Lesson Type: Practical   Estimated Time: 1-2 hours   Current Score: 0%

The Ethereum Name Service

Welcome to LearnWeb3 Punks - snoopies0839.eth!

Its an NFT collection for LearnWeb3 students.

Background

When web initially started the only way you could explore information on the web was by entering in its IP address. After that the concept of DNS was introduced which helped us to link a domain name to an IP address.

So whenever you type `learnweb3.io`, DNS takes care of translating it to the respective IP which is what the computer finally understands.

What is ENS?

ENS stands for `The Ethereum Name Service` and it behaves very similar to how DNS behaves in the web2 space. As we all know that Ethereum has long addresses which are hard to remember or type. ENS solves this issue by translating these wallet addresses, hashes etc into readable domains which are then saved on Ethereum blockchain.

The best part about ENS is unlike DNS servers which are centralized, ENS works with the help of a smart contract which is censorship resistant. So now when you are sending your wallet address to someone which looks like `0x1234huiahi....` you can actually send them `tom.eth` and the ENS would figure out that `tom.eth` is actually equal to your wallet address (`0x1234huiahi....`)

Additionally, ENS extends beyond just mapping wallet addresses to human-readable names. You can actually attach a profile picture, a description, social media links, as well as any custom types of data you'd want to attach.

Requirements

Its time to build something where we can use ENS. We will develop a website which can display the ENS for an address if it has one.

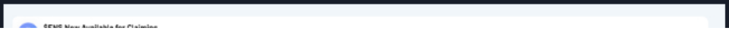
Lets gooo 🚀

Setup

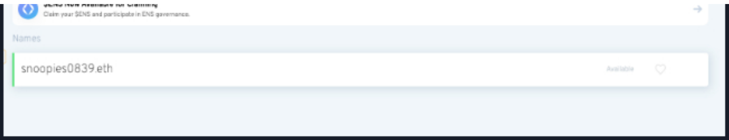
First lets get an ENS name for your address, start by opening up https://app.ens.domains/

Make sure when you open the website, your MetaMask is connected to the `Goerli Testnet` and it has some `Goerli Ether`

Search for an ENS domain name, any name you like, as long as it is available!. Click on `Available`



$ENS Now Available for Claiming

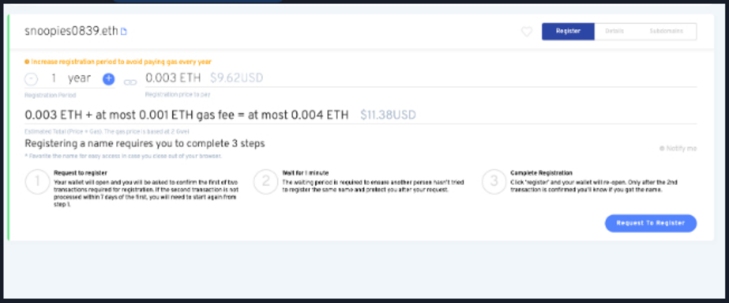


$ENS Now Available for Claiming
Claim your $ENS and participate in ENS governance.

Names

snoopies0839.eth

Then click on `Request To Register`



snoopies0839.eth

● Increase registration period to avoid paying gas every year

1 year — 0.003 ETH $9.62USD

Registration Period — Registration price to pay

0.003 ETH + at most 0.001 ETH gas fee = at most 0.004 ETH $11.38USD

Estimated Total (Price + Gas). The gas price is based at 2 Gwei

Registering a name requires you to complete 3 steps

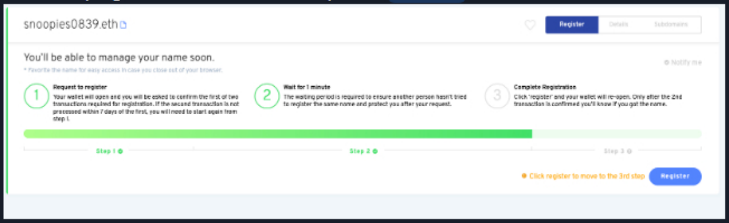
* Favorite the name for easy access in case you close out of your browser.

1 Request to register
Your wallet will open and you will be asked to confirm the first of two transactions required for registration. If the second transaction is not processed within 7 days of the first, you will need to start again from step 1.

2 Wait for 1 minute
The waiting period is required to ensure another person hasn't tried to register the same name and protect you after your request.

3 Complete Registration
Click 'register' and your wallet will re-open. Only after the 2nd transaction is confirmed you'll know if you got the name.

When the progress bar enters the 3rd step, Click `Register`



snoopies0839.eth

You'll be able to manage your name soon.

* Favorite the name for easy access in case you close out of your browser.

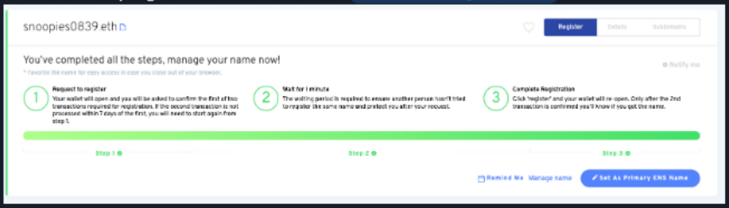
1 Request to register
Your wallet will open and you will be asked to confirm the first of two transactions required for registration. If the second transaction is not processed within 7 days of the first, you will need to start again from step 1.

2 Wait for 1 minute
The waiting period is required to ensure another person hasn't tried to register the same name and protect you after your request.

3 Complete Registration
Click 'register' and your wallet will re-open. Only after the 2nd transaction is confirmed you'll know if you got the name.

● Click register to move to the 3rd step

Then after the progress bar finishes click on `Set As Primary ENS Name`



snoopies0839.eth

You've completed all the steps, manage your name now!

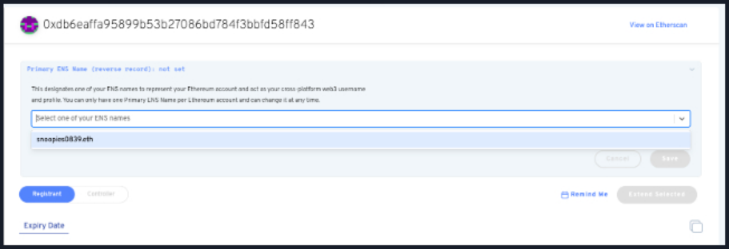
* Favorite the name for easy access in case you close out of your browser.

1 Request to register
Your wallet will open and you will be asked to confirm the first of two transactions required for registration. If the second transaction is not processed within 7 days of the first, you will need to start again from step 1.

2 Wait for 1 minute
The waiting period is required to ensure another person hasn't tried to register the same name and protect you after your request.

3 Complete Registration
Click 'register' and your wallet will re-open. Only after the 2nd transaction is confirmed you'll know if you got the name.

From the dropdown then select the ENS name you just created



0xdb6eaffa95899b53b27086bd784f3bbfd58ff843

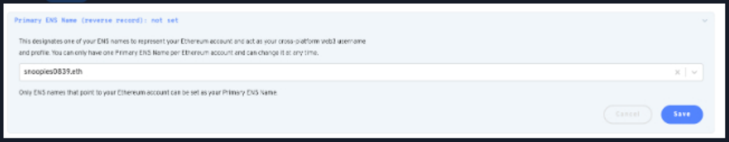
Primary ENS Name (reverse record): not set

This designates one of your ENS names to represent your Ethereum account and act as your cross-platform web3 username and profile. You can only have one Primary ENS Name per Ethereum account and can change it at any time.

Select one of your ENS names

snoopies0839.eth

Click `Save`



Primary ENS Name (reverse record): not set

This designates one of your ENS names to represent your Ethereum account and act as your cross-platform web3 username and profile. You can only have one Primary ENS Name per Ethereum account and can change it at any time.

snoopies0839.eth

Only ENS names that point to your Ethereum account can be set as your Primary ENS Name.

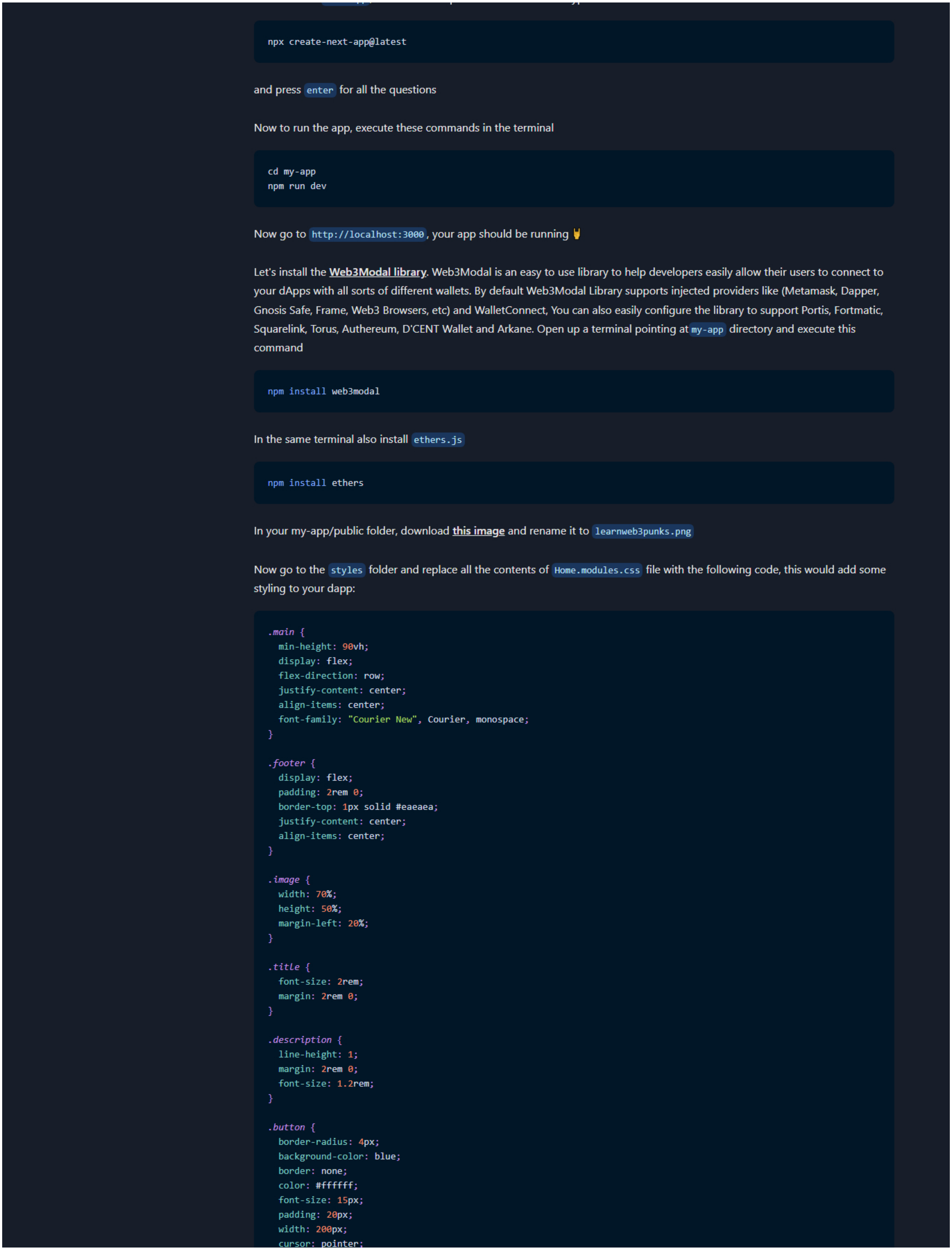
Now you have an ENS registered to your address on Goerli. Awesome, You did it ❤️

Website

To develop the website we will use **React** and **Next Js**. React is a javascript framework used to make websites and Next.js is a React framework that also allows writing backend APIs code along with the frontend, so you don't need two separate frontend and backend services. First, You will need to create a new `next` app. Your folder structure should look something like

```
- ENS
    - my-app
```

To create this `next-app`, in the terminal point to ENS folder and type



```
npx create-next-app@latest
```

and press `enter` for all the questions

Now to run the app, execute these commands in the terminal

```
cd my-app
npm run dev
```

Now go to `http://localhost:3000`, your app should be running 🤘

Let's install the **Web3Modal library**. Web3Modal is an easy to use library to help developers easily allow their users to connect to your dApps with all sorts of different wallets. By default Web3Modal Library supports injected providers like (Metamask, Dapper, Gnosis Safe, Frame, Web3 Browsers, etc) and WalletConnect, You can also easily configure the library to support Portis, Fortmatic, Squarelink, Torus, Authereum, D'CENT Wallet and Arkane. Open up a terminal pointing at `my-app` directory and execute this command

```
npm install web3modal
```

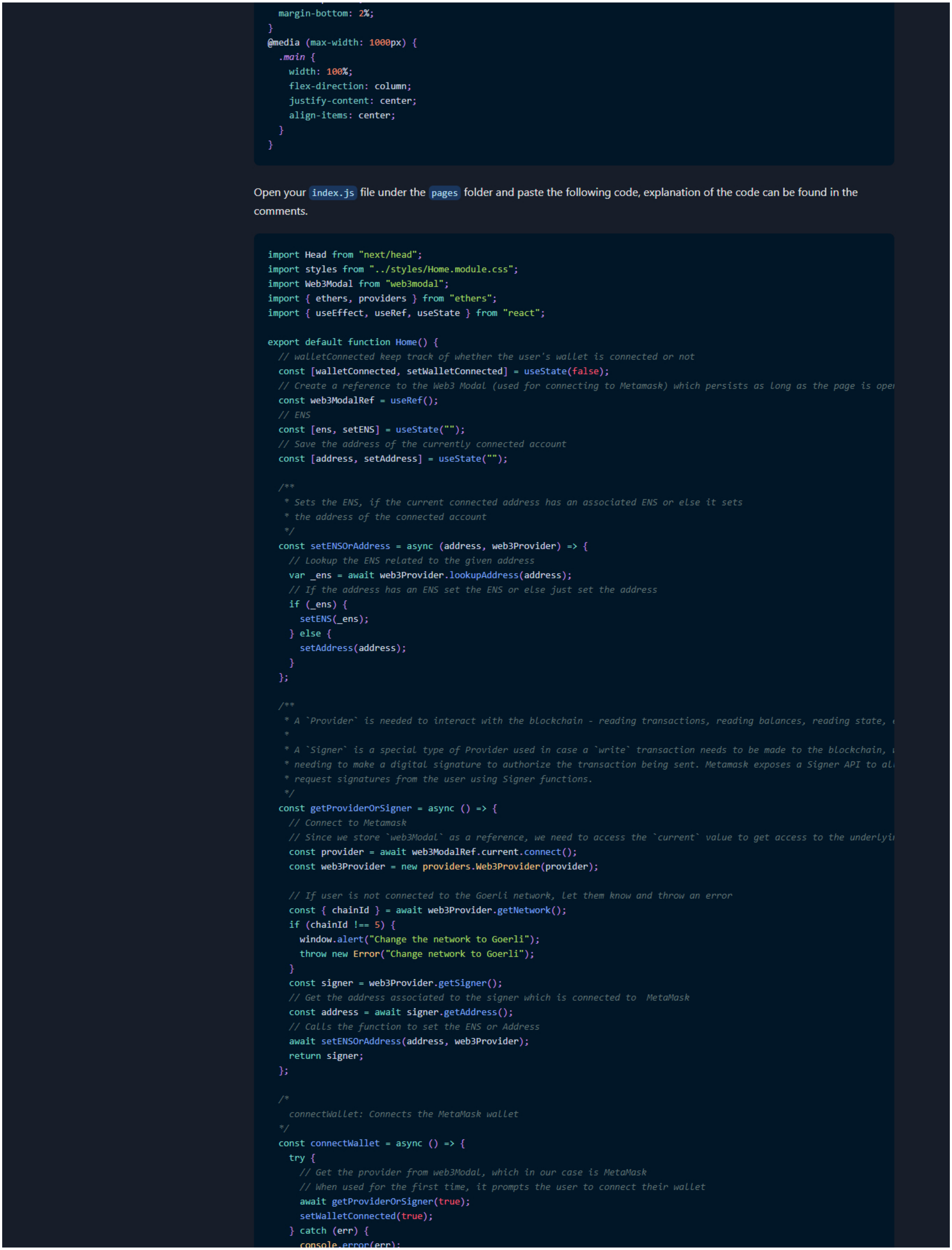
In the same terminal also install `ethers.js`

```
npm install ethers
```

In your my-app/public folder, download **this image** and rename it to `learnweb3punks.png`

Now go to the `styles` folder and replace all the contents of `Home.modules.css` file with the following code, this would add some styling to your dapp:

```
.main {
  min-height: 90vh;
  display: flex;
  flex-direction: row;
  justify-content: center;
  align-items: center;
  font-family: "Courier New", Courier, monospace;
}

.footer {
  display: flex;
  padding: 2rem 0;
  border-top: 1px solid #eaeaea;
  justify-content: center;
  align-items: center;
}

.image {
  width: 70%;
  height: 50%;
  margin-left: 20%;
}

.title {
  font-size: 2rem;
  margin: 2rem 0;
}

.description {
  line-height: 1;
  margin: 2rem 0;
  font-size: 1.2rem;
}

.button {
  border-radius: 4px;
  background-color: blue;
  border: none;
  color: #ffffff;
  font-size: 15px;
  padding: 20px;
  width: 200px;
  cursor: pointer;
```



```
  margin-bottom: 2%;
}
@media (max-width: 1000px) {
  .main {
    width: 100%;
    flex-direction: column;
    justify-content: center;
    align-items: center;
  }
}
```

Open your `index.js` file under the `pages` folder and paste the following code, explanation of the code can be found in the comments.

```
import Head from "next/head";
import styles from "../styles/Home.module.css";
import Web3Modal from "web3modal";
import { ethers, providers } from "ethers";
import { useEffect, useRef, useState } from "react";

export default function Home() {
  // walletConnected keep track of whether the user's wallet is connected or not
  const [walletConnected, setWalletConnected] = useState(false);
  // Create a reference to the Web3 Modal (used for connecting to Metamask) which persists as long as the page is ope
  const web3ModalRef = useRef();
  // ENS
  const [ens, setENS] = useState("");
  // Save the address of the currently connected account
  const [address, setAddress] = useState("");

  /**
   * Sets the ENS, if the current connected address has an associated ENS or else it sets
   * the address of the connected account
   */
  const setENSOrAddress = async (address, web3Provider) => {
    // Lookup the ENS related to the given address
    var _ens = await web3Provider.lookupAddress(address);
    // If the address has an ENS set the ENS or else just set the address
    if (_ens) {
      setENS(_ens);
    } else {
      setAddress(address);
    }
  };

  /**
   * A `Provider` is needed to interact with the blockchain - reading transactions, reading balances, reading state,
   *
   * A `Signer` is a special type of Provider used in case a `write` transaction needs to be made to the blockchain,
   * needing to make a digital signature to authorize the transaction being sent. Metamask exposes a Signer API to al
   * request signatures from the user using Signer functions.
   */
  const getProviderOrSigner = async () => {
    // Connect to Metamask
    // Since we store `web3Modal` as a reference, we need to access the `current` value to get access to the underlyi
    const provider = await web3ModalRef.current.connect();
    const web3Provider = new providers.Web3Provider(provider);

    // If user is not connected to the Goerli network, let them know and throw an error
    const { chainId } = await web3Provider.getNetwork();
    if (chainId !== 5) {
      window.alert("Change the network to Goerli");
      throw new Error("Change network to Goerli");
    }
    const signer = web3Provider.getSigner();
    // Get the address associated to the signer which is connected to  MetaMask
    const address = await signer.getAddress();
    // Calls the function to set the ENS or Address
    await setENSOrAddress(address, web3Provider);
    return signer;
  };

  /*
    connectWallet: Connects the MetaMask wallet
  */
  const connectWallet = async () => {
    try {
      // Get the provider from web3Modal, which in our case is MetaMask
      // When used for the first time, it prompts the user to connect their wallet
      await getProviderOrSigner(true);
      setWalletConnected(true);
    } catch (err) {
      console.error(err);
      console.error(err);
    }
  };

  /*
    renderButton: Returns a button based on the state of the dapp
  */
  const renderButton = () => {
    if (walletConnected) {
      <div>Wallet connected</div>;
    } else {
      return (
        <button onClick={connectWallet} className={styles.button}>
          Connect your wallet
        </button>
      );
    }
  };

  // useEffects are used to react to changes in state of the website
  // The array at the end of function call represents what state changes will trigger this effect
  // In this case, whenever the value of `walletConnected` changes - this effect will be called
  useEffect(() => {
    // if wallet is not connected, create a new instance of Web3Modal and connect the MetaMask wallet
    if (!walletConnected) {
      // Assign the Web3Modal class to the reference object by setting it's `current` value
      // The `current` value is persisted throughout as long as this page is open
      web3ModalRef.current = new Web3Modal({
        network: "goerli",
        providerOptions: {},
        disableInjectedProvider: false,
      });
      connectWallet();
    }
  }, [walletConnected]);

  return (
    <div>
      <Head>
        <title>ENS Dapp</title>
        <meta name="description" content="ENS-Dapp" />
        <link rel="icon" href="/favicon.ico" />
      </Head>
      <div className={styles.main}>
        <div>
          <h1 className={styles.title}>
            Welcome to LearnWeb3 Punks {ens ? ens : address}!
          </h1>
          <div className={styles.description}>
            Its an NFT collection for LearnWeb3 Punks.
          </div>
          {renderButton()}
        </div>
        <div>
          <img className={styles.image} src="./learnweb3punks.png" />
        </div>
      </div>

      <footer className={styles.footer}>
        Made with &#10084; by LearnWeb3 Punks
      </footer>
    </div>
  );
}
```

Now in your terminal which is pointing to my-app folder, execute

```
npm run dev
```

Your ENS dapp should now work without errors 🚀

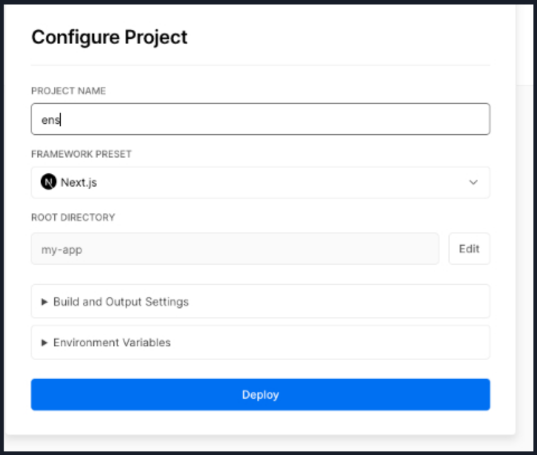
Push to github

Make sure before proceeding you have pushed all your code to github :)

Deploying your dApp

Deploying your dApp

We will now deploy your dApp, so that everyone can see your website and you can share it with all of your LearnWeb3 DAO friends.

- Go to **Vercel** and sign in with your GitHub
- Then click on `New Project` button and then select your ENS dApp repo
- 

Configure Project

PROJECT NAME

ens

FRAMEWORK PRESET

Next.js

ROOT DIRECTORY

my-app

- When configuring your new project, Vercel will allow you to customize your `Root Directory`
- Click `Edit` next to `Root Directory` and set it to `my-app`
- Select the Framework as `Next.js`
- Click `Deploy`
- Now you can see your deployed website by going to your dashboard, selecting your project, and copying the URL from there!

To pass the skill test for this level, input YOUR address you used to buy the ENS domain from in the verification box.

Share your website on Discord :D and as usual, feel free to ask any questions!

Submit Practical

Verify your smart contract address to pass the assessment for this level.

Ethereum Goerli

0x855267A0580836767bAFa23579BE1Ae1e37293dE

Submit

LearnWeb3

© 2022 LearnWeb3 DAO.

Product

Dashboard

Courses

Community

Blog

Company

About

Work With Us

We're Hiring

Buy us a coffee

Stay up to date

Your email address