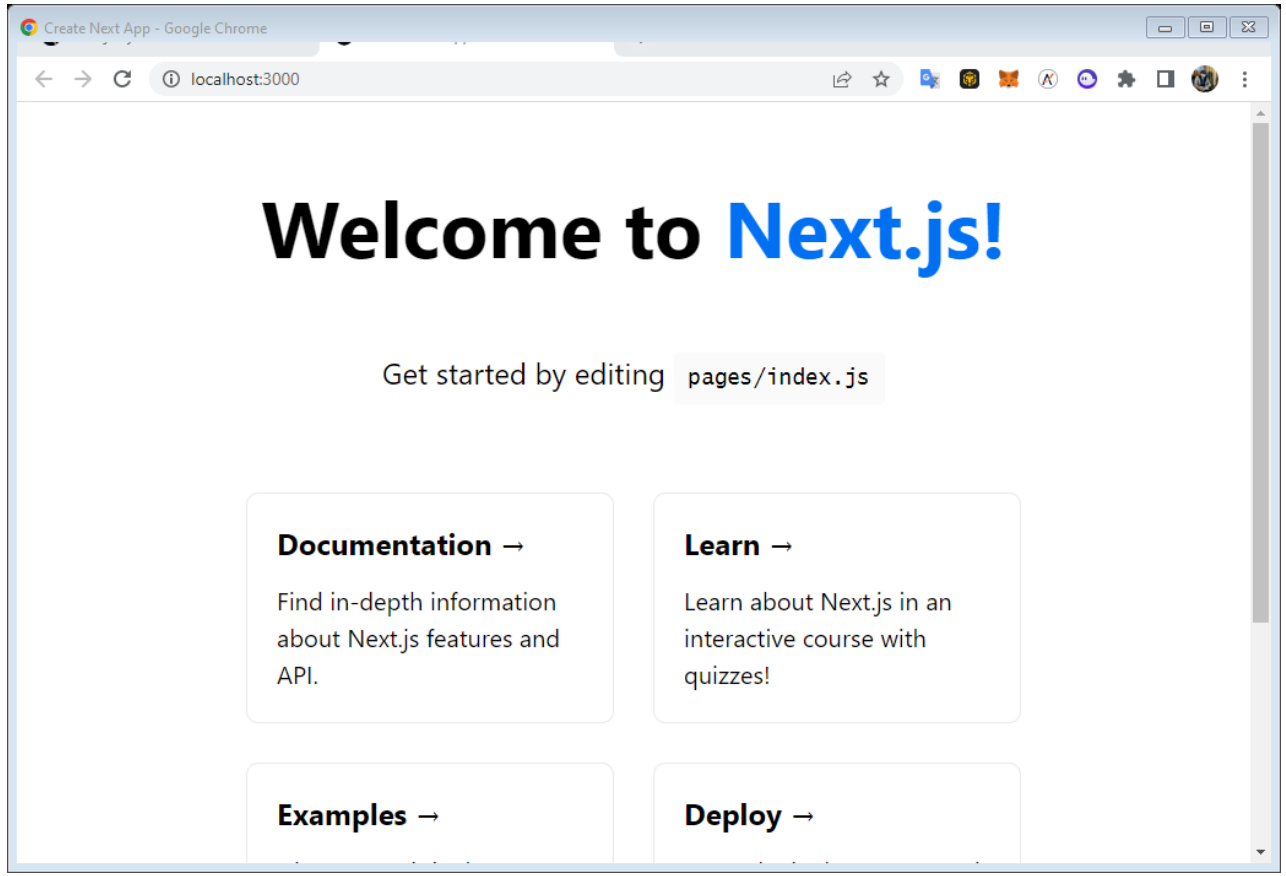
Make a new page test.js
Copy index.js content to test.js, clean all things except below

Run → http://localhost:3000/test you will see the test page on browser

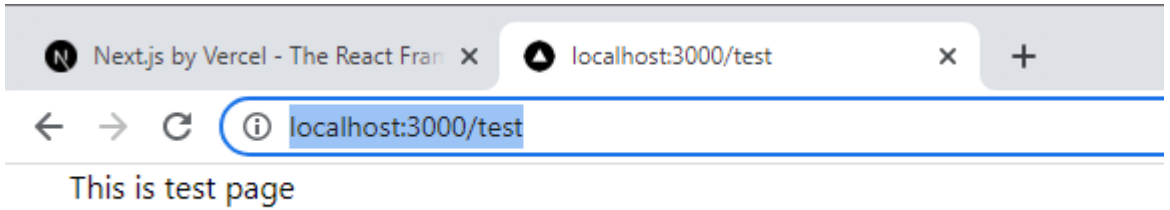
```
export default function Home() {
  return <div className={styles.container}>This
is test page</div>;
}
```

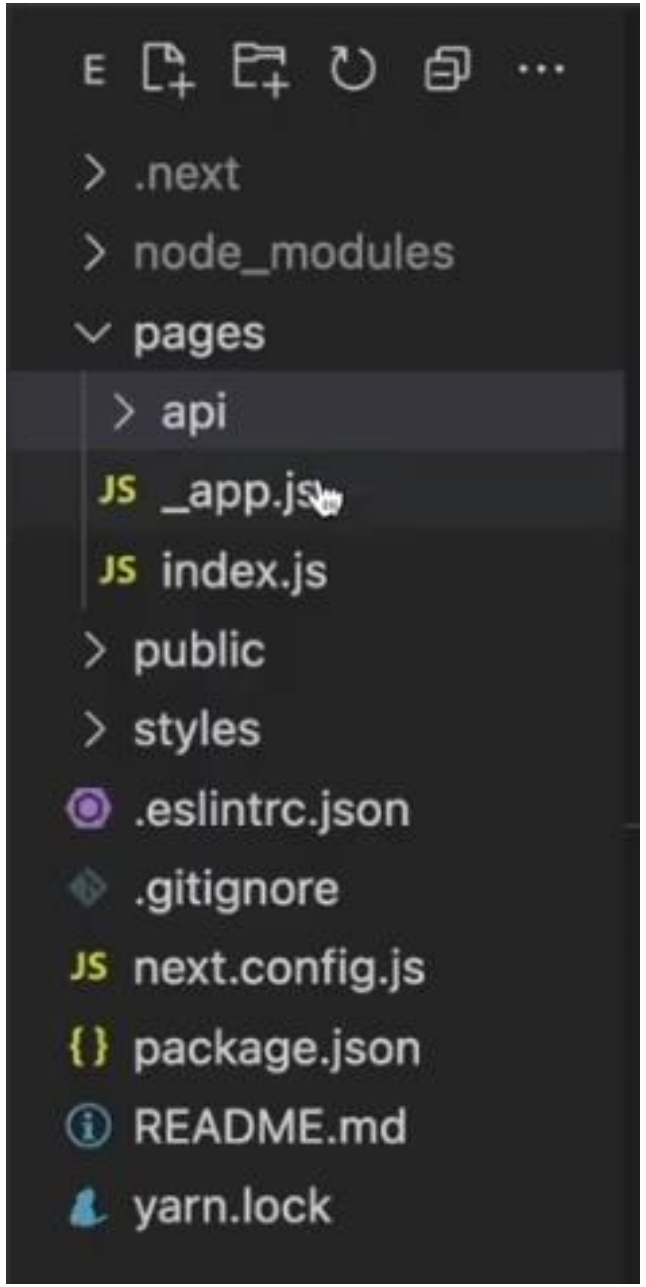
yarn run dev

result



http://localhost:3000/test





The syntax in files is react-syntax or JSX

İmports work with our front end
Require does not work with front end
Node js is not javascript
Backend js is a litte different from front end js

__app.js → is component based

```
 return <Component {...pageProps} />
Loads the index.js in site
```

Api → for http, get, post request

Delete .eslintrc.json

Add files for format and add prettier to auto format our code

.prettierrc

```
{
  "tabWidth": 4,
  "useTabs": false,
  "semi": false,
  "singleQuote": false,
  "printWidth": 99
}
```

yarn add --dev prettier

.prettierignore

```
node_modules
artifacts
cache
coverage*
gasReporterOutput.json
package.json
img
.env
.*
README.md
coverage.json
deployments
.next
```

16.48.50

https://www.w3schools.com/react/react_components.asp

Create **components** folder in root directory. Add Manuel**Header.jsx** file

export default → allows other files to use this function

```
export default function ManuelHeader() {
    return <div>Hi</div>
}
```

İmport in index.js

```
import ManuelHeader from "../components/Header"
```

```
And use as component in <ManuelHeader /> like html tags
```

```
See in localhost:3000 "hi" message
```

HARD WAY

THERE IS DIFFERENT WAYS TO CONNECT WALLET AND DOING OTHER THINGS

ETHERS
….

THIS IS WITH REACT-MORALIS

https://www.npmjs.com/package/react-moralis

yarn add moralis react-moralis → run this command

Note: We don't use dev dependicies. Because we use this for production. Dev dependencies for developer to use in development for help

To enable WEB3 with moralis

index.js M | ManuelHeader.jsx U

```
1  import { useMoralis } from "react-moralis"
2
3  export default function ManuelHeader() {
4      const { enableWeb3 } = useMoralis()
5      return <div>Hi</div>
6  }
7
```

TO ACTIVATE MORALIS PROVIDER

index.js M | ManuelHeader.jsx U | _app.js M

```
1  import "../styles/globals.css"
2  import { MoralisProvider } from "react-moralis"
3
4  function MyApp({ Component, pageProps }) {
5      return (
6          <MoralisProvider initializeOnMount={false}>
7              <Component {...pageProps} />
8          </MoralisProvider>
9      )
10 }
11
12 export default MyApp
13
```

initializeOnMount → to hook server to add more features to server

React Hooks 16.58.44

https://www.w3schools.com/react/react_hooks.asp

Hooks to work with state. When we connected to wallet the page will refresh and will change the content. (this is the one of the things hooks can do)

Const {enableWeb3} = useMoralis() is same with ethers code --→ Await ethereum.request({method:"eth_requestAccounts"})

And this Works with only metamask

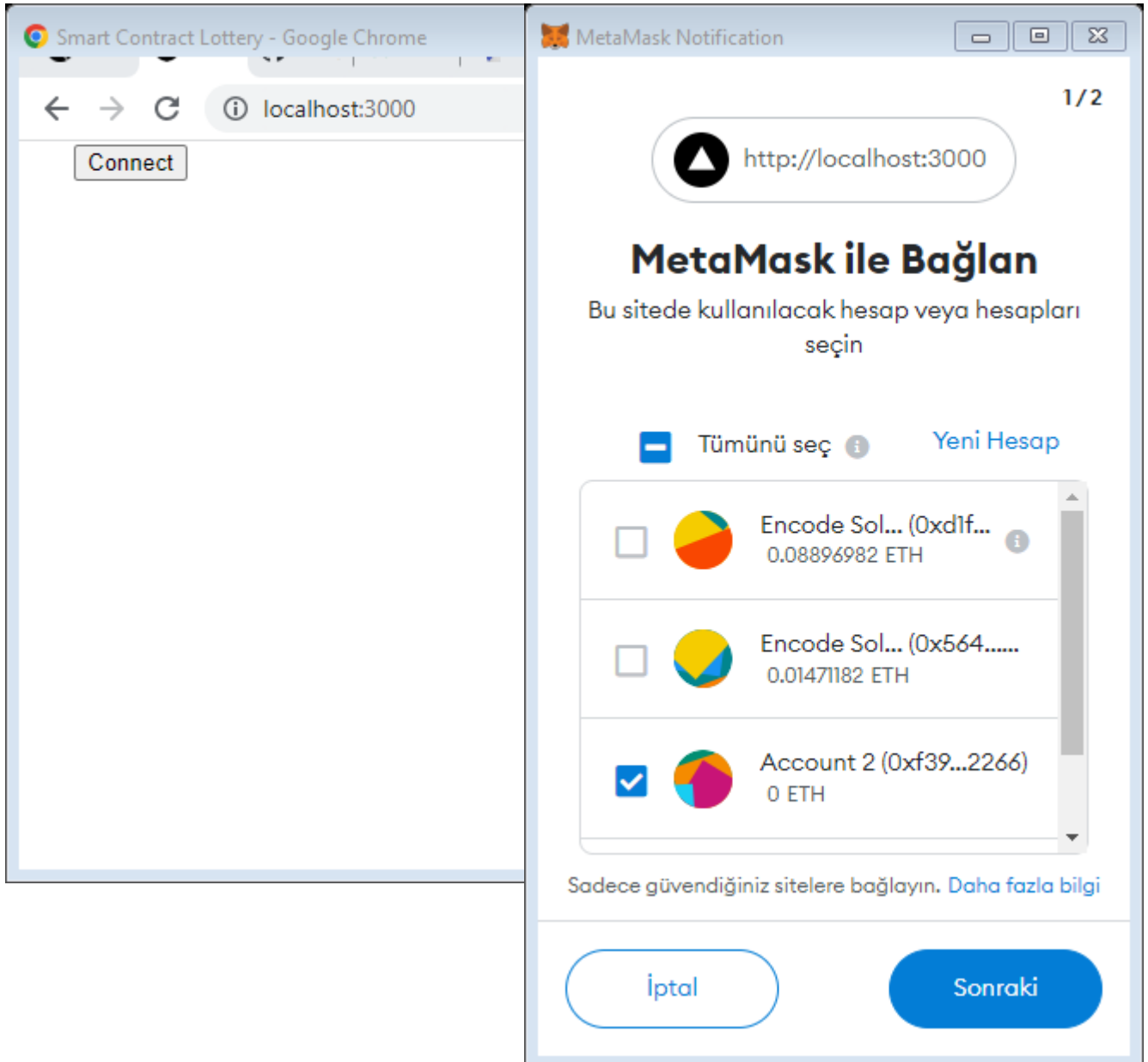
Manuel Header II 17.01.06

Add connect button

index.js M | ManuelHeader.jsx U | _app.js M

```jsx
import { useMoralis } from "react-moralis"

export default function ManuelHeader() {
    const { enableWeb3 } = useMoralis()
    return (
        <div>
            <button
                onClick={async () => {
                    await enableWeb3()
                }}
            >
                Connect
            </button>
        </div>
    )
}
```

Test to connect metamask



When click connect button metamask appears

To change button state when metamask connected and Show account

```
export default function ManuelHeader() {
    const { enableWeb3, account } =
useMoralis()
    return (
        <div>
            {account ? (
                <div>Connected !</div>
            ) : (
                <button
                    onClick={async () => {
                        await enableWeb3()
                    }}
                >
                    Connect
                </button>
            )}
        </div>
    )
}
```

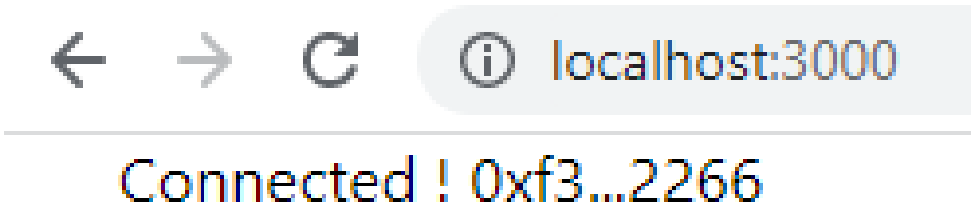
Note : when refresh page, connected turns to connect

To Show account

```
{account ? (
                <div>Connected !
{account}</div>
            ) : (
```

To short account name

```
<div>
                    Connected !
{account.slice(0,
4)}...{account.slice(account.length - 4)}
                </div>
```



useEffect Hook 17.05

This is for; when refresh page to get button / connect state
When refresh page its checks if we are connected

https://reactjs.org/docs/hooks-effect.html

İn ManuelHeader.jsx

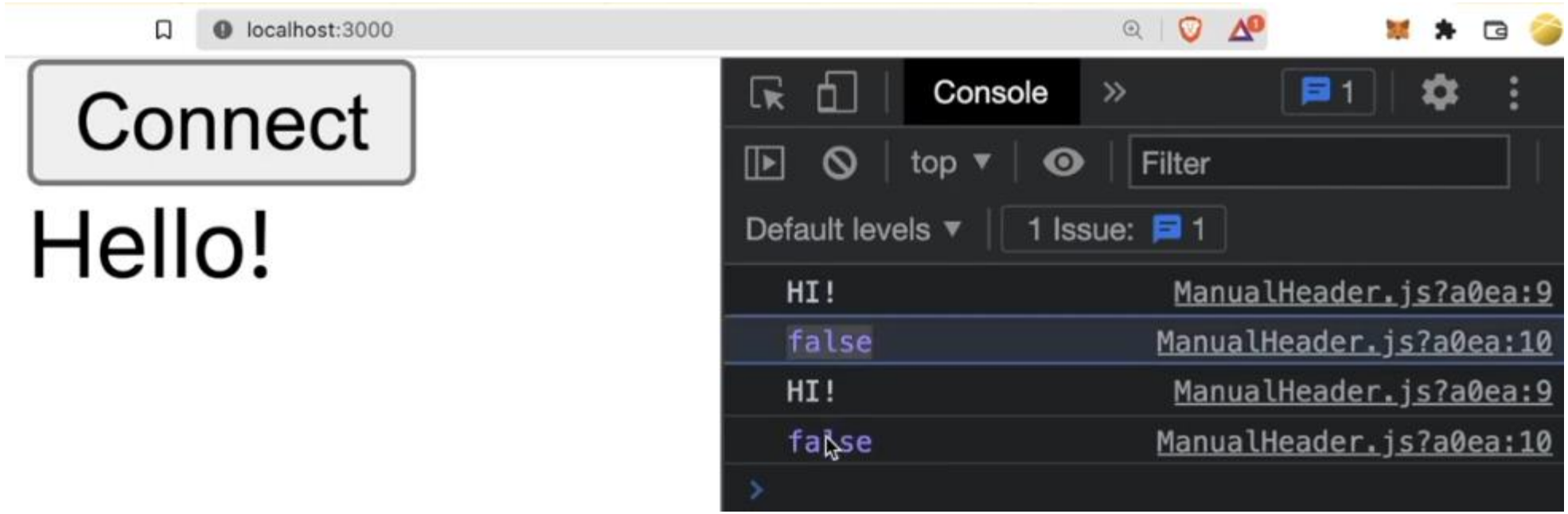
```
import { useEffect } from "react"
```

useEffect(() => {}, []) → takes two parameters. Function is the first parameter, second paremeter is a dependency array. Useeffect keep checking the values in dependency array, and anything in this depen. Rate changes, its going to call the function (first parameter), then render the front end.

For ex: const {enableWeb3, accont, isWeb3Enabled } = useMoralis()

```
useEffect(() =>{console.log("hi") console.log(isWeb3Enabled)}, [isWeb3Enabled])
```

Useeffect follows the change of isWeb3Enabled



We see hi twice. Useeffect automatically run on load, then it will run checking the value.

With out no dependency array, run anytime something re-renders
CAREFUL WİTH THIS! Because then you can get circular renders

Giving blank dependency array, run once on load

……
}, [])

```
export default function ManuelHeader() {
    const { enableWeb3, account, isWeb3Enabled } = useMoralis()

    useEffect(() => {
        console.log("Hi!")
        console.log(isWeb3Enabled)
    }, [])

    return (
```

Browser Local Storage 17.10.28

We use useeffect() when we refresh page, it remembers that we are actually connected
For this we will use isWeb3Enabled.

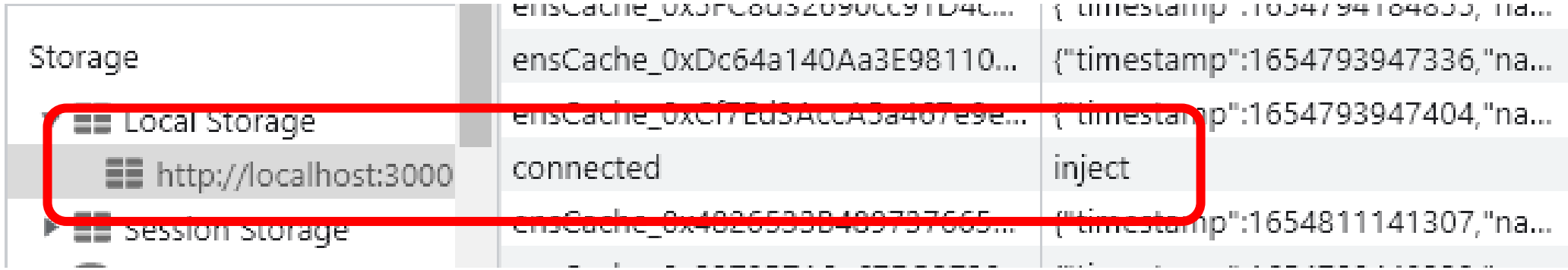
First usage is ;

```
useEffect(() => {
        if (isWeb3Enabled) return

        enableWeb3()
    }, [])
```

But, when disconnected from account, and when refresh browser, the metamaks opens always

To see if we connected to metamask we will use local storage. When hit connect button local storage saves and remembers. For this we will set a new key-value (connected-inject)

```
<button
                    onClick={async () => {
                        await enableWeb3()
                        if (typeof window !== "undefined") {
                            window.localStorage.setItem("connected", "inject")
                        }
                    }}
                >
```




```
useEffect(() => {
        if (isWeb3Enabled) return
        if (typeof window !== "undefined") {
            if (window.localStorage.getItem("connected")) {
                enableWeb3()
            }
        }
    }, [])
```

When refresh browser,
it checks key-value
End automaticall
enables web3

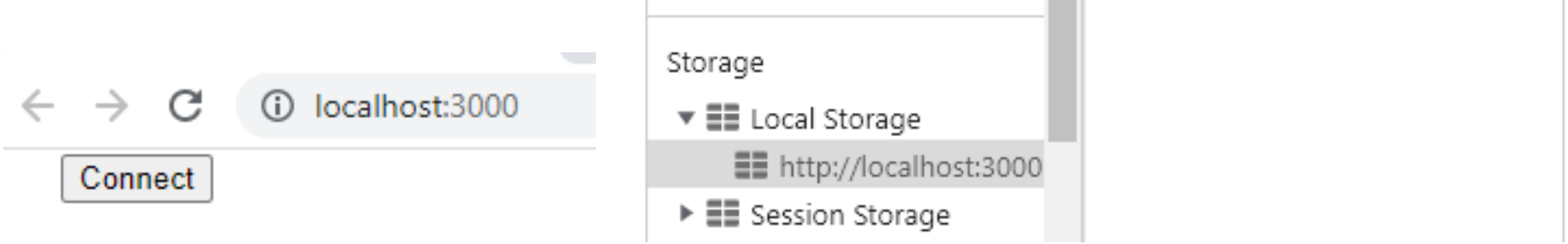
To remember if we were disconnected; we will use another useeffect() and Moralis.onAccountChanged, deactivateWeb3

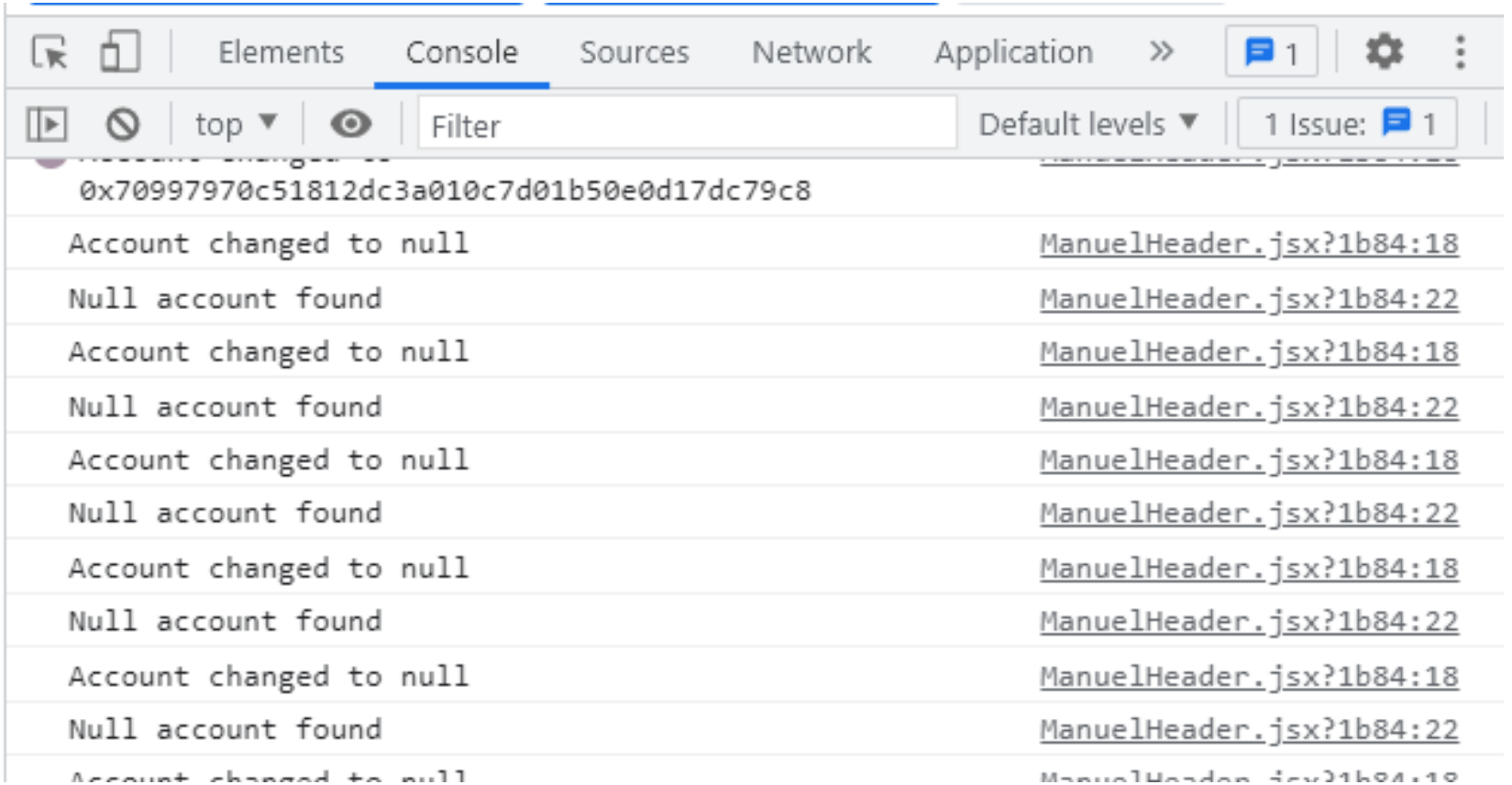
```
const { enableWeb3, account, isWeb3Enabled, Moralis, deactivateWeb3 } = useMoralis()
```

Second useeffect

```
useEffect(() => {
        Moralis.onAccountChanged((account) => {
            console.log(`Account changed to ${account}`)
            if (account == null) {
                window.localStorage.removeItem("connected")
                deactivateWeb3()
                console.log("Null account found")
            }
        })
```

When disconnect it will delete key-value (connected) and deactivate web3


localhost:3000
Connect
Storage
Local Storage
http://localhost:3000
Session Storage

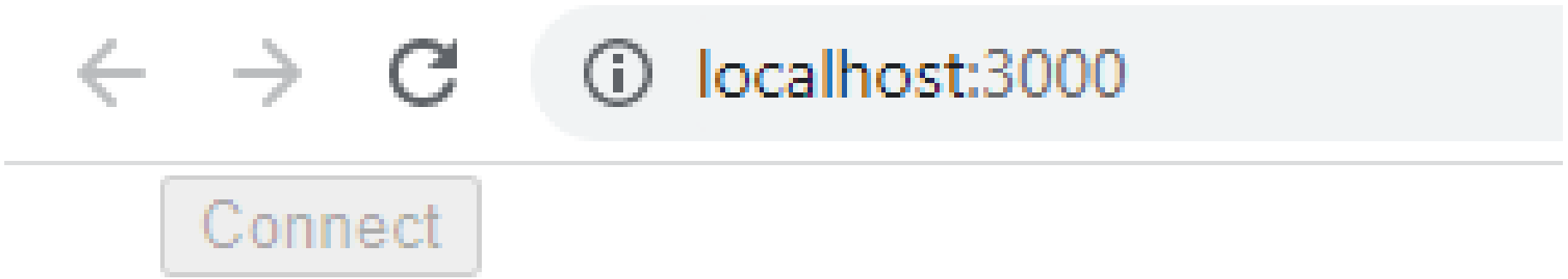

Elements
Console
Sources
Network
Application
1
top
Filter
Default levels
1 Issue: 1
0x70997970c51812dc3a010c7d01b50e0d17dc79c8
Account changed to null ManuelHeader.jsx?1b84:18
Null account found ManuelHeader.jsx?1b84:22
Account changed to null ManuelHeader.jsx?1b84:18
Null account found ManuelHeader.jsx?1b84:22
Account changed to null ManuelHeader.jsx?1b84:18
Null account found ManuelHeader.jsx?1b84:22
Account changed to null ManuelHeader.jsx?1b84:18
Null account found ManuelHeader.jsx?1b84:22
Account changed to null ManuelHeader.jsx?1b84:18
Null account found ManuelHeader.jsx?1b84:22

isWeb3Enable Loading 17.18.20

```
export default function ManuelHeader() {
    const { enableWeb3, account, isWeb3Enabled, Moralis, deactivateWeb3,
isWeb3EnableLoading } =
        useMoralis()
```

İn button properties

```
disabled={isWeb3EnableLoading}
```



When connecting disables button

WEB3UIKIT 17.19.26

EASY WAY EASY WAY EASY WAY

https://github.com/web3ui/web3uikit

yarn add web3uikit

Components/Header.js

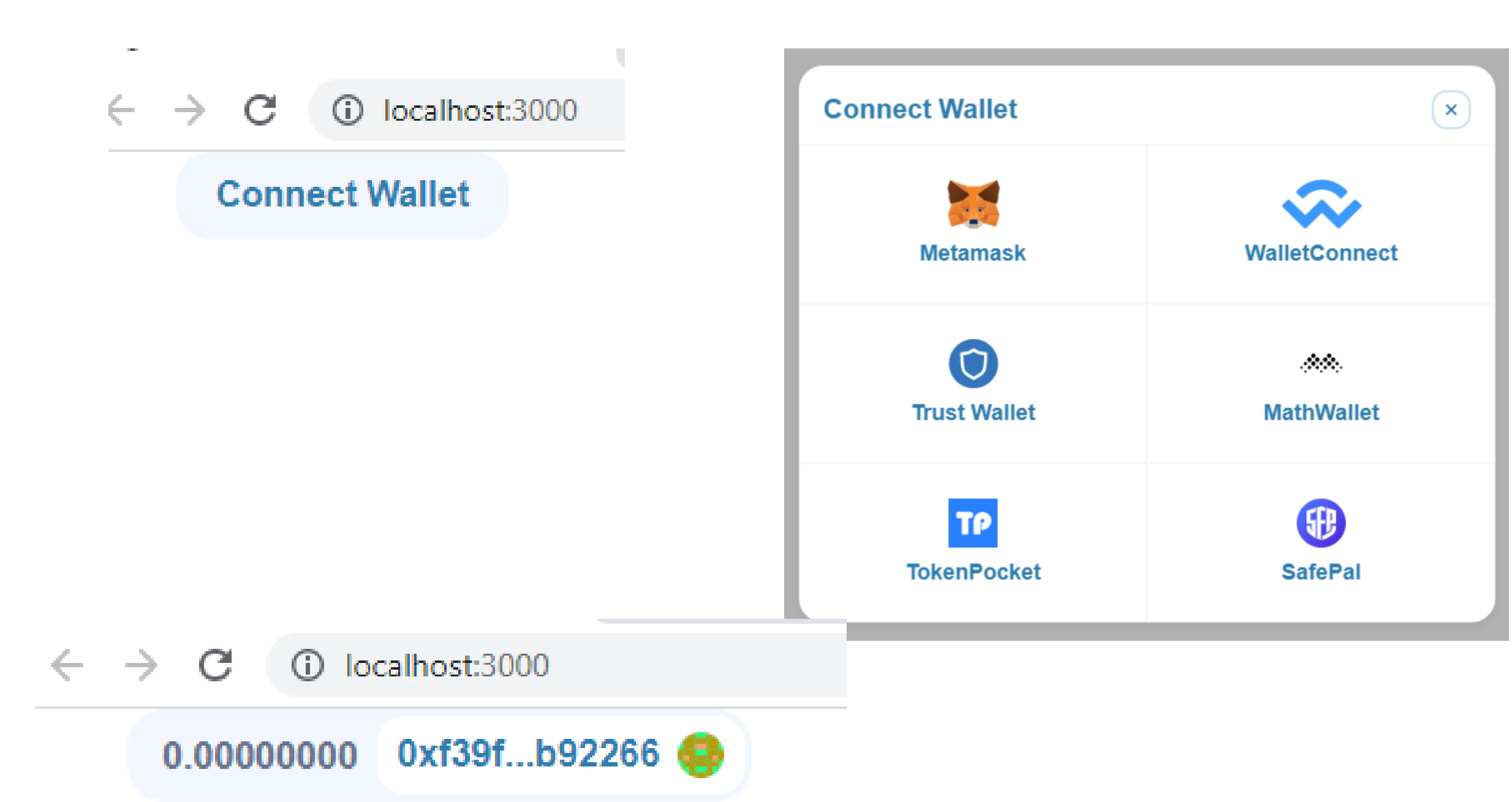
```
import { ConnectButton } from "web3uikit"

export default function Header() {
    return (
        <div>
            <ConnectButton moralisAuth={false}></ConnectButton>
        </div>
    )
}
```

İndex.js

```
//import ManuelHeader from
"../components/ManuelHeader"
import Header from "../components/Header"
```

```
</Head>
            {/*<ManuelHeader /> */}
            <Header />
```

```
Yarn run dev // if shows error run app again
```


localhost:3000
Connect Wallet
Connect Wallet
Metamask
WalletConnect
Trust Wallet
MathWallet
TokenPocket
SafePal
localhost:3000
0.00000000
0xf39f...b92266

Introduction To Calling Functions in NextJS 17.22.25

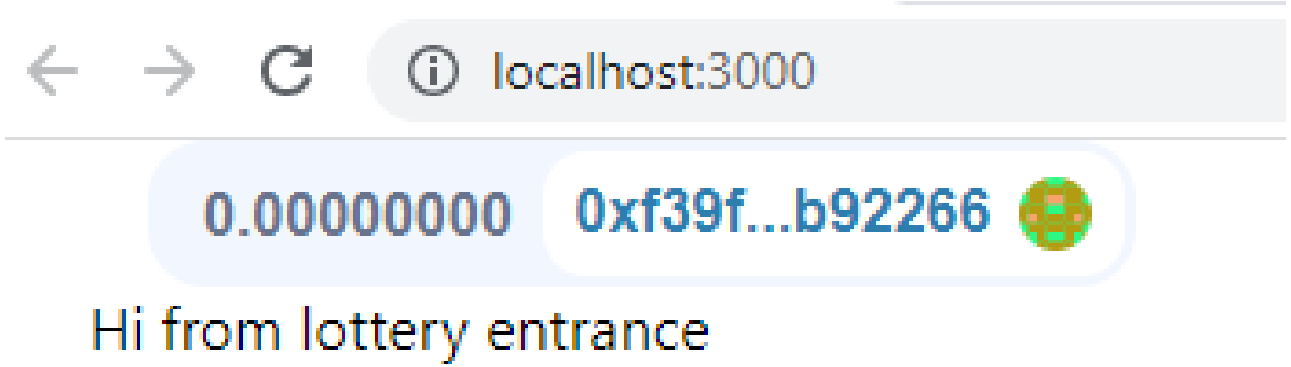
components/LotteryEntrance.js

```
export default function () {
    return <div>Hi from lottery entrance</div>
}
```

İndex.js

```
import LotteryEntrance from "../components/LotteryEntrance"
```

```
            <Header />
            <LotteryEntrance />
```



The function to ENTER THE LOTTERY - For this in moralis we use useWeb3Contract()
https://github.com/moralisWeb3/react-moralis#useweb3contract

components/LotteryEntrance.js

```
import { useWeb3Contract } from "react-moralis"

export default function () {
    const { runContractFunciton: enterRaffle } = useWeb3Contract({
        abi: //,
        contractAddress: //,
        functionName: //,
        params: {},
        msgValue: //
    })
    return <div>Hi from lottery entrance</div>
}
```

Automatic Constant Value UI Updater 17.26.30

For function parameter which are constant, make **constants** dir and add files

constants/abi.json

constants/contractAddresses.json

We will add a new script for front end in contract app.

deploy/99-update-front-end.js

This script will connnected to front-end. When we deploy contract, no matter what chain, we can update constants folder on our front end.

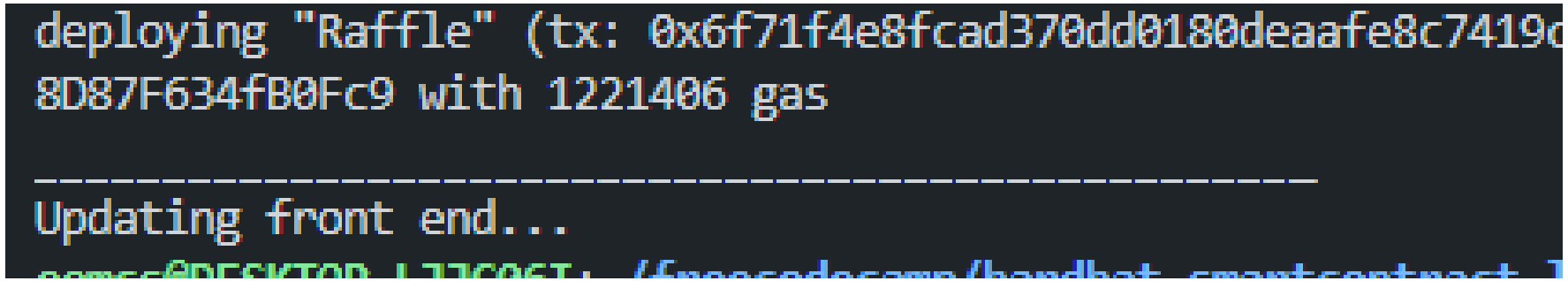
deploy/99-update-front-end.js

```
module.exports = async function () {
    if (process.env.UPDATE_FRONT_END) {
        console.log("Updating front end...")
    }
}
```

.env

```
UPDATE_FRONT_END=true
```

```
To test script deploy contract

Hh deploy
```



In font-end; json files must include "{}" curly brackets

To get contract address

```
async function updateContractAddresses() {
    const raffle = await ethers.getContract("Raffle")
    const chainId = network.config.chainId.toString()
    const currentAddress = JSON.parse(fs.readFileSync(FRONT_END_ADDRESSES_FILE, "utf-8"))
    if (chainId in currentAddress) {
        if (!currentAddress[chainId].includes(raffle.address)) {
            currentAddress[chainId].push(raffle.address)
        }
    } else {
        currentAddress[chainId] = [raffle.address]
    }
    fs.writeFileSync(FRONT_END_ADDRESSES_FILE, JSON.stringify(currentAddress))
}
```

To get abi

```
async function updateAbi() {
    const raffle = await ethers.getContract("Raffle")
    fs.writeFileSync(FRONT_END_ABI_FILE,
raffle.interface.format(ethers.utils.FormatTypes.json))
}
```

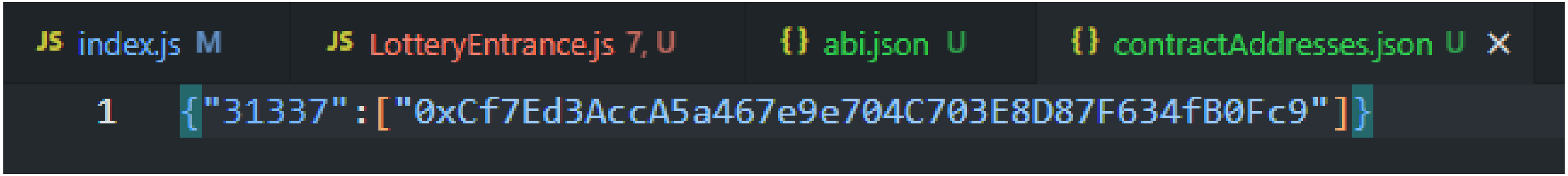
Main function

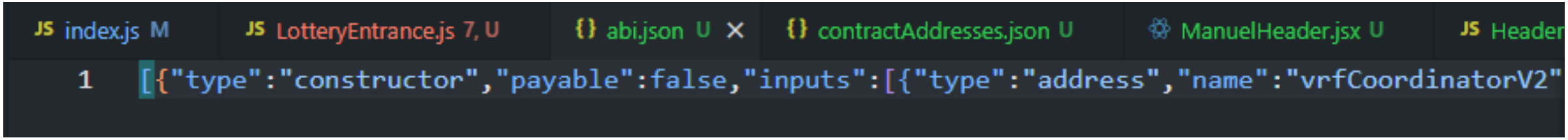
```
module.exports = async function () {
    if (process.env.UPDATE_FRONT_END) {
        console.log("Updating front end...")
        await updateContractAddresses()
        await updateAbi()
    }
}
```

Consts and imports

```
const { ethers, network } = require("hardhat")
const fs = require("fs")

const FRONT_END_ADDRESSES_FILE =
    "../nextjs-smartcontract-lottery-fcc/constants/contractAddresses.json"

const FRONT_END_ABI_FILE = "../nextjs-smartcontract-lottery-fcc/constants/abi.json"
```

Run ;
hh deploy OR hh node Contract will connect front-end ap and will update address and abi json files





WHEN TEST THE FROND END, LOCAL CHAIN MUST BE RUN BACKGROUND

To import address and abi with one line create the following file

constans/index.js

```
const contractAddresses = require("./contractAddresses.json")
const abi = require("./abi.json")

module.exports = {

    contractAddresses,
    abi,
}
```

İmport address and abi to LotteryEntrance.js

```
import { abi, contractAddresses } from "../constants"
```

```
export default function LotteryEntrance() {
    const { runContractFunciton: enterRaffle } = useWeb3Contract({
        abi: abi,
        contractAddress: contractAddresses[????chainid][0], // specify the network id
        functionName: "enterRaffle",
        params: {},
        msgValue: //
    })
    return <div>Hi from lottery entrance</div>
}
```

To get chain id

```
import { useMoralis } from "react-moralis"

export default function LotteryEntrance() {
    const { chainId } = useMoralis()
    console.log("Chain Id is :", chainId)

    // const { runContractFunciton: enterRaffle } = useWeb3Contract({
    //     abi: abi,
    //     contractAddress: contractAddresses[????chainid][0], // specify the network id
    //     functionName: "enterRaffle",
    //     params: {},
    //     msgValue: //
    // })
    return <div>Hi from lottery entrance</div>
}
```

Moralis knows the chainid. Because back in our header component, the header actuall passes all information about the metamask to the morales provider. And morales provider passes it down all the components inside these morales provided tags

When refresh the page from browser you could see chain id

```
Chain Id is : 0x539                                    LotteryEntrance.js?03c2:8
>
```

```
Chain Id is : 0x4                                      LotteryEntrance.js?03c2:8
Chain Id is : 0x4                                      LotteryEntrance.js?03c2:8
>
```

To convert hex to decimal

```
const { chainId: chainIdHex } = useMoralis()
    console.log("Chain Id is :", parseInt(chainIdHex))
```

```
Chain Id is : 4                                        LotteryEntrance.js?03c2:8
```

```
import { useMoralis } from "react-moralis"

export default function LotteryEntrance() {
    const { chainId: chainIdHex } = useMoralis()
    console.log("Chain Id is :", parseInt(chainIdHex))
    const chainId = parseInt(chainIdHex)
    const raffleAddres = chainId in contractAddresses ? contractAddresses[chainId][0] :
null

    const { runContractFunciton: enterRaffle } = useWeb3Contract({
        abi: abi,
        contractAddress: raffleAddres, // specify the network id
        functionName: "enterRaffle",
```

How to get msg.value

RUN CONTRACT-
FUNCTION CAN BOTH SEND TRANSACTIONS AND READ STATE

To get msg.value we will use getEntranceFee() view function from the contract. It was declerated in contract constructor.

```
const { chainId: chainIdHex, isWeb3Enabled } = useMoralis()
```

```
    useEffect(() => {
        if (isWeb3Enabled) {
            //try to read the raffle entrance fee
        }
    })

    return <div>Hi from lottery entrance</div>
}
```

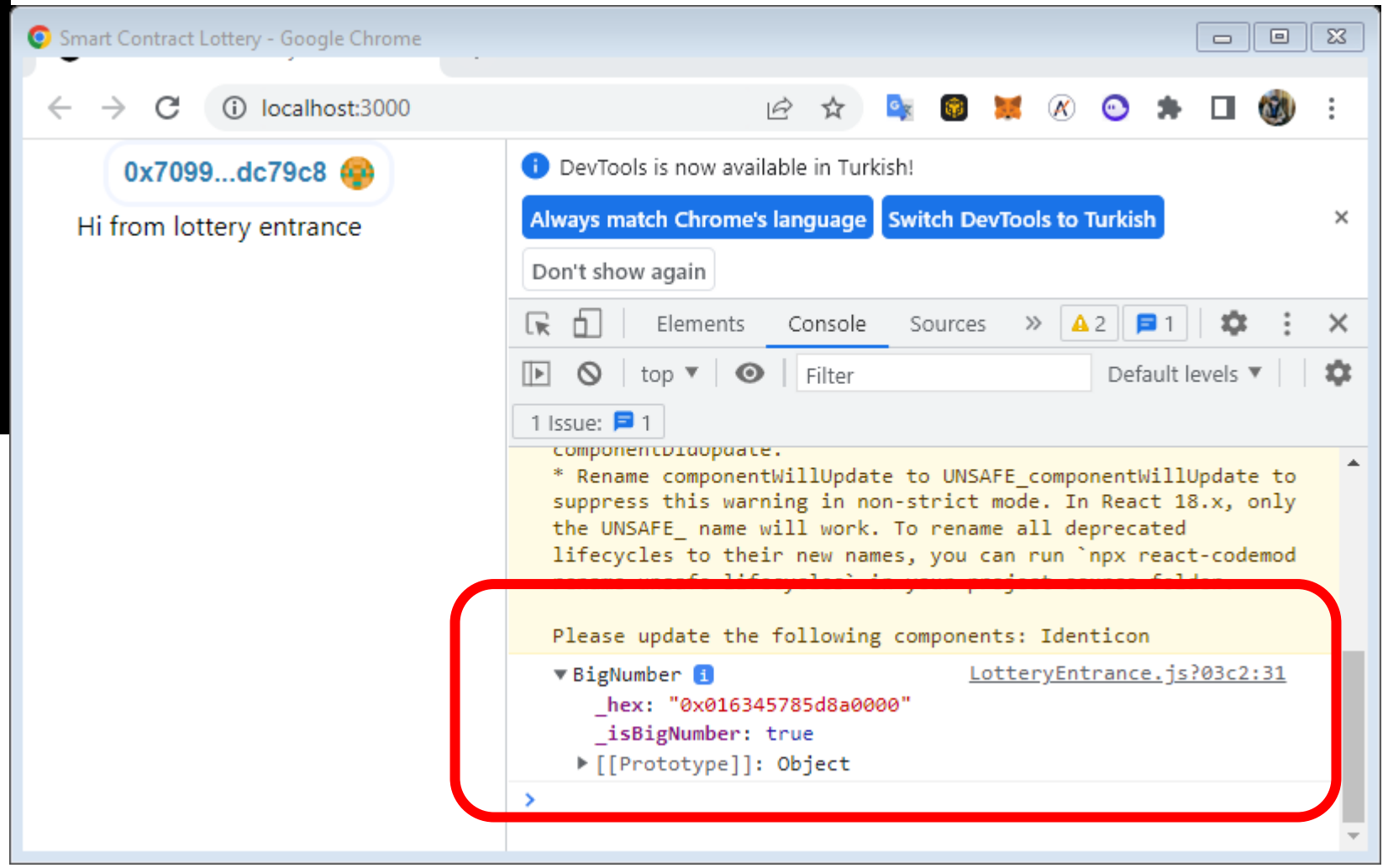
This code after enterraffle()

```
    const { runContractFunction: getEntranceFee } = useWeb3Contract({
        abi: abi,
        contractAddresses: raffleAddres,
        functionName: "getEntranceFee",
        params: {},
    })
```

Now we will call this code in useeffect()

```
    useEffect(() => {
        if (isWeb3Enabled) {
            async function updateUI() {
                const entranceFeeFromContract = await getEntranceFee()
                console.log(entranceFeeFromContract)
            }
            updateUI()
        }
    }, [])

    return <div>Hi from lottery entrance</div>
```

```
First time run of useeffect
we cant see the price fee.
When turns to true it will
be seem.

WARNING : TO SEE ENTRANC FEE
// hh node // connect
metamask to hardhat-
localhost network

THEN CHANGE AND TRY:
                    },
[isWeb3Enabled])
```



useState 17.45.50

SHOW ENTRANCE FEE ON OUR UI

```
const raffleAddres = chainId in contractAddresses ? contractAddresses[chainId][0] : null

    let entranceFee = ""
```

```
async function updateUI() {
            entranceFee = (await getEntranceFee()).toString()
            console.log(entranceFee)
        }
```

```
[Fast Refresh] done in 421ms          hot-dev-client.js?1600:139
10000000000000000000                  LotteryEntrance.js?03c2:32
>
```

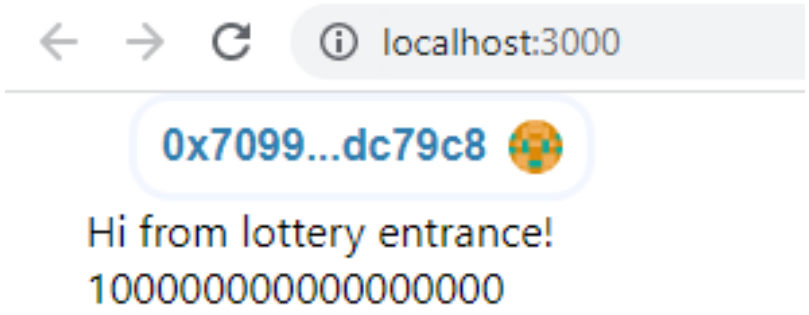
```
<div>
            Hi from lottery entrance!<div>{entranceFee}</div>
        </div>
```

We dont see entrance fee on UI. Because when etranceFee updates the browser not rendering. Because entrancefee is normal variable. To render when it update we will use usestate

https://reactjs.org/docs/hooks-state.html

```
const [entranceFee, setEntranceFee] = useState("0")  → FIRST VALUE IS VARIABLE. SECOND VALUE IS FUNCTION TO UPDATE THE VARIABLE

FIRST PARAMETER IS STATE OF VARIABLE.

0-ZERO IS THE STARTING VALUE OF VARIABLE
```

```
const entranceFeeFromCall = (await getEntranceFee()).toString()
                setEntranceFee(entranceFeeFromCall)
                console.log(entranceFee)
```




```
               const entranceFeeFromCall = (await getEntranceFee()).toString()
                setEntranceFee(ethers.utils.formatUnits(entranceFeeFromCall, "ether"))
                console.log(entranceFee)
```

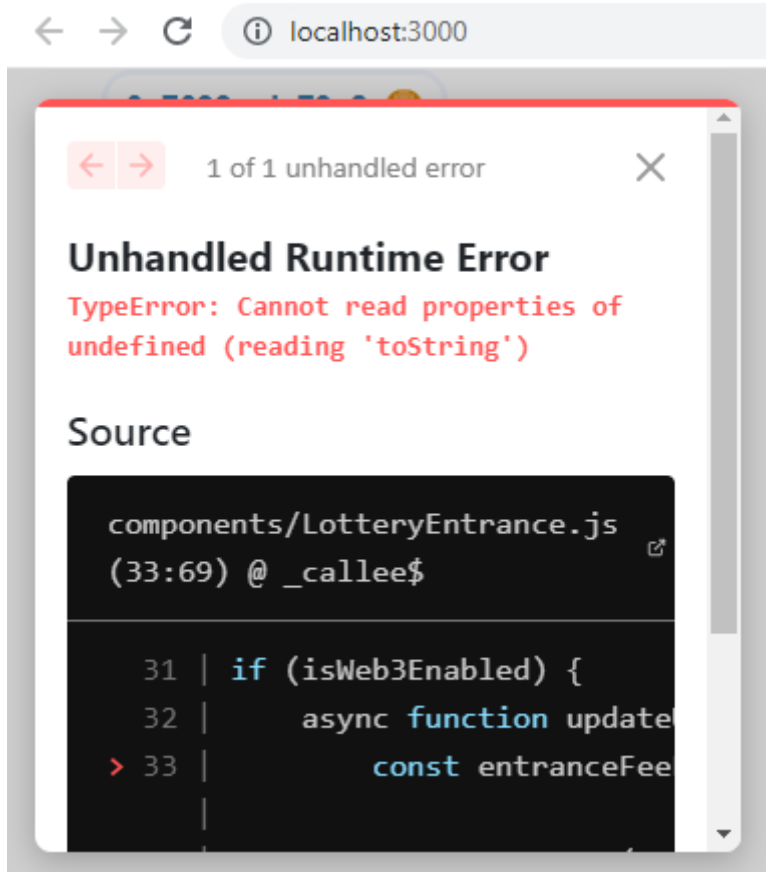
```
Entrance Fee : {entranceFee} ETH
```

Hi from lottery entrance!
Entrance Fee : 0.1

CALLING FUNCTIONS IN NEXTJS 17.49.51

When we change network to mainnet from metamask,it gets error. Because entrance fee is null on mainnet. No contract

To fix this eror.

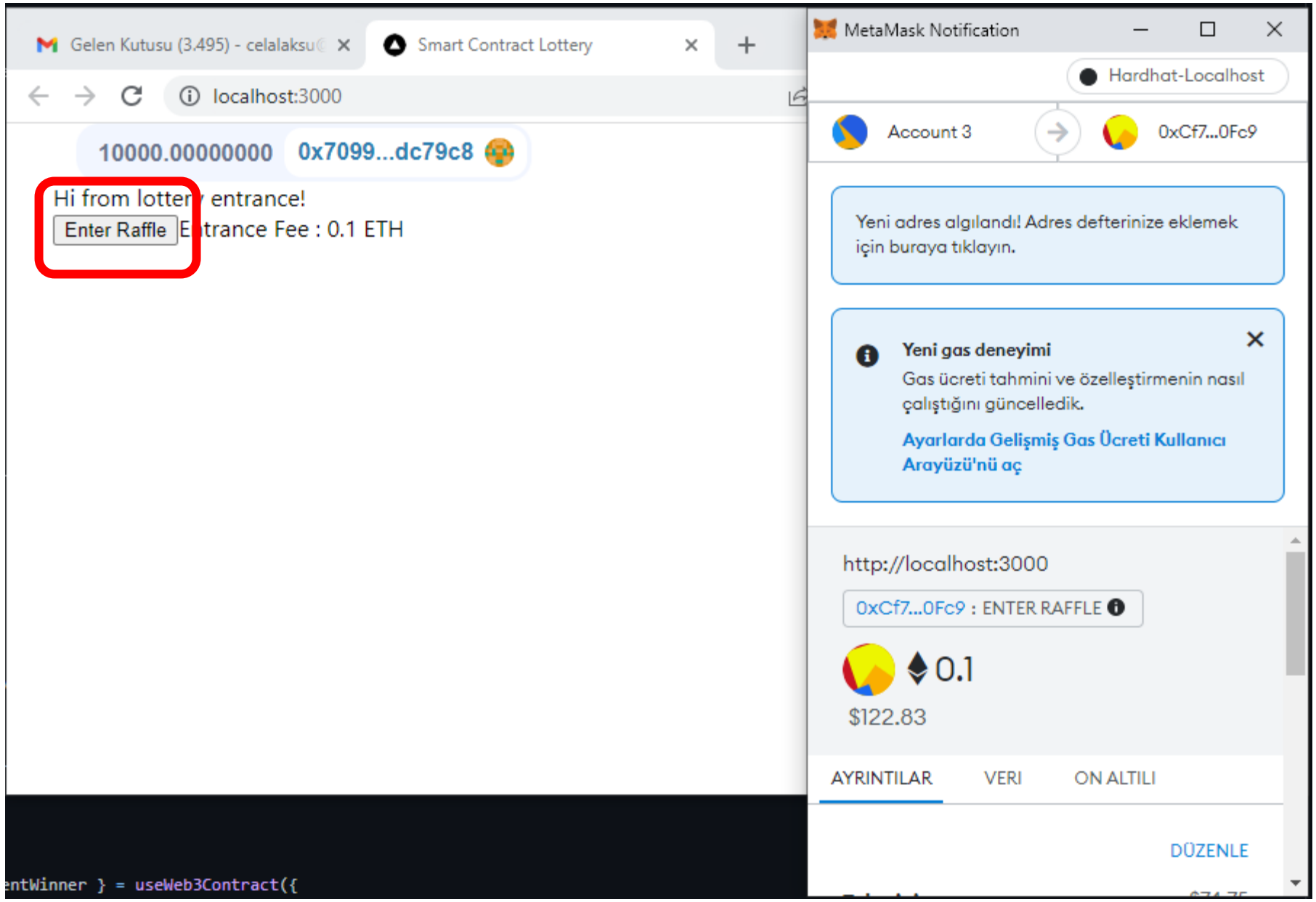


```
<div>
          Hi from lottery entrance!
          {raffleAddres ? (
              <div>
              Entrance Fee : {ethers.utils.formatUnits(entranceFee, "ether")} ETH
              </div>
          ) : (
              <div>No Raffle Address Detected</div>
          )}
      </div>
```

To run ENTERRAFFE FROM CONTRACT

```
const { runContractFunction: enterRaffle } = useWeb3Contract({
        abi: abi,
        contractAddress: raffleAddres,
        functionName: "enterRaffle",
        msgValue: entranceFee,
        params: {},
    })
```

```
{raffleAddres ? (
                <div>
                    <button
                        onClick={async () => {
                            await enterRaffle()
                        }}
                    >
                        Enter Raffle
                    </button>
                    Entrance Fee : {ethers.utils.
```


Gelen Kutusu (3.495) - celalaksu
Smart Contract Lottery
localhost:3000
10000.00000000
0x7099...dc79c8
Hi from lotter entrance!
Enter Raffle
Entrance Fee : 0.1 ETH
entWinner } = useWeb3Contract({
MetaMask Notification
Hardhat-Localhost
Account 3
0xCf7...0Fc9
Yeni adres algılandı! Adres defterinize eklemek için buraya tıklayın.
Yeni gas deneyimi
Gas ücreti tahmini ve özelleştirmenin nasıl çalıştığını güncelledik.
Ayarlarda Gelişmiş Gas Ücreti Kullanıcı Arayüzü'nü aç
http://localhost:3000
0xCf7...0Fc9 : ENTER RAFFLE
0.1
$122.83
AYRINTILAR
VERI
ON ALTILI
DÜZENLE

useNotification 17.52.59

https://web3ui.github.io/web3uikit/?path=/story/5-popup-notification--hook-demo

pages/_app.js

```
import { NotificationProvider } from "web3uikit"
```

```
            <NotificationProvider>
                <Component {...pageProps} />
            </NotificationProvider>
```

components/LotteryEntrance.js

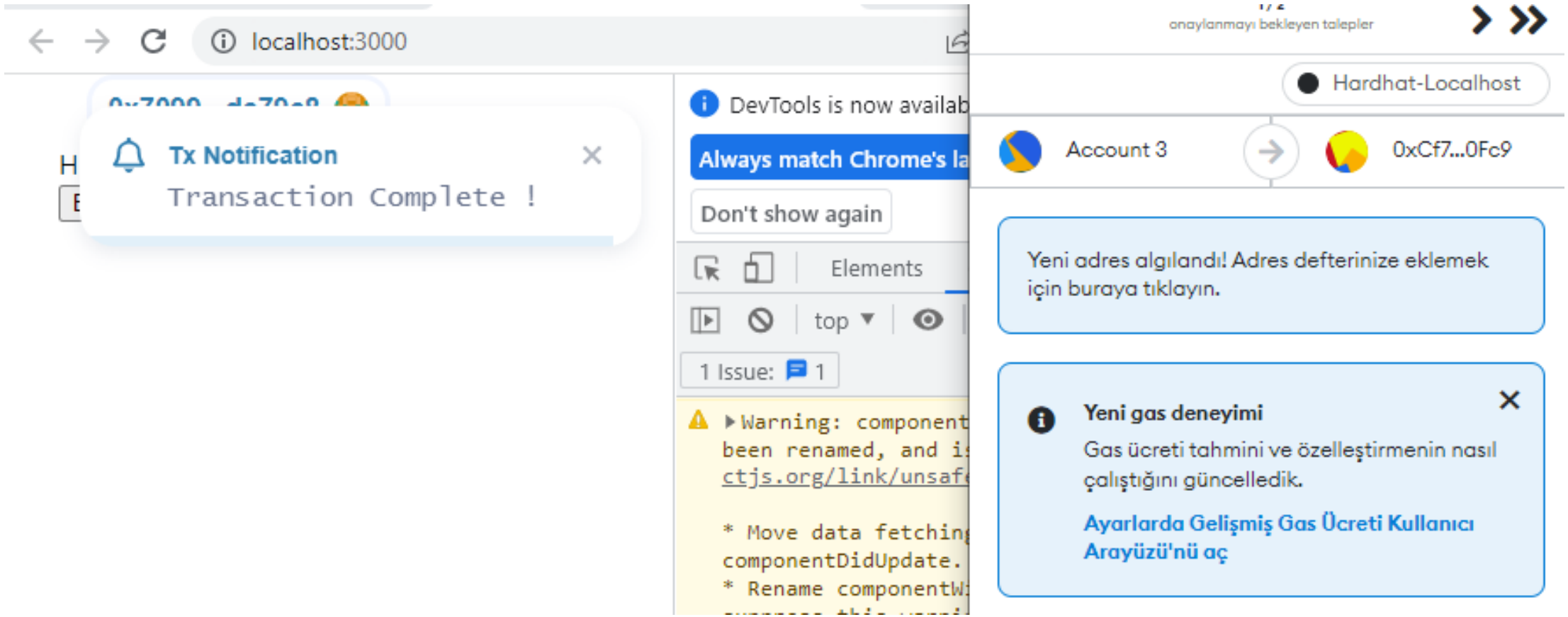
```
import { useNotification } from "web3uikit"
```

```
    const [entranceFee, setEntranceFee] = useState("0")
    const dispatch = useNotification() // like a little pop up
```

components/LotteryEntrance.js

```
    const hanleSuccess = async function (tx) {
        await tx.wait(1)
       handleNewNotification()
    }

    const handleNewNotification = function () {
        dispatch({
            type: "info",
            message: "Transaction Complete !",
            title: "Tx Notification",
            position: "topR",
            icon: "bell",
        })
    }
    return (
```

```
await enterRaffle({
                                onSuccess:
handleSuccess,
                            })
```


localhost:3000
Tx Notification
Transaction Complete !
DevTools is now availab
Always match Chrome's la
Don't show again
Elements
top
1 Issue: 1
Warning: component
been renamed, and is
ctjs.org/link/unsafe
* Move data fetching
componentDidUpdate.
* Rename componentW
onaylanmayı bekleyen talepler
Hardhat-Localhost
Account 3
0xCf7...0Fc9
Yeni adres algılandı! Adres defterinize eklemek için buraya tıklayın.
Yeni gas deneyimi
Gas ücreti tahmini ve özelleştirmenin nasıl çalıştığını güncelledik.
Ayarlarda Gelişmiş Gas Ücreti Kullanıcı Arayüzü'nü aç

Reading & Displaying Contract Data 17.58.05

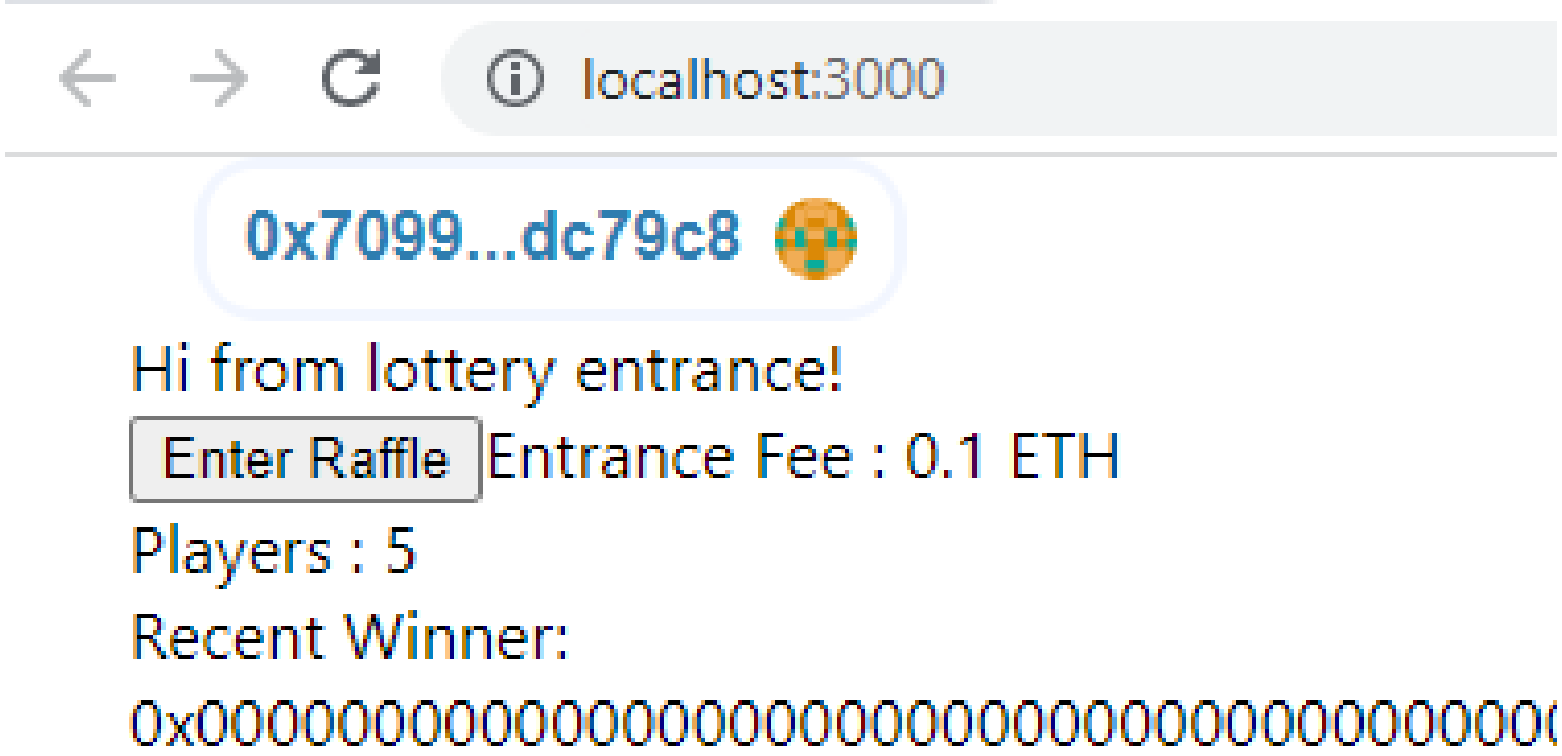
How many people in this game, who is the last winner

```
const [numPlayer, setNumPlayer] = useState("0")
const [recentWinner, setRecentWinner] = useState("0")
```

```
const { runContractFunction: getNumberOfPlayers } = useWeb3Contract({
    abi: abi,
    contractAddress: raffleAddres,
    functionName: "getNumberOfPlayers",
    params: {},
})
const { runContractFunction: getRecentWinner } = useWeb3Contract({
    abi: abi,
    contractAddress: raffleAddres,
    functionName: "getRecentWinner",
    params: {},
})
```

```
async function updateUI() {
                const entranceFeeFromCall = (await getEntranceFee()).toString()
                const numPlayersFromCall = (await getNumberOfPlayers()).toString()
                const recentWinnerFromCall = await getRecentWinner()
                setEntranceFee(entranceFeeFromCall)
                setNumPlayers(numPlayersFromCall)
                setRecentWinner(recentWinnerFromCall)
```

```
Entrance Fee : {ethers.utils.formatUnits(entranceFee, "ether")} ETH
                    <br></br>
                    Players : {numPlayers}
                    <br></br>
                    Recent Winner: {recentWinner}
```



```
async function updateUI() {
    const entranceFeeFromCall = (await getEntranceFee()).toString()
    const numPlayersFromCall = (await getNumberOfPlayers()).toString()
    const recentWinnerFromCall = await getRecentWinner()
    setEntranceFee(entranceFeeFromCall)
    setNumPlayers(numPlayersFromCall)
    setRecentWinner(recentWinnerFromCall)
    console.log(entranceFee)
}

useEffect(() => {
    if (isWeb3Enabled) {
        updateUI()
    }
}, [isWeb3Enabled])

const handleSuccess = async function (tx) {
    await tx.wait(1)
    handleNewNotification(tx)
    updateUI()
}
```

TO TEST GETTING A RECENT WINNER 18.01.46

IN CONCTRACT PROJEC CREATE scripts/mockOffchain.js COPY CODE FROM GITHUB

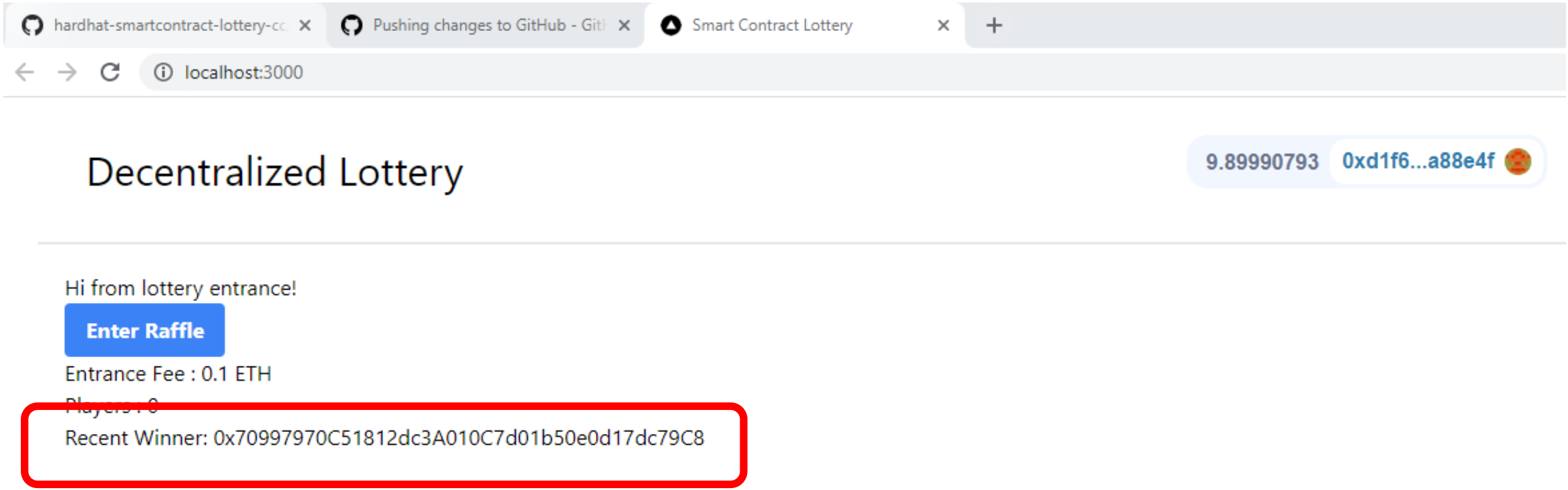
https://github.com/PatrickAlphaC/hardhat-smartcontract-lottery-fcc/blob/main/scripts/mockOffchain.js

```
const { ethers, network } = require("hardhat")

async function mockKeepers() {
    const raffle = await ethers.getContract("Raffle")
    const checkData = ethers.utils.keccak256(ethers.utils.toUtf8Bytes(""))
    const { upkeepNeeded } = await raffle.callStatic.checkUpkeep(checkData)
    if (upkeepNeeded) {
        const tx = await raffle.performUpkeep(checkData)
        const txReceipt = await tx.wait(1)
        const requestId = txReceipt.events[1].args.requestId
        console.log(`Performed upkeep with RequestId: ${requestId}`)
        if (network.config.chainId == 31337) {
            await mockVrf(requestId, raffle)
        }
    } else {
        console.log("No upkeep needed!")
    }
}

async function mockVrf(requestId, raffle) {
    console.log("We on a local network? Ok let's pretend...")
    const vrfCoordinatorV2Mock = await ethers.getContract("VRFCoordinatorV2Mock")
    await vrfCoordinatorV2Mock.fulfillRandomWords(requestId, raffle.address)
    console.log("Responded!")
    const recentWinner = await raffle.getRecentWinner()
    console.log(`The winner is: ${recentWinner}`)
}

mockKeepers()
    .then(() => process.exit(0))
    .catch((error) => {
        console.error(error)
        process.exit(1)
    })
```

```
RUN

yarn hardhat run scripts/mockOffchain.js --network localhost
```

```
Add following setting to hardhat.config.js
```

```
        localhost: {
            chainId: 31337,
        },
```


hardhat-smartcontract-lottery-cc
Pushing changes to GitHub - Gitl
Smart Contract Lottery
localhost:3000
Decentralized Lottery
9.89990793
0xd1f6...a88e4f
Hi from lottery entrance!
Enter Raffle
Entrance Fee : 0.1 ETH
Players : 0
Recent Winner: 0x70997970C51812dc3A010C7d01b50e0d17dc79C8

A note about onSuccess 18.02.55

```
 await enterRaffle({
                                onSuccess: handleSuccess,
                            })
```

Onsuccess isnt checking that the transaction has a block confirmatin, its just checking to see that the transaction was successfully sent to metamask.
So thats why up in that other function we do TX.wait(1) because thats the piece that actuall waits for the transaction to be confirmed.

A CHALLENGE TO YOU 18.03.24

https://tailwindcss.com/

https://tailwindcss.com/docs/guides/nextjs

Tailwind & Styling 18.04.12

INSTALL

```
yarn add --dev tailwindcss postcss autoprefixer
```

TO INIT - CONFIG FILI

```
yarn tailwindcss init -p
```

```
Created Tailwind CSS config file: tailwind.config.js
Created PostCSS config file: postcss.config.js
```

tailwind.config.js

```
/** @type {import('tailwindcss').Config} */
module.exports = {
    content: ["./pages/**/*.{js,ts,jsx,tsx}", "./components/**/*.{js,ts,jsx,tsx}"],
    theme: {
        extend: {},
    },
    plugins: [],
}
```

styles/globals.css // Delete all other lines

```
@tailwind base;
@tailwind components;
@tailwind utilities;
```

Install PostCSS Language Support Extension on VS Code

```
Searc for border classname css setting in tailwind docs and add;

    return (
        <div className="border-b-2">

RE RUN APP
```

```
INSTALL TAILWIND EXTENSION ON VS CODE . Tailwind CSS IntelliSense - Tailwind labs
```

components/Header.js TAILWIND SETTINGS

```
export default function Header() {
    return (
        <div className="p-5 border-b-2 flex
flex-row">
            <h1 className="py-4 px-4 font-blog
text-3xl">Decentralized Lottery</h1>
            <div className="ml-auto py-2 px-4">
                <ConnectButton
moralisAuth={false}></ConnectButton>
            </div>
        </div>
    )
```

components/LotteryEntrance.js

```
<div>
                    <button
                        className="bg-blue-500 hover:bg-blue-700 text-white font-bold
py-2 px-4 rounded ml-auto"
                        onClick={async () => {
```

```
const { runContractFunction: enterRaffle, isLoading, isFetching } = useWeb3Contract({
```

```
 onClick={async () => {
                            await enterRaffle({
                                onSuccess: handleSuccess,
                            })
                        }}
                        disabled={isLoading || isFetching}
```

Add spin animation to Enter Raffle button

```
    disabled={isLoading || isFetching}
                    >
                        {isLoading || isFetching ? (
                            <div className="animate-spin spinner-border h-8 w-8 border-b-2
rounded-full"></div>
                        ) : (
                            <div>Enter Raffle</div>
                        )}
                    </button>
```

DEPLOY FRONTEND TO IPFS 18.13

Deploying front end in a more decentralized way

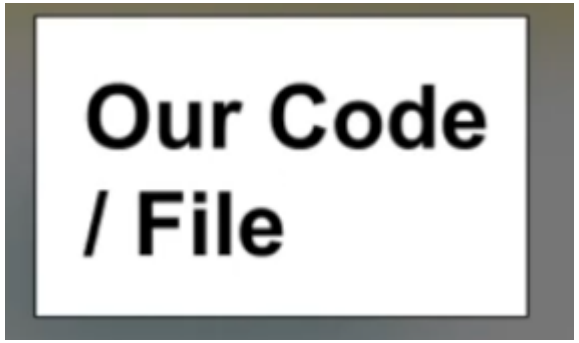
IPFS

https://ipfs.io/

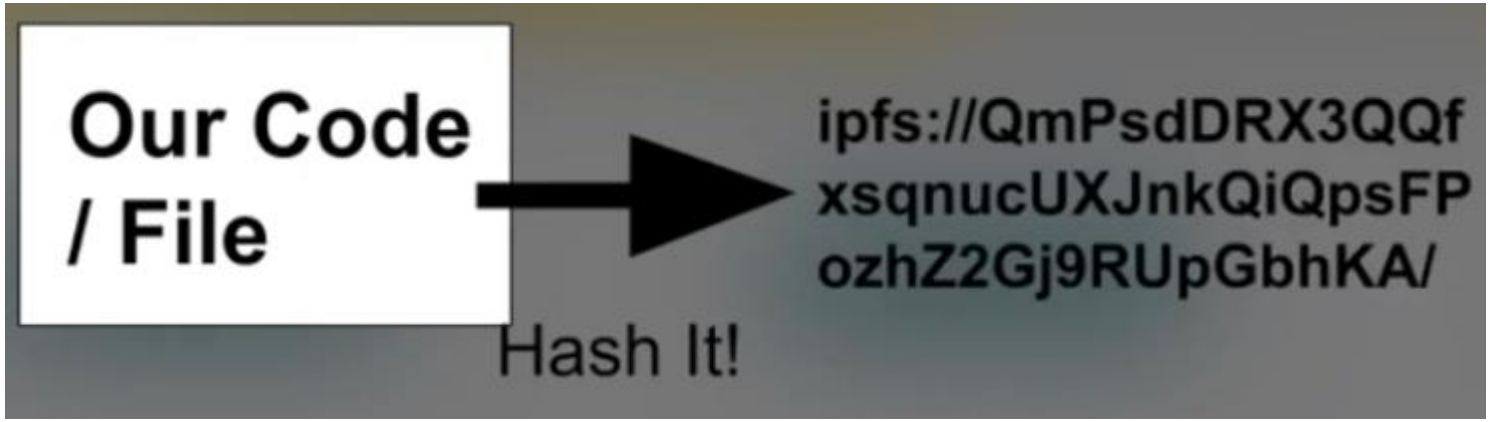
https://vercel.com/

HOW IT WORKS

https://ipfs.io/#how



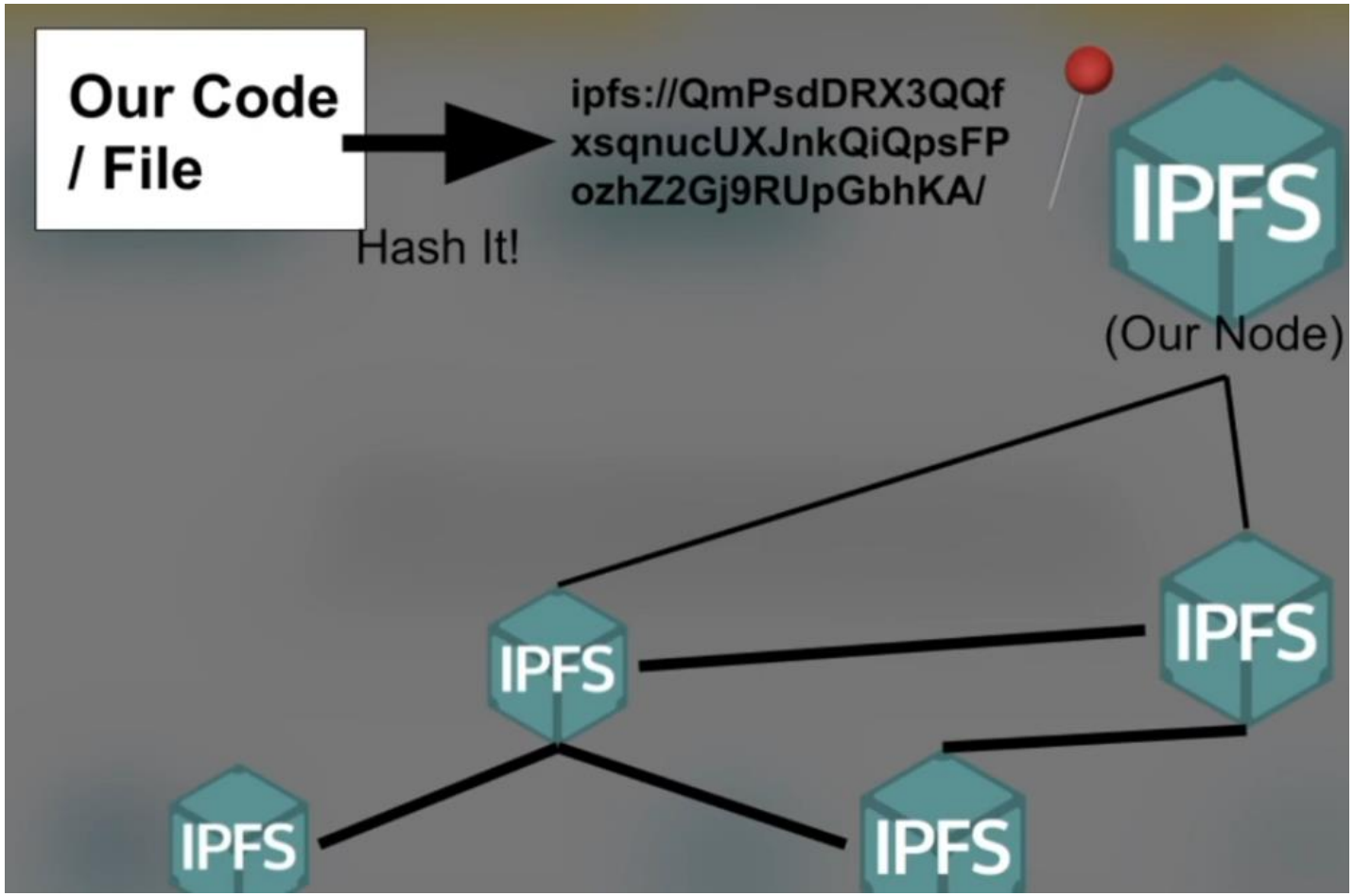
Hash data/file to a unique code. This is the first thing what ipfs does.The hash is only points that data.Its a massive code data on ton of text. You can encode all of that into a single hash function.



Your IPFS node does this hahing for you. And every sing IPFS node on the planet has the exact same hashing function, kind of like a blockchain.

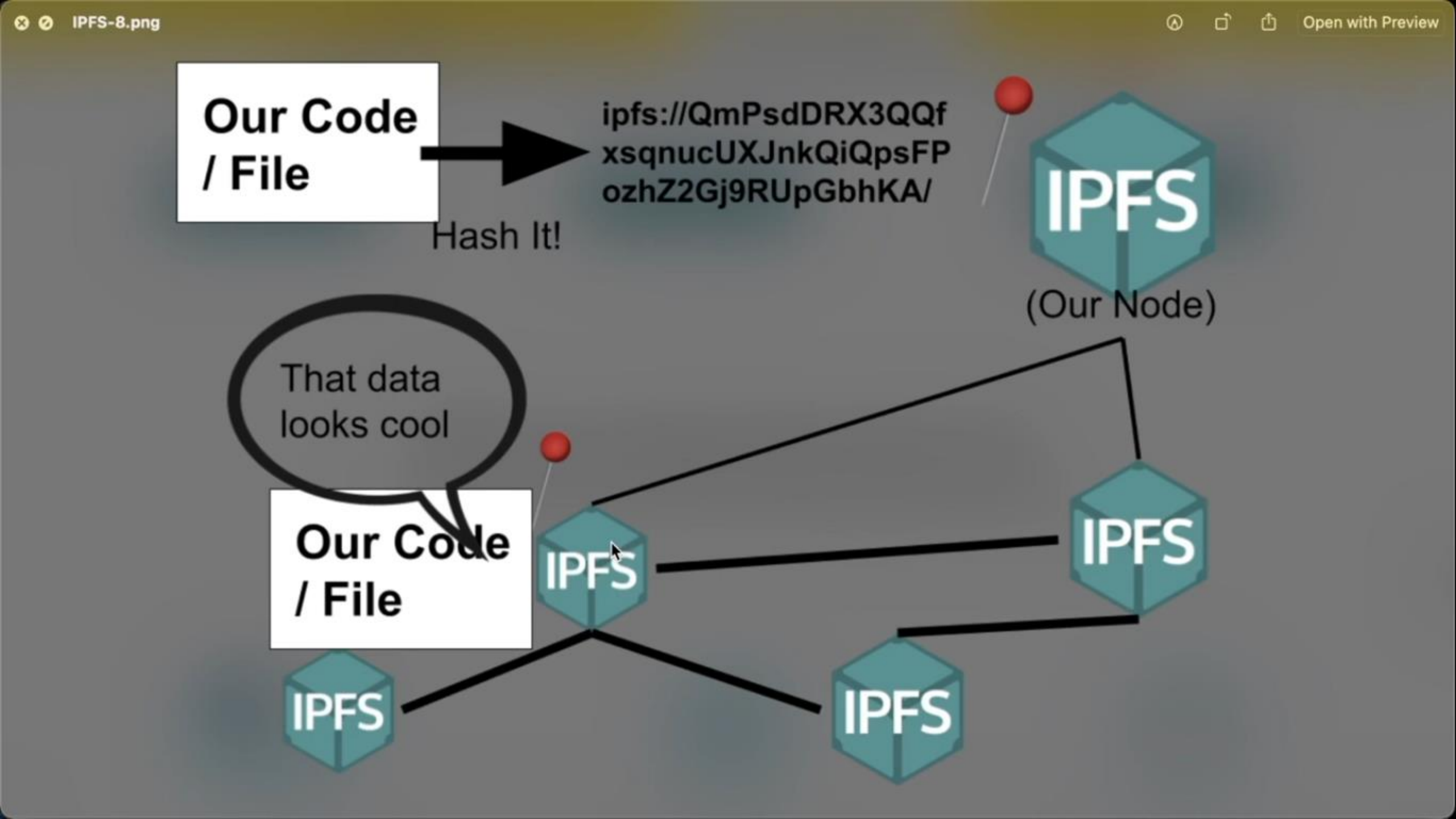
Pin that code/file/data to our node.

Our node is connected to a network of other IPFS nodes. So there is a massive network of people running IPFS nodes, they incredibly lightweight, way ligther weight than any other blockchain node.And they all talk to each other. So i ask to network "i want to get this hash" all these nodes talks to each other and eventually our node saying " i found a node that has that hash". Here is the file associated with it. Than other nodes pins data, copy data to their node.

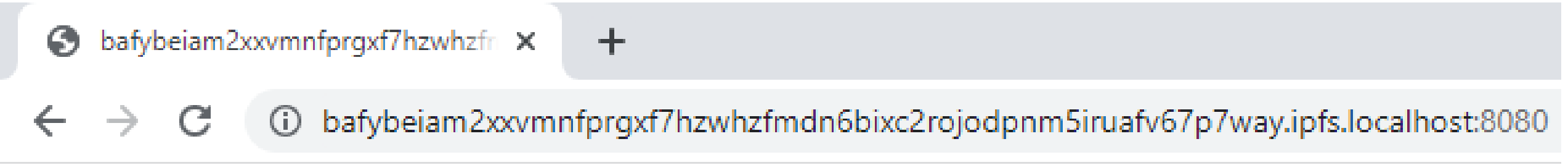


IPFS network drastically different than a blockchain. Its cant do smart contract, there is no execution.it can really only storega, just decentralized storage that IPFS can do.

IPFS optionally choose which data they want to pin.



Hosting on IPFS 18.18.51




```
/** @type {import('next').NextConfig} */
const nextConfig = {
  reactStrictMode: true,
}

module.exports = nextConfig
```

Or using ipfs gateway
https://ipfs.io/ipfs/<paste cid code>

With brave Works better

Install IPFS DESKTOP.
Open app
Goto FILES section
Click IMPORT

Select a file (next.config.js- selected)

Double click file on IPFS

From MORE button COPY the CID

Wiew in our browser:
add ipfs companinon

İn browser address line
ipfs://<paste cid code>

Hoşgeldiniz | IPFS Refakatçisi - Google Chrome

IPFS Refakatçisi | chrome-extension://nibjojkomfdiaoajekhjakgkdhaomnch/dist/landing-pages/welcome/index.html

IPFS Refakatçisi

Düğümünüz 373 eşe bağlı.

Durum | Dosyalar | Eşler

IPFS HAKKINDA BILGI EDININ

- IPFS Refakatçisinin özellikleri hakkında bilgi edinin
- IPFS Nasıl Çalışır kılavuzundaki temel kavramları öğrenin
- Daha derine inmek için IPFS dokümantasyon sitesini ziyaret edin

IPFS ÜZERINE INŞA EDIN

- Başlamanıza yardımcı olacak nasıl yapılırlar ve öğreticiler bulun
- Başkalarının IPFS ile ne gibi harika şeyler geliştirdiğini görün

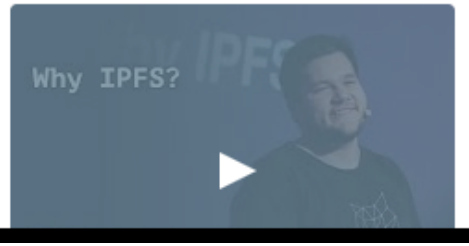
YARDIM AL

- Resmi forumlarda sorular sorun ve IPFS'yi tartışın

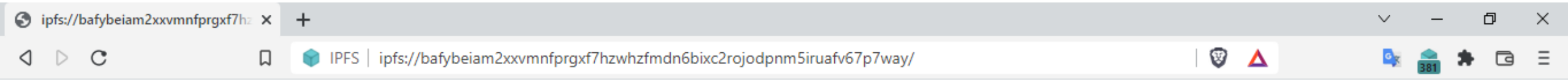
TOPLULUĞA KATIL

- Kod, belgeler ve daha fazlasıyla katkıda bulunun
- IPFS'yi favori dilinize çevirin
- Tüm IPFS topluluk kaynaklarını keşfedin

NEDEN IPFS?



IPFS NASIL ÇALIŞIR?




```
/** @type {import('next').NextConfig} */
const nextConfig = {
  reactStrictMode: true,
}

module.exports = nextConfig
```

HOW TO DEPLOY WITH LONG WAY:

Remember:
IPFS dont execute anything. Just stores files
so we have to create a static web site

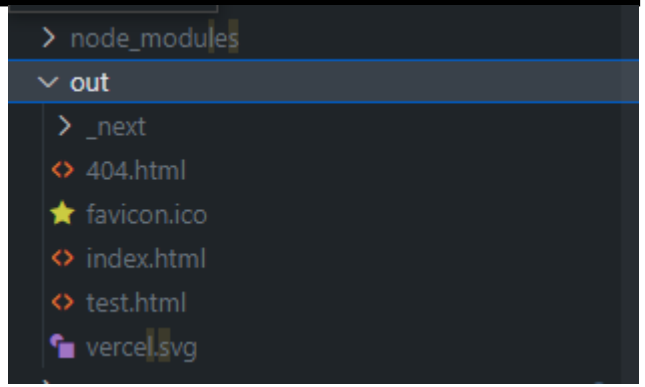
First: We will create static web site

In Project :
To create static seb site run following command

yarn build // this builds site with out any server code - this is production build

yarn next export // gives error - error screenshot and solution in next slayt

YARN NEXT EXPORT COMMAND makes an folder "OUT".



```
eemcs@DESKTOP-LJJC06I:~/freecodecamp/nextjs-smartcontract-lottery-fcc$ yarn next export
yarn run v1.22.18
$ /home/eemcs/freecodecamp/nextjs-smartcontract-lottery-fcc/node_modules/.bin/next export
info  - using build directory: /home/eemcs/freecodecamp/nextjs-smartcontract-lottery-fcc/.next
info  - Copying "static build" directory
info  - No "exportPathMap" found in "/home/eemcs/freecodecamp/nextjs-smartcontract-lottery-fcc/next.config.js". Generating map from "./p
ages"
Error: Image Optimization using Next.js' default loader is not compatible with `next export`.
  Possible solutions:
    - Use `next start` to run a server, which includes the Image Optimization API.
    - Use any provider which supports Image Optimization (like Vercel).
    - Configure a third-party loader in `next.config.js`.
    - Use the `loader` prop for `next/image`.
  Read more: https://nextjs.org/docs/messages/export-image-api
    at /home/eemcs/freecodecamp/nextjs-smartcontract-lottery-fcc/node_modules/next/dist/export/index.js:157:23
    at async Span.traceAsyncFn (/home/eemcs/freecodecamp/nextjs-smartcontract-lottery-fcc/node_modules/next/dist/trace/trace.js:79:20)
error Command failed with exit code 1.
info Visit https://yarnpkg.com/en/docs/cli/run for documentation about this command.
```

next.config.js SOLUTION

```
module.exports = {
    nextConfig,
    images: {
        loader: "akamai",
        path: "",
    },
}
```

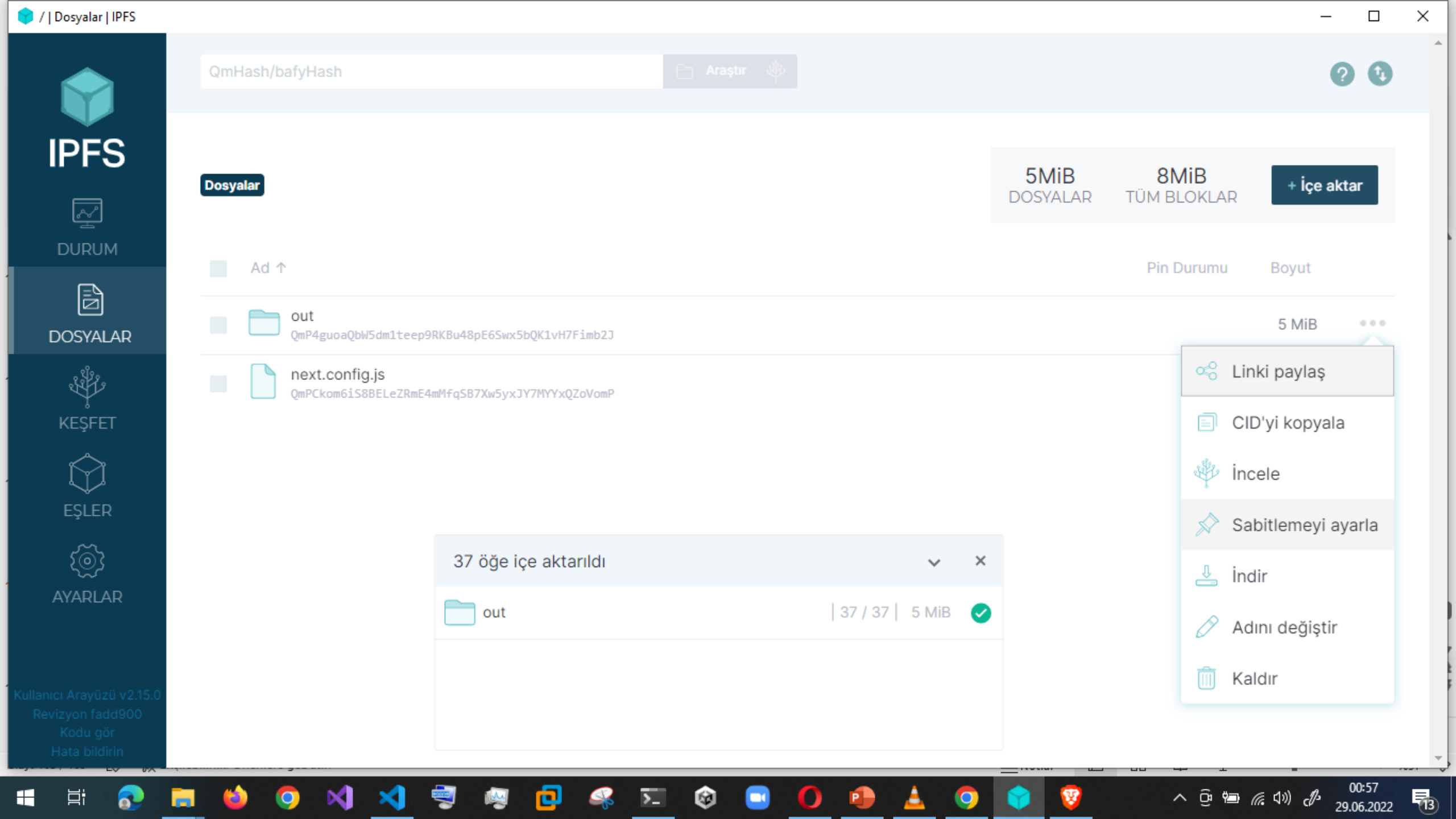
```
eemcs@DESKTOP-LJJC06I:~/freecodecamp/nextjs-smartcontract-lottery-fcc$ yarn next export
yarn run v1.22.18
$ /home/eemcs/freecodecamp/nextjs-smartcontract-lottery-fcc/node_modules/.bin/next export
info  - using build directory: /home/eemcs/freecodecamp/nextjs-smartcontract-lottery-fcc/.next
info  - Copying "static build" directory
info  - No "exportPathMap" found in "/home/eemcs/freecodecamp/nextjs-smartcontract-lottery-fcc/next.config.js". Generating map from "./p
ages"
info  - Launching 7 workers
warn  - Statically exporting a Next.js application via `next export` disables API routes.
This command is meant for static-only hosts, and is not necessary to make your application static.
Pages in your application without server-side data dependencies will be automatically statically exported by `next build`, including pag
es powered by `getStaticProps`.
Learn more: https://nextjs.org/docs/messages/api-routes-static-export
info  - Copying "public" directory
info  - Exporting (3/3)
Export successful. Files written to /home/eemcs/freecodecamp/nextjs-smartcontract-lottery-fcc/out
Done in 2.43s.
```

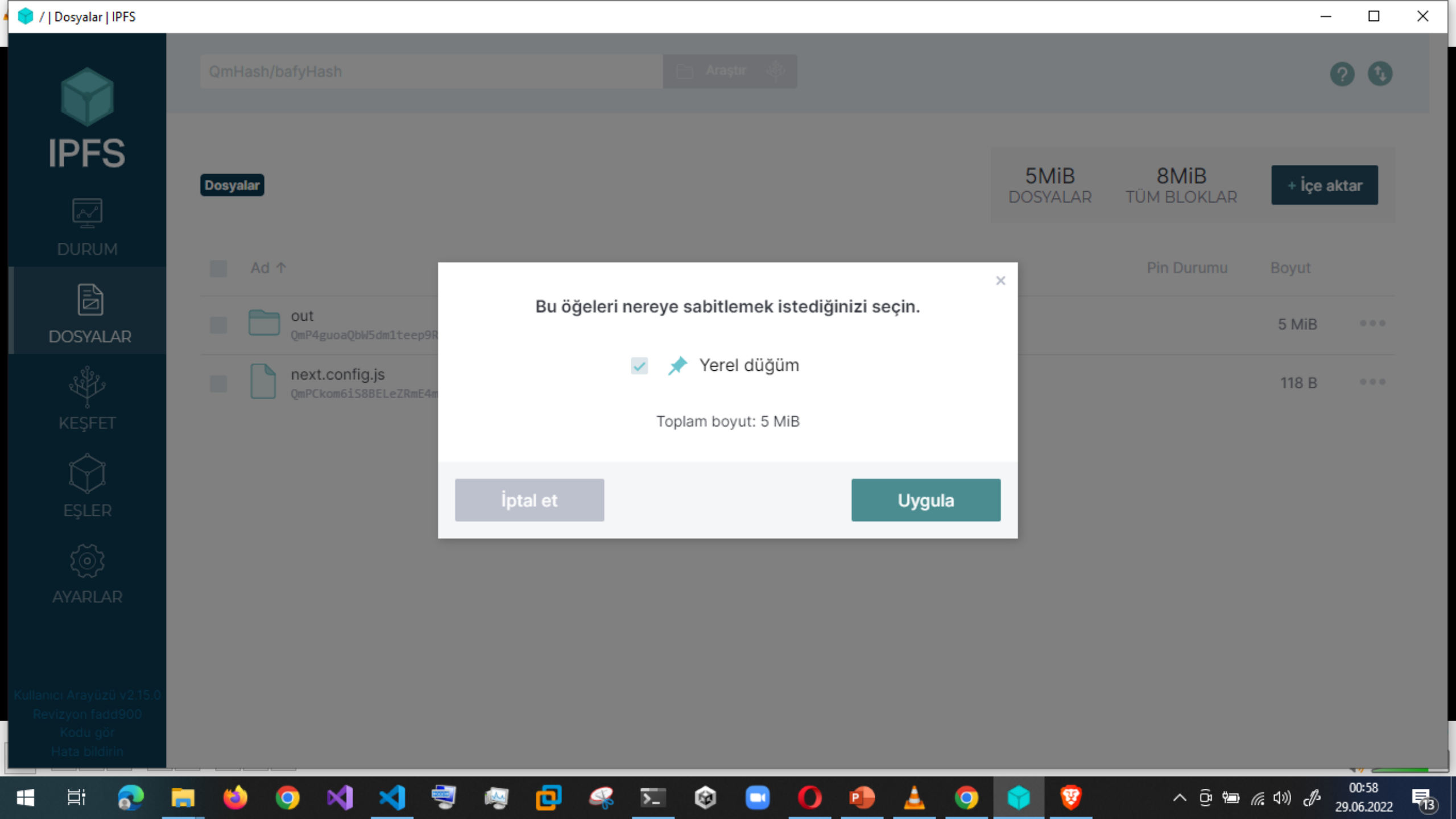
Yarn build OUTPUT:

```
eemcs@DESKTOP-LJJC06I:~/freecodecamp/nextjs-smartcontract-lottery-fcc$ yarn build
yarn run v1.22.18
$ next build
info  - Checking validity of types
warn  - No ESLint configuration detected. Run next lint to begin setup
info  - Creating an optimized production build
warn  - Compiled with warnings

./node_modules/moralis/lib/browser/Web3Connector/MagicWeb3Connector.js
Module not found: Can't resolve 'magic-sdk' in '/home/eemcs/freecodecamp/nextjs-smartcontract-lottery-fcc/node_modules/moralis/lib/browser/Web3Connector'

Import trace for requested module:
./node_modules/moralis/lib/browser/MoralisWeb3.js
./node_modules/moralis/lib/browser/Parse.js
./node_modules/moralis/index.js
./node_modules/react-moralis/lib/index.esm.js
./pages/_app.js

info  - Collecting page data
info  - Generating static pages (4/4)
info  - Finalizing page optimization
```

```
Page                                      Size     First Load JS
┌ ○ / (1024 ms)                           2.35 kB        1.08 MB
├   /_app                                 0 B            1.07 MB
├ ○ /404                                  196 B          1.07 MB
├ λ /api/hello                            0 B            1.07 MB
└ ○ /test (1002 ms)                       466 B          1.07 MB
+ First Load JS shared by all             1.07 MB
  ├ chunks/framework-45405dbdcddf505d.js   45.3 kB
  ├ chunks/main-30435f12318d8710.js        28.7 kB
  ├ chunks/pages/_app-387b03afe2b5f2bc.js  993 kB
  ├ chunks/webpack-9fc06fa7df06a8d1.js     1.91 kB
  └ css/86a2d7b7c4530015.css               1.76 kB

λ  (Server)  server-side renders at runtime (uses getInitialProps or getServerSideProps)
○  (Static)  automatically rendered as static HTML (uses no initial props)

Done in 74.78s.
```

```
GOTO THE IPFS:

IMPORT- FOLDER - COPY "OUT" FOLDER TO IPFS

PIN THE FOLDER ( NEXT SLAYT SHOWS )

Copy CID

Text ipfs addres to browser
       ipfs://<cid>
```


/ | Dosyalar | IPFS
IPFS
DURUM
DOSYALAR
KEŞFET
EŞLER
AYARLAR
Kullanıcı Arayüzü v2.15.0
Revizyon fadd900
Kodu gör
Hata bildirin
QmHash/bafyHash
Araştır
Dosyalar
5MiB
DOSYALAR
8MiB
TÜM BLOKLAR
+ İçe aktar
Ad
Pin Durumu
Boyut
out
QmP4guoaQbW5dm1teep9RKBu48pE6Swx5bQK1vH7Fimb2J
5 MiB
next.config.js
QmPCkom6iS8BELeZRmE4mMfqSB7Xw5yxJY7MYYxQZoVomP
Linki paylaş
CID'yi kopyala
İncele
Sabitlemeyi ayarla
İndir
Adını değiştir
Kaldır
37 öğe içe aktarıldı
out
37 / 37
5 MiB
00:57
29.06.2022


/ | Dosyalar | IPFS
IPFS
DURUM
DOSYALAR
KEŞFET
EŞLER
AYARLAR
QmHash/bafyHash
Araştır
Dosyalar
5MiB
DOSYALAR
8MiB
TÜM BLOKLAR
+ İçe aktar
Ad
Pin Durumu
Boyut
out
5 MiB
next.config.js
118 B
Bu öğeleri nereye sabitlemek istediğinizi seçin.
Yerel düğüm
Toplam boyut: 5 MiB
İptal et
Uygula
Kullanıcı Arayüzü v2.15.0
Revizyon fadd900
Kodu gör
Hata bildirin
00:58
29.06.2022

Decentralized Lottery

Connect Wallet

Hi from lottery entrance!
No Raffle Address Deteched

Hosting on IPFS & Filecoin using Fleek 18.25.45

EASY WAY TO DEPLOY FRONT END

https://fleek.co/

Before open this site, turn off ipfs companion from browser extension. Because it opens on ipfs

Smart Contract Lottery | Fleek: Build on the New Internet

bafybeiarrje6j344nfquzi6fxc4eqs3lmjdohgqq4tekukze4shs3rswhe.ipfs.localhost:8080

⚡ Release Update! ⚡ Ethereum Wallet Authentication Now Live On Fleek - Learn More

fleek

PRODUCTS BLOG COMMUNITY DOCS PRICING SIGN IN SIGN UP

Smart Contract Lottery | Fleek: Build on the New Internet

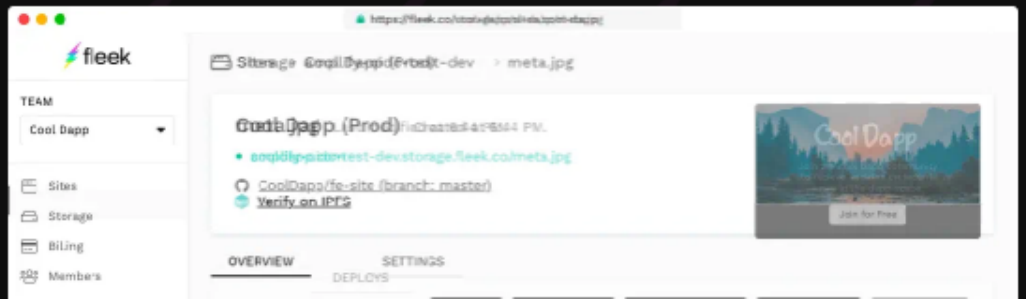
fleek.co

⚡ Release Update! ⚡ Ethereum Wallet Authentication Now Live On Fleek - Learn More

fleek

PRODUCTS BLOG COMMUNITY DOCS PRICING SIGN IN SIGN UP

Build on the New Internet

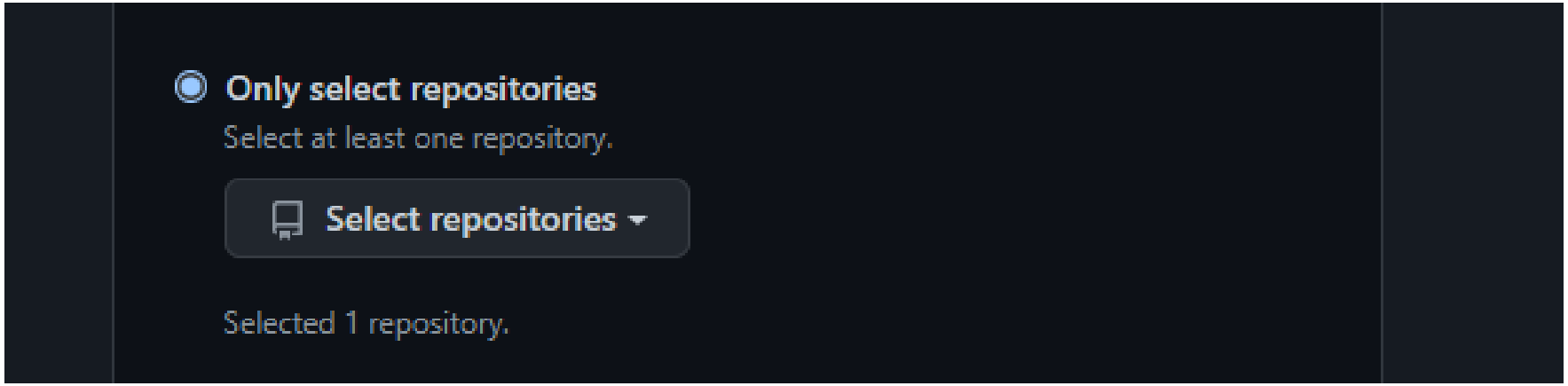


Sign up with Github. Using github helps us to automaticall deploy.

On fleek.co "Crate a new web Site"

Then push the front end to github.

On fleek.co "Connect with github"



Install and Authorize

Fleek: New Site - Google Chrome

app.fleek.co/#/start/choose-repository?accountId=900e8882-0b4d-47ed-b9f4-606e3d4d8de5

🔥 Release Update! 🔥 Ethereum Wallet Authentication Now Live On Fleek - Learn More

fleek

Team

Celal AKSU Team

- Hosting
- Storage
- Billing
- Members
- Team Settings

Celal AKSU

Hosting

← Back

Choose github repo,
Select hosting services as IPFS

Create a new site

From zero to hero, three easy steps to get your site live.

1. Connect to Github ✓ 2. Pick a repository 3. Deploy location 4. Build options, and deploy!

Continuous Deployment: GitHub App

Choose the repository you want to link to your site on Fleek. When you push to Git we run your build tool on our services and deploy the result.

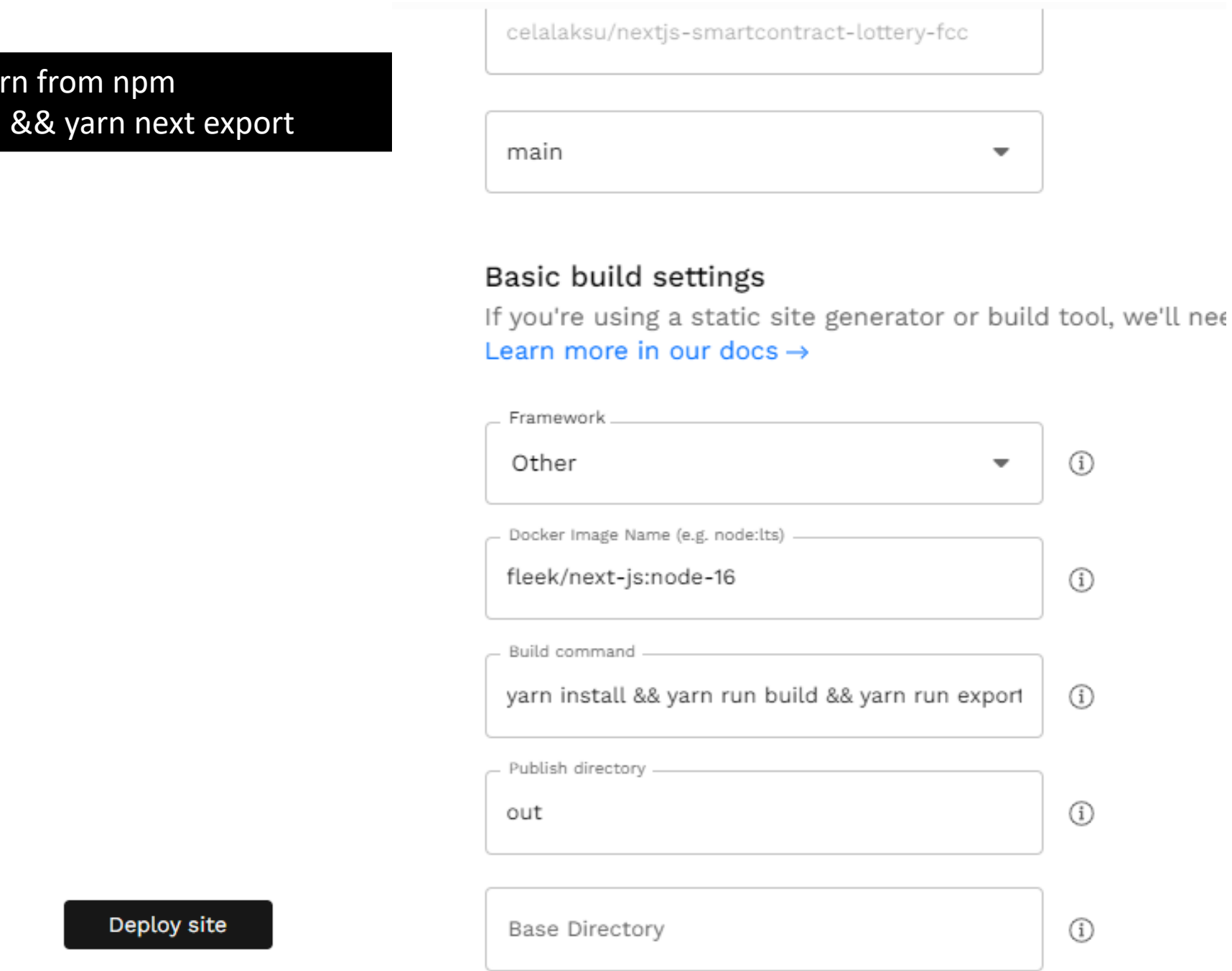
celalaksu

Search repos

celalaksu/nextjs-smartcontract-lottery-fcc

Can't see your repo here? Configure the Fleek app on GitHub. →

Change build command to yarn from npm
yarn install && yarn run build && yarn next export



Fleek: Hosting - Google Chrome

app.fleek.co/#/sites/tiny-mountain-3650/deploys/2022-06-29T07:59:59.063Z?accountId=900e8882-0b4d-47ed-b9f4-606e3d4d8de5

Release Update! Ethereum Wallet Authentication Now Live On Fleek - Learn More

fleek

Team

Celal AKSU Team

- Hosting
- Storage
- Billing
- Members
- Team Settings

Celal AKSU

Hosting > tiny-mountain-3650

If fails, change settigs
And click to trigger deploy

tiny-mountain-3650 Created at 10:59 AM

• Site deploy in progress

celalaksu/nextjs-smartcontract-lottery-fcc (branch: main)

IPFS Hash Pending

OVERVIEW | DEPLOYS | SETTINGS

← Back to All Deploys

Deploy Status

Deploy in progress

Deployment: main@b0fc074 Today at 11:00 AM

Fleek's robots are busy building and deploying your site to IPFS and our CDN

Deploy settings | Cancel deploy

Deploy Log

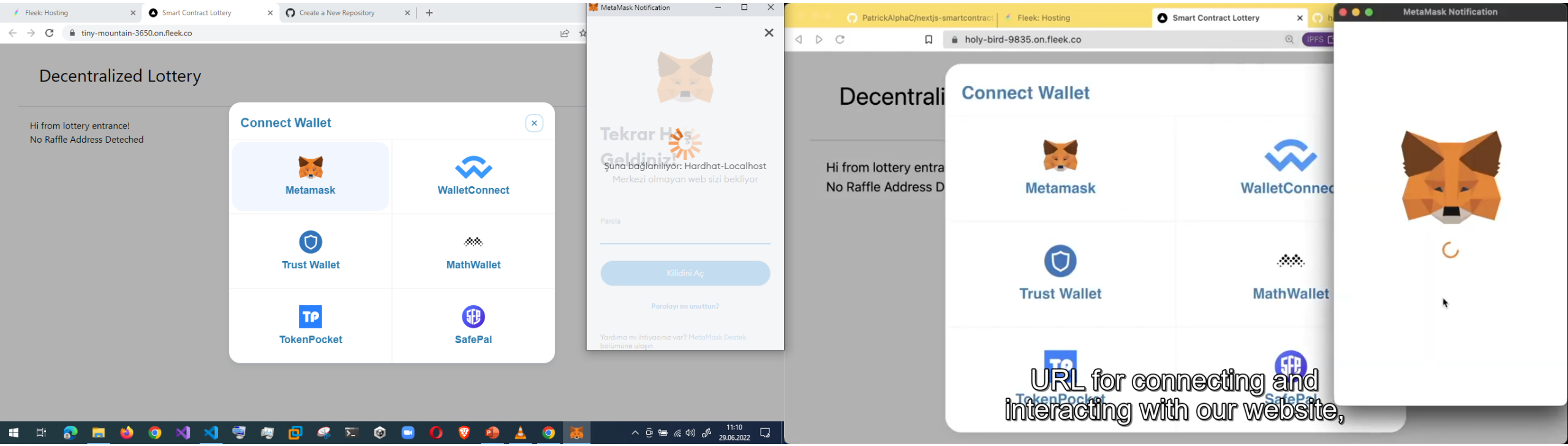
Preview | Copy to Clipboard

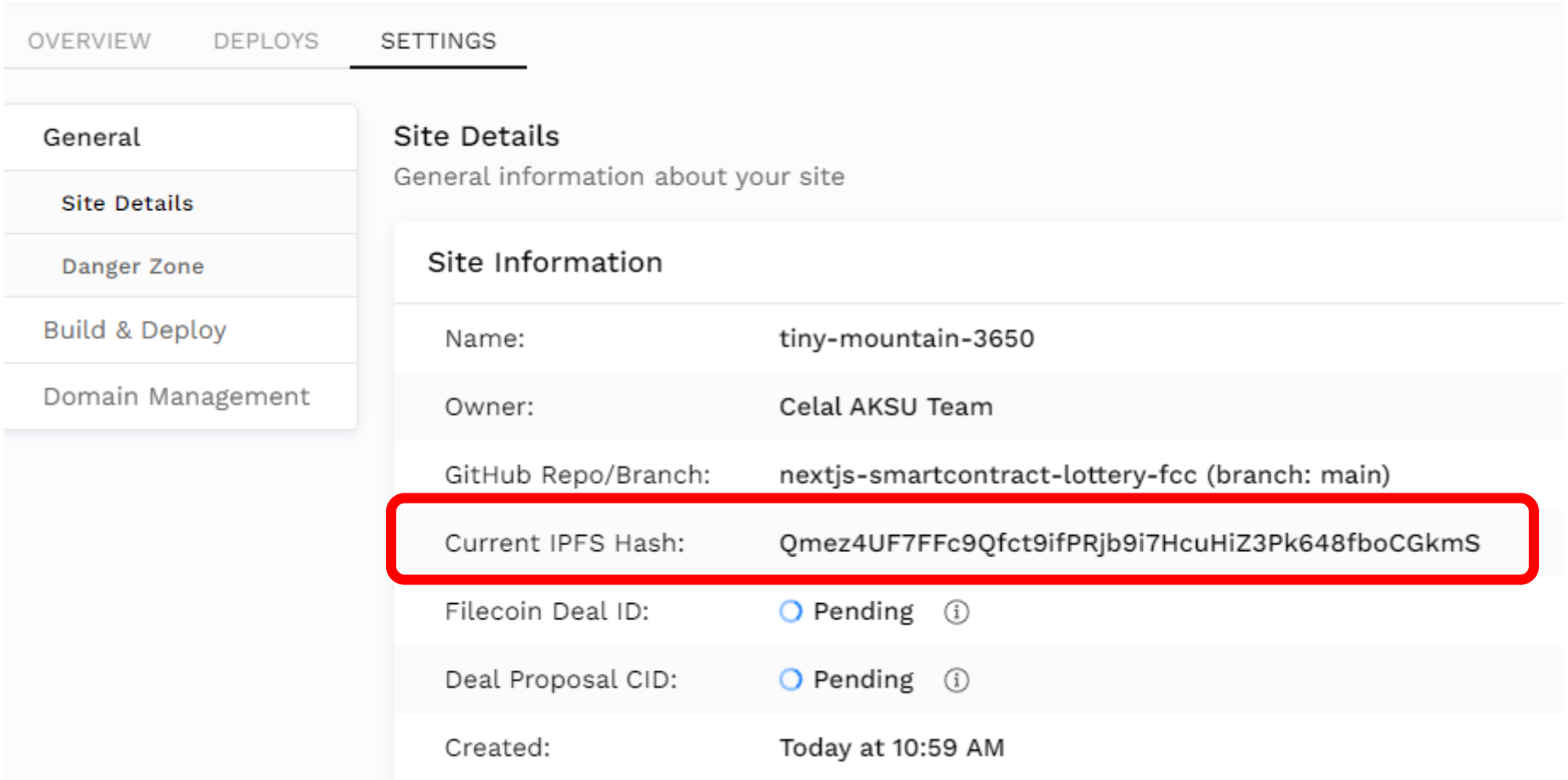
Deploy started at 10:59:59 AM 06/29/2022

Filecoin Deal ID : This is blockchain that helps you pin your data/site and uses decentralized storeage to do.
Fleek helps you create those deals and helps you pin your data with this filecoin Blockchain.

After deploy you can see settings from Hosting section on fleek.co

WHEN U CHANGE ANYTHING ON GITHUB REPO;
FLEEK.CO AUTOMATICALLY DEPLOYS SITE AGAIN

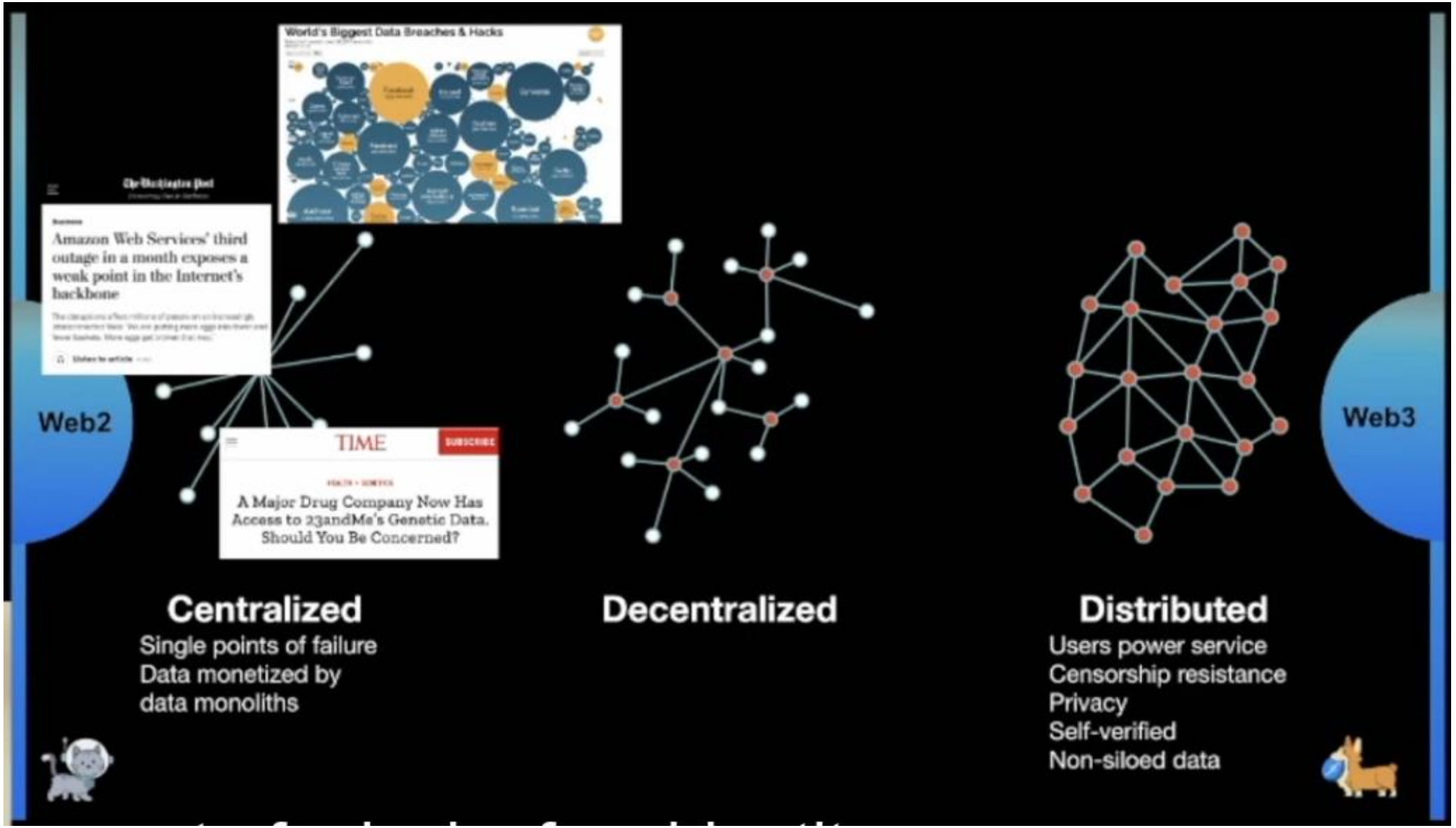

Fleek: Hosting
Smart Contract Lottery
Create a New Repository
tiny-mountain-3650.on.fleek.co
Decentralized Lottery
Hi from lottery entrance!
No Raffle Address Detected
Connect Wallet
Metamask
WalletConnect
Trust Wallet
MathWallet
TokenPocket
SafePal
MetaMask Notification
Şuna bağlanılıyor: Hardhat-Localhost
Merkezi olmayan web sizi bekliyor
Parola
Kilidini Aç
Parolayı mı unuttun?
Yardıma mı ihtiyacınız var? MetaMask Destek bölümüne ulaşın
11:10
29.06.2022
PatrickAlphaC/nextjs-smartcontract
Fleek: Hosting
Smart Contract Lottery
holy-bird-9835.on.fleek.co
IPFS
Decentrali
Hi from lottery entra
No Raffle Address D
Connect Wallet
Metamask
WalletConnec
Trust Wallet
MathWallet
TokenPocket
SafePal
MetaMask Notification
URL for connecting and
interacting with our website,


OVERVIEW
DEPLOYS
SETTINGS
General
Site Details
Danger Zone
Build & Deploy
Domain Management
Site Details
General information about your site
Site Information
Name:
tiny-mountain-3650
Owner:
Celal AKSU Team
GitHub Repo/Branch:
nextjs-smartcontract-lottery-fcc (branch: main)
Current IPFS Hash:
Qmez4UF7FFc9Qfct9ifPRjb9i7HcuHiZ3Pk648fboCGkmS
Filecoin Deal ID:
Pending
Deal Proposal CID:
Pending
Created:
Today at 10:59 AM

Filecoin Overview 18.31.27

Data, data everywhere...
but what to do with it?

Data, information and knowledge are some of the most important assets of our connected era and have become critical to human development & cooperation.


World's Biggest Data Breaches & Hacks
Amazon Web Services' third outage in a month exposes a weak point in the Internet's backbone
TIME
SUBSCRIBE
A Major Drug Company Now Has Access to 23andMe's Genetic Data. Should You Be Concerned?
Web2
Web3
Centralized
Single points of failure
Data monetized by data monoliths
Decentralized
Distributed
Users power service
Censorship resistance
Privacy
Self-verified
Non-siloed data

IPFS
Interplanetary File System

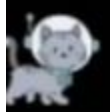
File system

Files + Folders

Interplanetary

Distributed (no central server)

Resilient / Offline first



IPFS
Interplanetary File System

IPFS addresses content by **what** it is, instead of **where** it is

It replaces a folder or file location **with a Content ID**

Computer	**file://** path/to/index.html
Web2 IP + port	**http://domain.com/** path/to/index.html
IPFS Content ID	**ipfs://[CID]/** path/to/index.html

Cryptographic Hash function
(Secure Hash Algorithm-256)

Content

```
hello world
```

Content Identifier (CID)

```
QmbfJNQGh73kr72RQ5h6nX8s9aN7aVKXiwGEYabxx3eQT9
```

Same content =
Same CID

Content ID can be **reproduced anytime** from the original content

Copies of content are **verifiable** by their CID

IPFS:

distributed system for storing and accesing

- files
- folders
- websites
- applications
- data

designed to be able to work even when the networks between planets

no central authority servers

designed to be offline first for resilience

IPFS protocol is the standard it usess for addresing content on the network

unique

uses content addresing. This means each peace of data, each meme or event full file system has its own unique cryptographically verifiable fingerprint, you might call it.

change one pixel of main image, the content ID or CID also changes

hash function is upgradeable

quantum computing breaks out current secure hash algorithms, it can upgragede standad. It means you can get the same content.

Using IPFS in the wild

No nodes, no retrievability…so who runs p2p nodes and why?

1a You can run your own node,...

1b ...run your own network of nodes,... (and an infrastructure team to manage it)

2 ...pay a pinning service to pin your content on their nodes (Pinata, Temporal, Infura + more)

3 ...and/or hope your content becomes popular enough that other nodes will pin it.

Filecoin is storage designed for Web3 from the ground up

Avg* Price
$5.484e-4
TiB/year

100% Verifiably stored with trackable cryptographic proofs

Proof of replication:
Ensures that providers are **actually** storing data and keeping it safe

Proof of spacetime: Uses block rewards as incentives & collateral slashing as penalties

Permanence Control - Your data, Your choice!

Market Deals:
Gives you **control** over the permanence & resilience of your data

5 minutes or 500 years - **your** data, **your** choice

An internet-scale, decentralised storage network

18,000,000 TB of capacity across the globe provided by thousands of storage providers

Governed by consensus, instead of a single corporation

Community driven - by a foundation funded for 50 years of open community development

The anatomy of a Filecoin storage deal

1. A client makes a time-limited deal (e.g., via app or smart contract) to store data

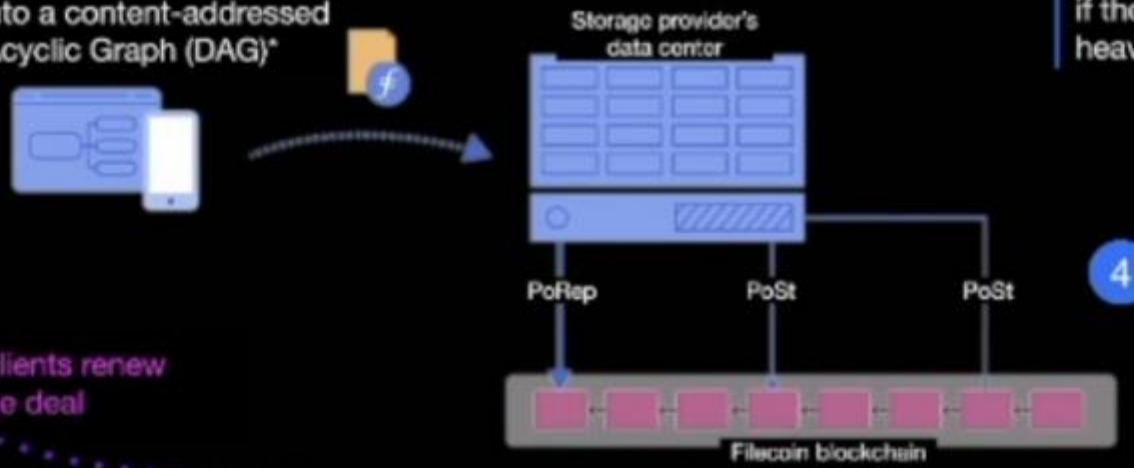
   The Filecoin protocol transforms the data into a content-addressed Directed Acyclic Graph (DAG)*

2. Providers generate an initial Proof of Replication (PoRep) to verify they received the client's data

   That PoRep is published to the Filecoin blockchain

3. Over time, providers prove that they still have random subsets of client data, creating Proofs of SpaceTime (PoST) published to the blockchain

   Providers are paid incrementally in FIL if they deliver the PoSTs and penalized heavily ("slashing") if they do not



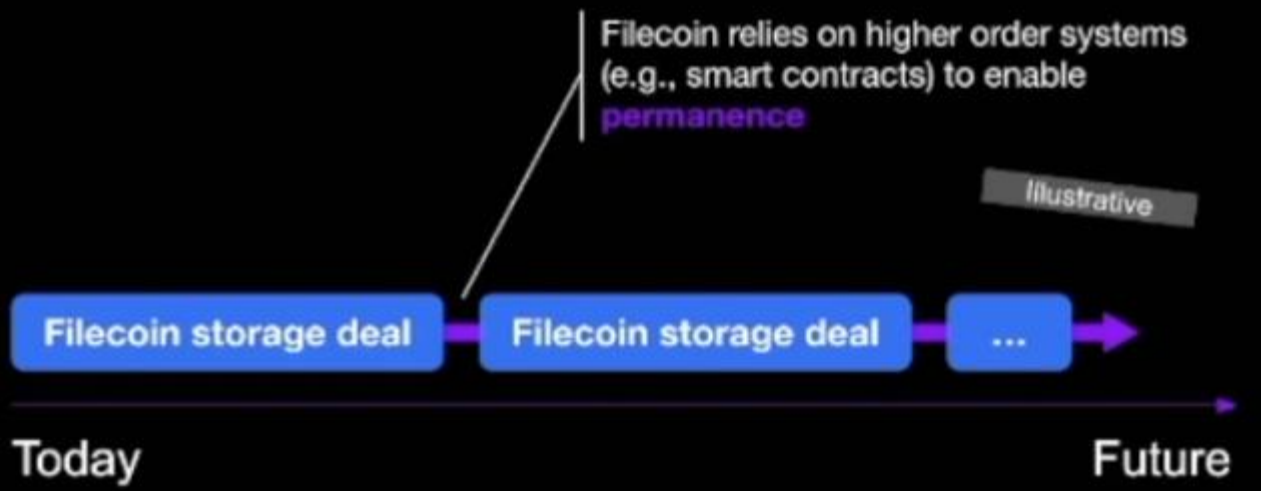
4. Towards the end of the storage deal, clients either renew the storage deal or let it expire

Optional: Clients renew their storage deal

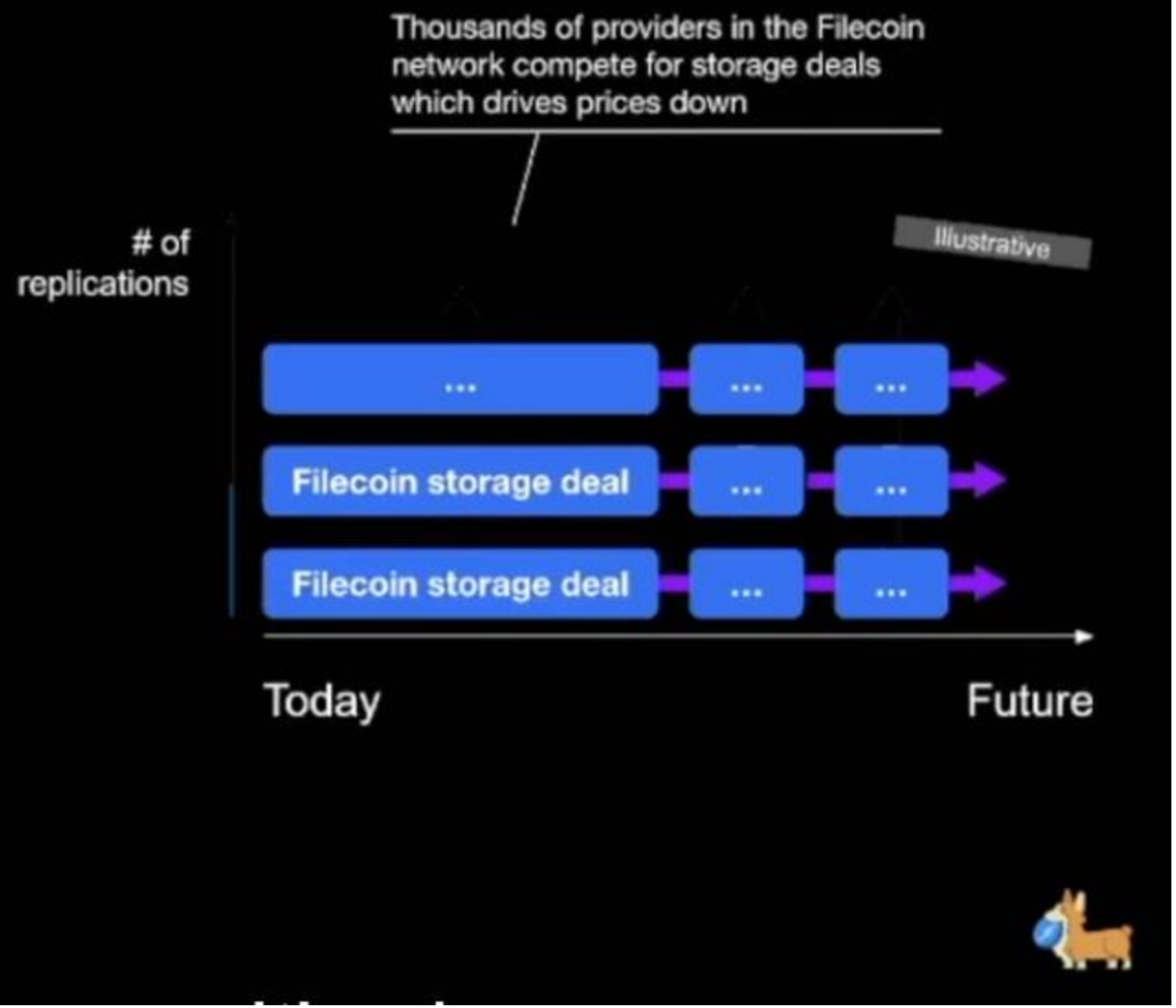
Optional: Clients make a retrieval deal to retrieve their data

* Filecoin uses the same process as IPFS so the resulting Content IDs are compatible

For permanence, any Storage deal can be renewed an infinite # of times by anyone or anything (e.g., a smart contract)

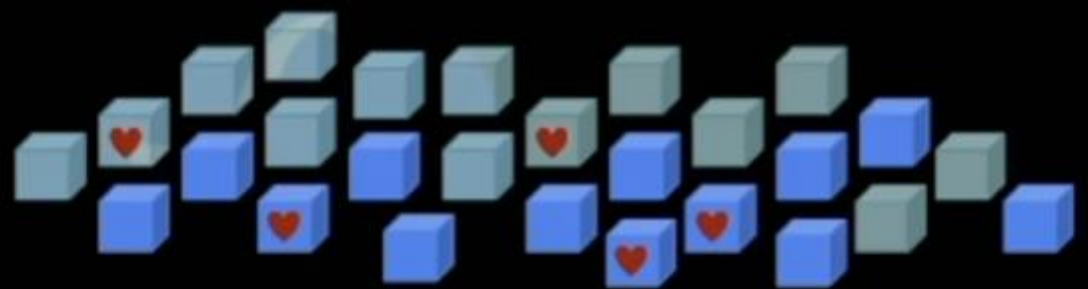


For redundancy, there can be an infinite number of copies of the same Storage deal



IPFS ❤️ Filecoin

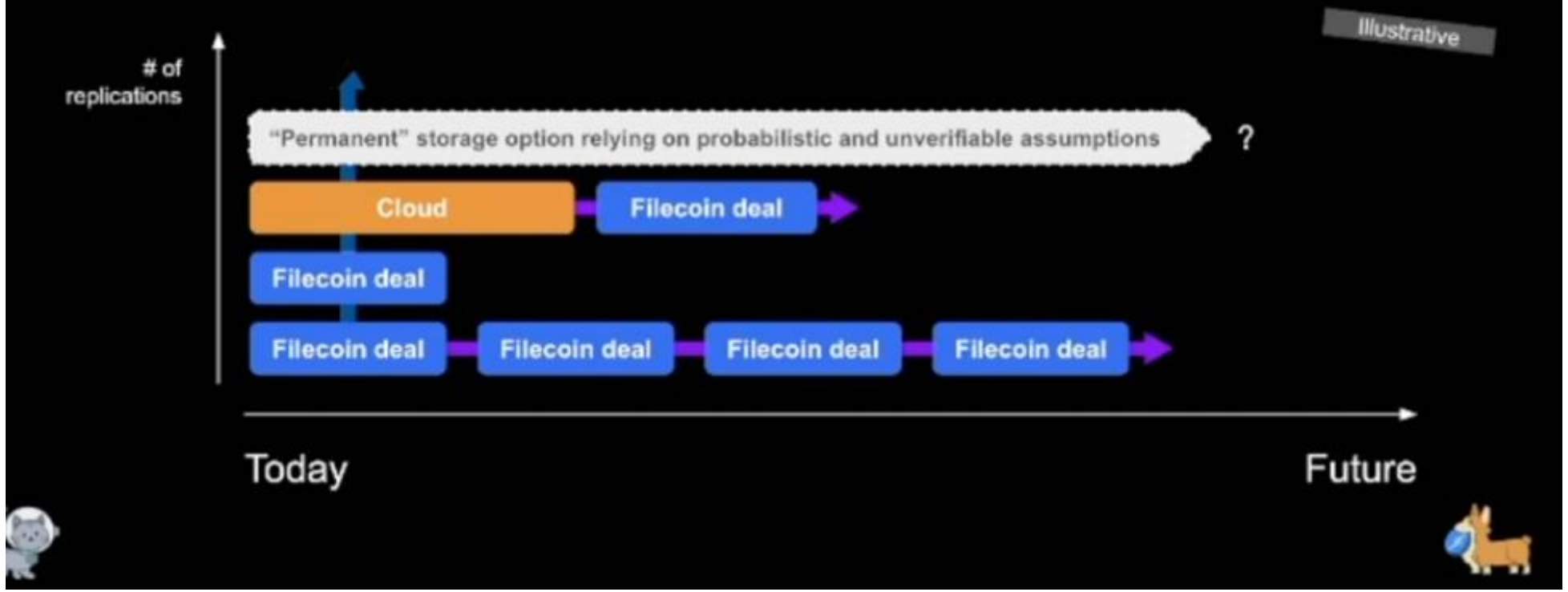
IPFS for fast, flexible retrieval (gateways, local nodes, browsers)



Filecoin for persistence and verifiability

Using IPFS with Filecoin future-proofs storage

The magic of IPFS Content IDs is that they're a property of the data itself which makes them storage-layer agnostics allowing for full flexibility and modularity



IPFS & Filecoin: DevTools

Aka I want to use these things - where do I start

The Projects

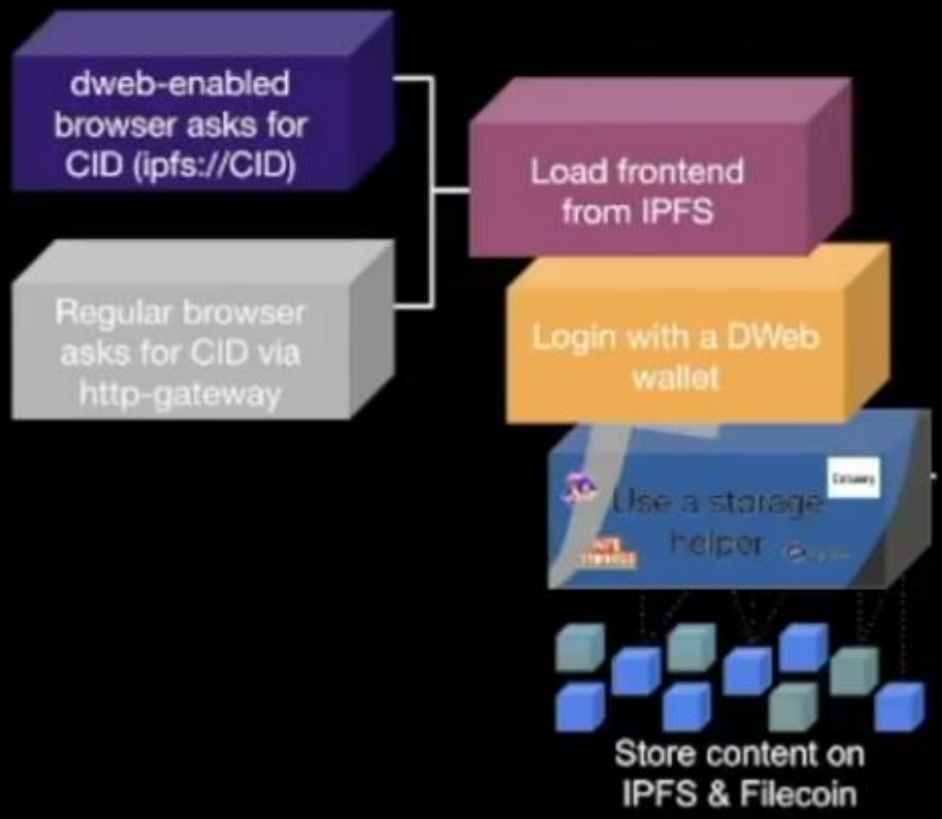
https://bit.ly/PL-Launchpad

Get started contributing to IPFS Projects, with these high-traffic repos

PL Repos

- go-ipfs is a high traffic project, with many contributors from outside PL.
- libp2p is the home of the OSS project that makes up the networking layer used by PL.
- ipfs-cluster provides data orchestration across a swarm of IPFS daemons by allocating, replicating and tracking a global pinset distributed among multiple peers.
- js-ipfs has multiple JS projects in a single repo. See the what they each do.
- multiformats is a library that defines common interfaces and low level building blocks for multiformat technologies (multicodec, multihash, multibase, and CID).
- js-libp2p is an implementation in JavaScript. This is a project that needs contribution and would be a good place for JS developers to dig in and learn about the PL Networking stack.
- Filecoin Improvement Requests (FIPs) contains the set of fundamental governing principles for the Filecoin Network. It outlines the vision for Filecoin and it also describes how improvements to these rules can be proposed and ratified.

Web3 all the way down



Use storage helpers

Simplify dealmaking
Find providers, negotiate prices, wait for confirmations, verify storage

Store to both IPFS and Filecoin with a single call
IPFS for fast, flexible retrieval, Filecoin for persistence and verifiability

Fleek hosting

Designed for

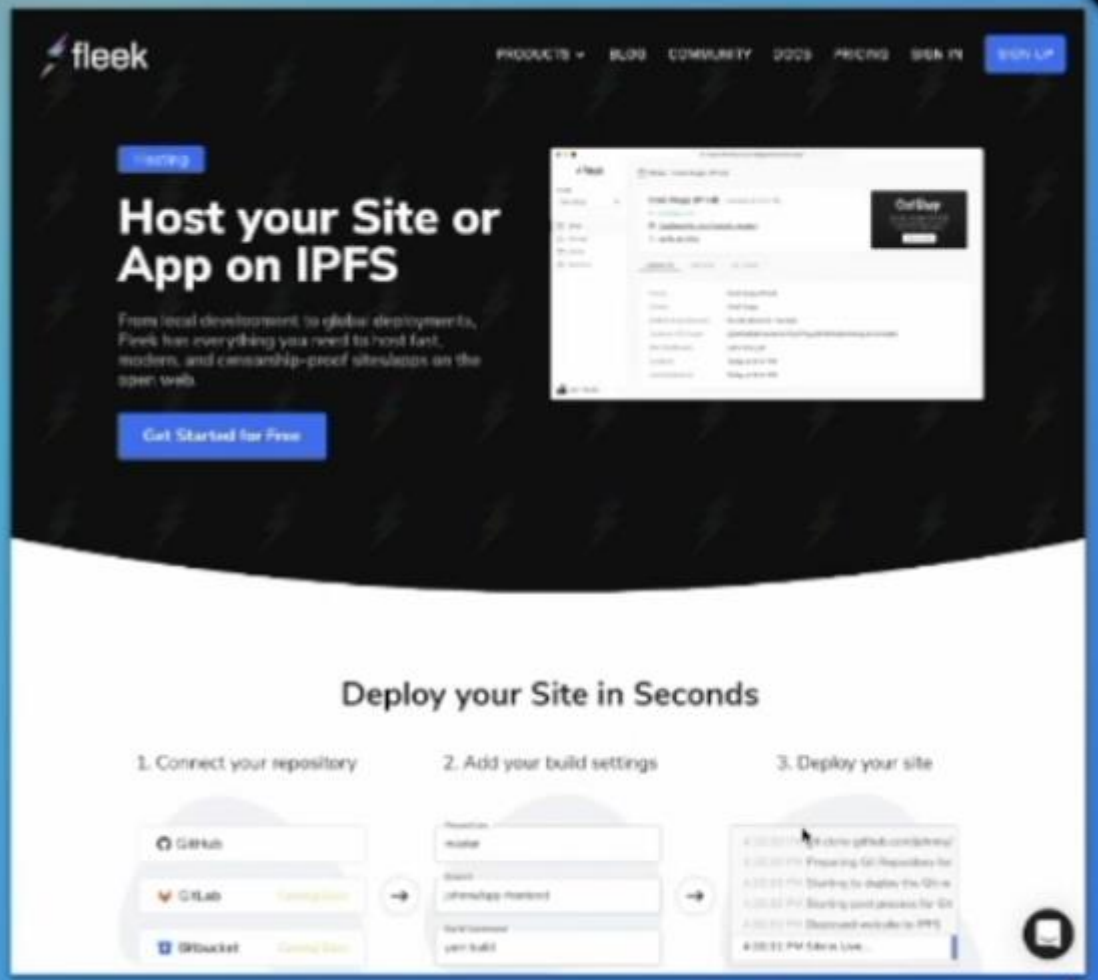
- Fast, modern, censorship-proof websites and web-apps on the open web
- Works with most modern frameworks (Docker, Gatsby, React, Webflow, Hugo, Next, Jekyll, etc.)

How it works

- Connect your Github repository
- Add build settings
- Deploy your site to IPFS
- Backed up to Filecoin

Usage

- Github Actions
- CLI
- GraphQL API



fleek.co/hosting

NFT.Storage

Designed for

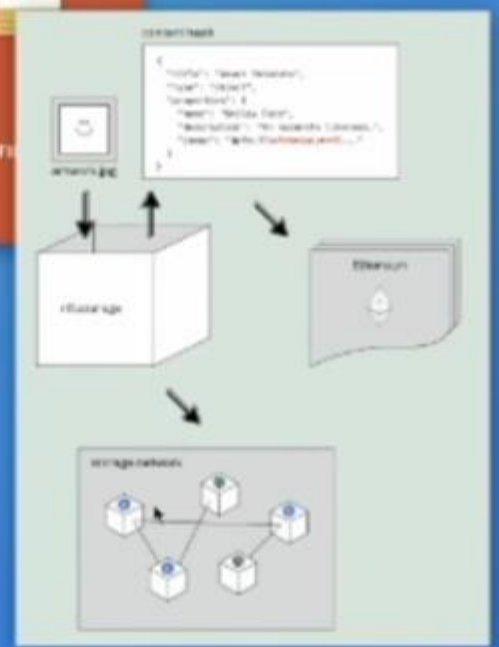
- NFT devs who want **free**, multi-generational, decentralized storage

How it works

- Compute CID of data locally that can be used in an NFT as a pointer to your content
- Once data is uploaded, available to retrieve via IPFS and backed up to Filecoin (>8x redundancy)
- Long-term decentralization in the works (smart contract-enforced redundancy, DAO-based funding)
- Store everywhere else that makes you comfortable

Usage

- Javascript client library
- HTTP API + remote pinning service
- Web interface



https://nft.storage/

Web3.Storage

Designed for

- General Filecoin & IPFS storage
- Familiar and simple interfaces
- Production-level storage and retrieval reliability + performance

How it works

- Compute CID of data locally and upload to edge worker
- Data immediately available to the IPFS network
- Automatically stores with 6+ Filecoin providers around the world
- 1TiB always free

Usage

- Javascript and Go client libraries
- HTTP API + remote pinning service
- Web interface

https://web3.storage/

Textile Powergate

Designed for

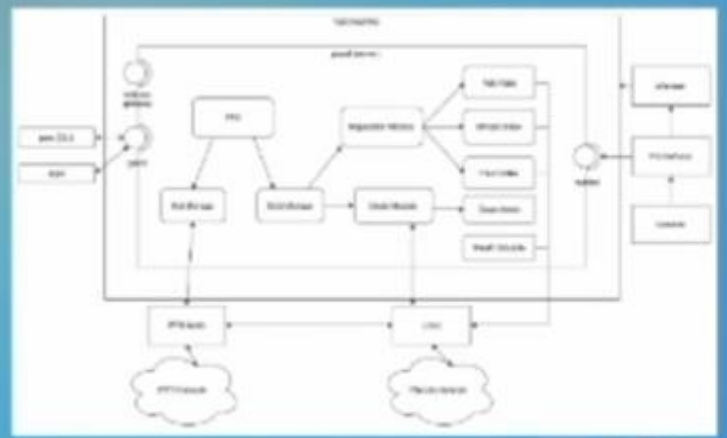
- Developers who want powerful ways to connect & extend Libp2p, IPFS, and Filecoin
- Bridges to NEAR and (coming soon) ETH, **Polygon**, & others

How it works

- Docker container wrapped around an IPFS node + Filecoin node
- Stage, store, and retrieve data
- Default configs for miner selection

Usage

- JS & Go Clients
- gRPC API
- CLI



docs.textile.io/powergate

Estuary.tech

Designed for

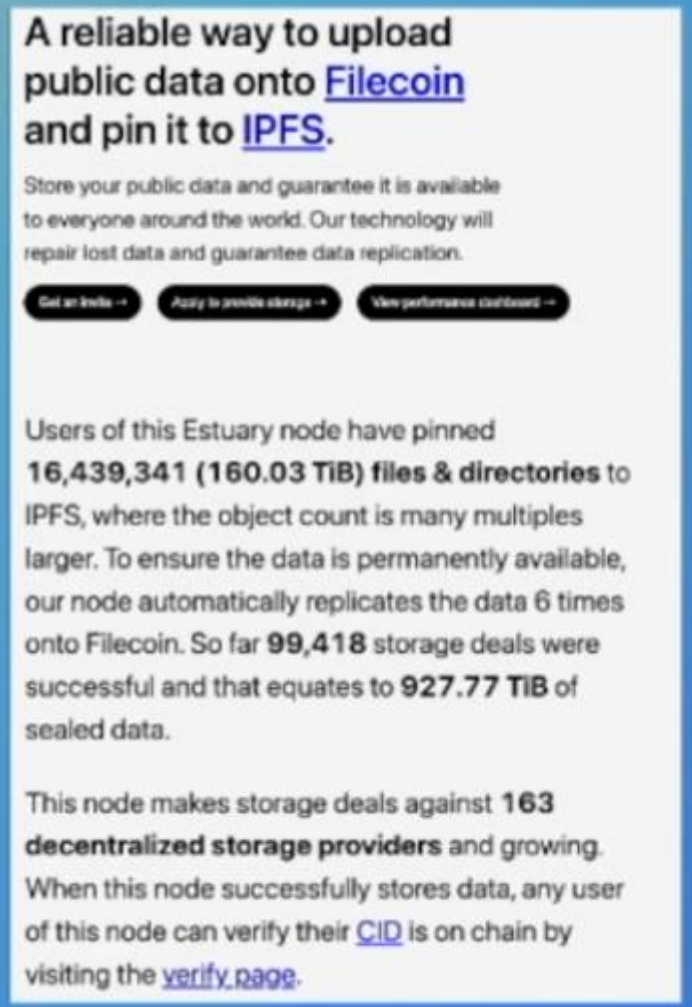
- People with large datasets or storing meaningful public data
- Invite only

How it works

- Runs Estuary nodes for IPFS & Filecoin
- Uses decentralized data-storage protocols: the Filecoin network and IPFS. The Filecoin network allows for *persistent*, *interoperable*, *verifiable*, and *provable* decentralized storage. IPFS is used for content addressing and cached retrievals.

Usage

- See the estuary docs



estuary.tech

OrbitDB

Designed for

- Peer-to-Peer distributed database

How it works

- Uses IPFS as data storage and IPFS Pubsub to automatically sync databases with peers
- Implemented on top of ipfs-log

Usage

- Currently in active development
- Works in both browsers & node.js
- Compatible with js-ipfs & go-ipfs

github.com/orbitdb

Web3-enabled Architectures

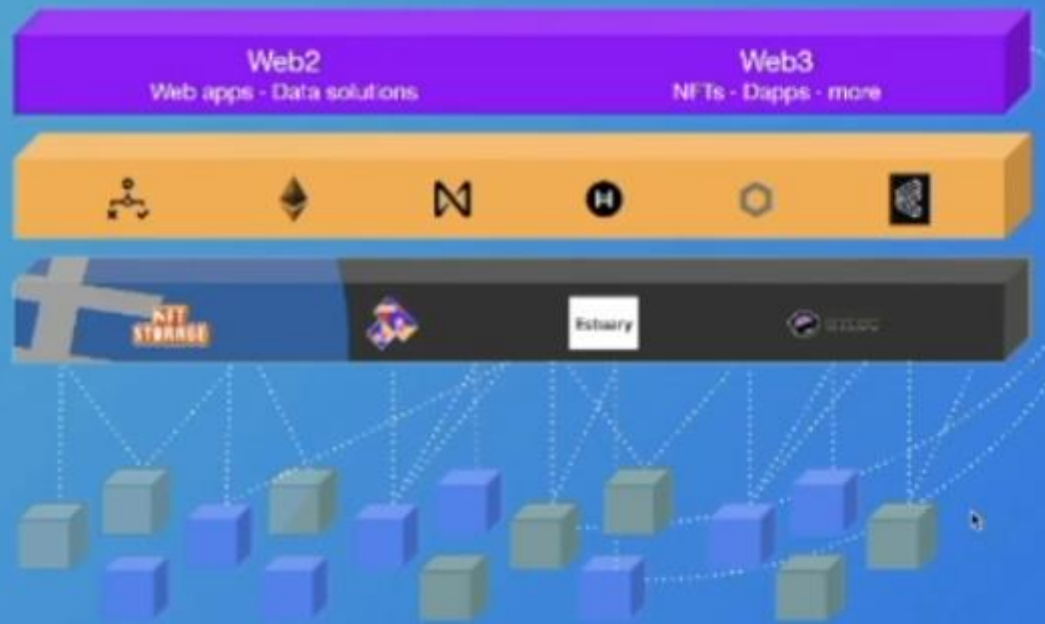
...with decentralized possibilities at every layer

Frontend

Logic
Standard application logic or smart contracts

Storage Helpers
Simplify storing & retrieving data (SDKs, hosted APIs, and more)

Filecoin & IPFS nodes



Filecoin Virtual Machine (FVM)

The FVM brings general programmability and EVM-compatible smart contacts to the Filecoin blockchain!!!!

https://fvm.filecoin.io/

Storage + Smart Contracts ❤️

IPFS & Filecoin: Resources & Getting Involved

Resources

Learn

- proto.school
- NFTschool.dev

Contribute & Grants

- Hackathons.filecoin.io
- Grants (microgrants, specific projects grants + bounties)
- Github repo's are all open with open bounties on many

https://tiny-mountain-3650.on.fleek.co/

The end of lesson

Typescript version in github

Deploying on rinkeby testnet

hh deploy --network rinkeby

```
--------------------------------------------------
Updating front end...
eemcs@DESKTOP-LJJC06I:~/freecodecamp/hardhat-smartcontract-lottery-cc$ hh deploy --network rinkeby
Nothing to compile
reusing "Raffle" at 0xF20E5E203dd1C1956dF8c60eEa3e787Ce5E1fA4E
Verifiying....
Verifying contract...
Nothing to compile
Already Verified!

--------------------------------------------------
Updating front end...
```

Contract address
0xF20E5E203dd1C1956dF8c60eEa3e787Ce5E1fA4E

https://tiny-mountain-3650.on.fleek.co/

İpfs://QmPZ7K8XPZf3HjDHRZe7HGkbGkndWRD6VSuoyk1CXg6T5T