**How to Build Ethereum dApps in Angular**

**Blockchain technology is facilitating the development of Web 3.0. Distributed apps are becoming the next gold rush on the internet. Nevertheless, there are many lucrative opportunities in the blockchain space for dApp developers. When you build Ethereum dApps in Angular and Moralis, it speeds up the development process while increasing the chances of your dApp business success. Angular and Moralis combined can make the Ethereum dApp development process super easy and fast.**

**Start by** [**signing up with Moralis**](https://admin.moralis.io/register)**. Moralis is free and gives you snippets of JavaScript to copy into your code to help you create the dApp backend Infrastructure according to your taste. You can save months of dApp coding and technical preparation time using Moralis.**

**What is Angular and Ethereum Angular dApps?**

[Angular](https://moralis.io/angular-explained-what-is-angular/) is a scalable and open-source web application framework. It offers numerous advantages for Ethereum dApp development. The coding process of dApp becomes fast and simplistic. It enables you to create high-quality frontend UIs for dApps.



Ethereum is a blockchain with its cryptocurrency called Ether. It offers the advantage of storing data on the blockchain that is impossible to tamper with or hack. Decentralized Applications (dApps) exploit this data.

Blockchain technology offers several advantages over conventional apps (Applications). When you add the UI to your Ethereum dApp using Angular, it is safe to call it an Ethereum Angular dApp. It means that you are creating the dApp using the Ethereum blockchain technology with a user interface coded in Angular.

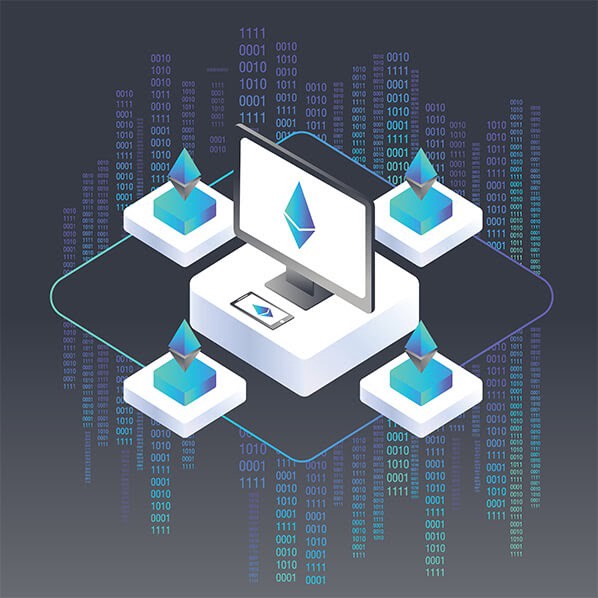
You can also build the frontend of your dApp with JavaScript bypassing Angular, but you will miss out on the powerful features of Angular to develop dApps. Moreover, UIs that Angular offers translate into unparalleled simplicity, significantly cutting down the coding time. Angular dApps uses the very simple but powerful language TypeScript that offers many advantages over JavaScript. The Angular TypeScript code ultimately converts into JavaScript before execution at runtime.

**The Ethereum Blockchain**

Ethereum is not the only blockchain platform that offers to build dApps. There are other platforms like Polygon, Bsc, Avalanche, and Fantom. Moralis supports all the mentioned four popular blockchain platforms at the time of this writing for creating dApps. More dApp blockchain platforms are in the development pipeline of the Moralis team.

Ethereum blockchain is the most popular option to store any data in its blockchain. The distributed applications harness this stored data taking advantage of the power of the blockchain network. It is worth mentioning that all the blockchain networks do not allow the creation of dApps.

Ethereum offers cryptocurrency by the name of Ether while being the largest, most well-established, and most secure, open-ended decentralized platform.



Another technology that can help you build Ethereum dApps in Angular is Moralis. Moralis is fast emerging as the [Web 3.0](https://moralis.io/the-ultimate-guide-to-web3-what-is-web3/) Operating System. The dApp development process becomes more streamlined and straightforward when you use Moralis since the platform does all the heavy lifting for you at the backend. It provides you with the backend infrastructure that every dApp needs.

Thus, the duration is to set up a backend infrastructure for your dApp is cut down to a few hours. It will take several months if you attempt to set up and code the same backend functionality that Moralis offers for your dApp. Moreover, there is a high probability that you would have to go through the same laborious process for your next dApp project without Moralis.

[](https://www.youtube.com/embed/UTJMZ7tWJvA?feature=oembed)

**Why Use Moralis as your Ethereum dApp Backend?**

When you build Ethereum dApps in Angular framework, 90% to 95% of the dApp have the same backend infrastructure. Moralis provides it free of charge. You can focus on and spend more time enhancing the frontend that your dApp, users will adore. Moreover, Moralis' creators intend to keep most of the services to power your dApps free of charge well into the future. Moralis is also infinitely scalable; Moralis grows with your dApp business and will offer seamless support for your dApp backend, so do try Moralis out.

Do you think you can build Ethereum dApps in Angular that can form headlines? Alternatively, do you want to play around with code that may become something impressive?

Whatever your intentions may be, Moralis is the platform that can help you get started quickly and launch your idea in no time.

Moralis platform provides you with the boilerplate code facilitating integration with Angular framework right out of the box. The platform offers several features that almost all dApps require at their backend.

Moralis adds these functionalities for your dApp users in a matter of a few hours rather than months. dApp login may tempt your users to search for information like all their transactions, assets such as NFTs, etc. More on this as you read along.

You can provide the user with a dashboard furnished with different kinds of information that all dApps need. You may give them this information in many different ways or forms using Moralis.

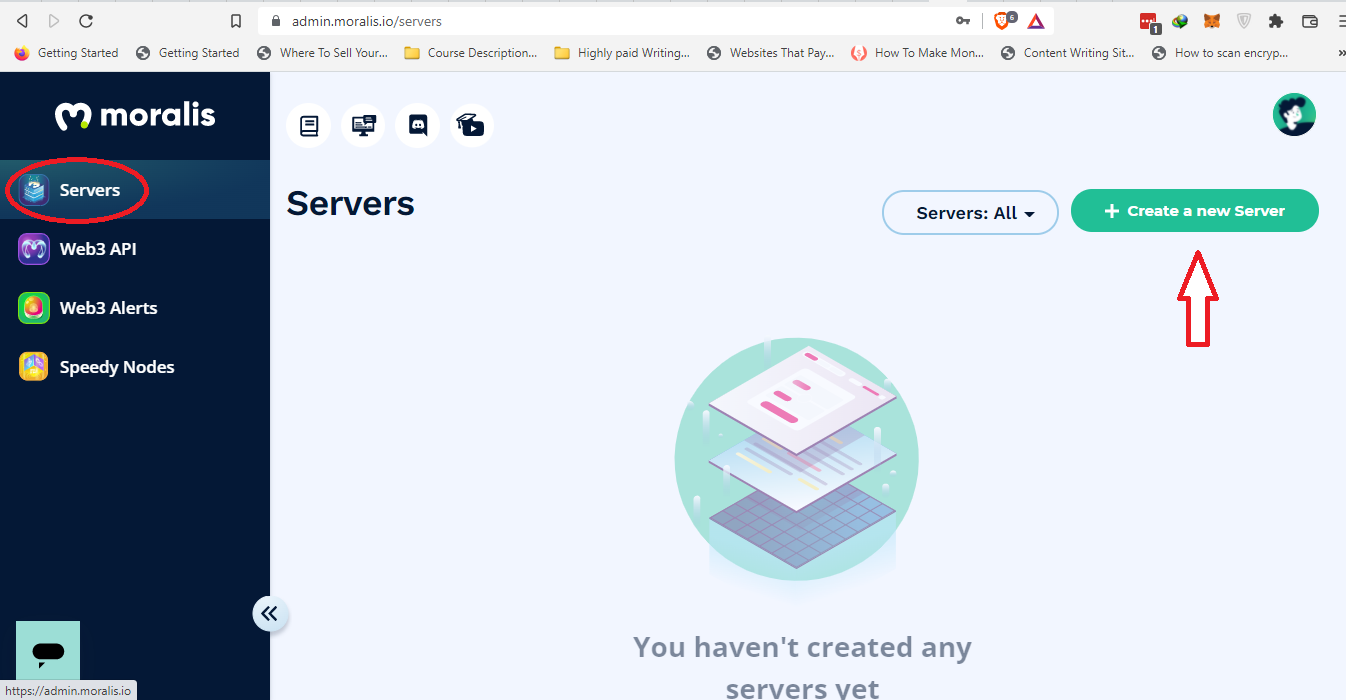
In other words, you do not have to reinvent the wheel while creating fascinating dApps.

**Using the Moralis SDK to Build Ethereum dApps in Angular**

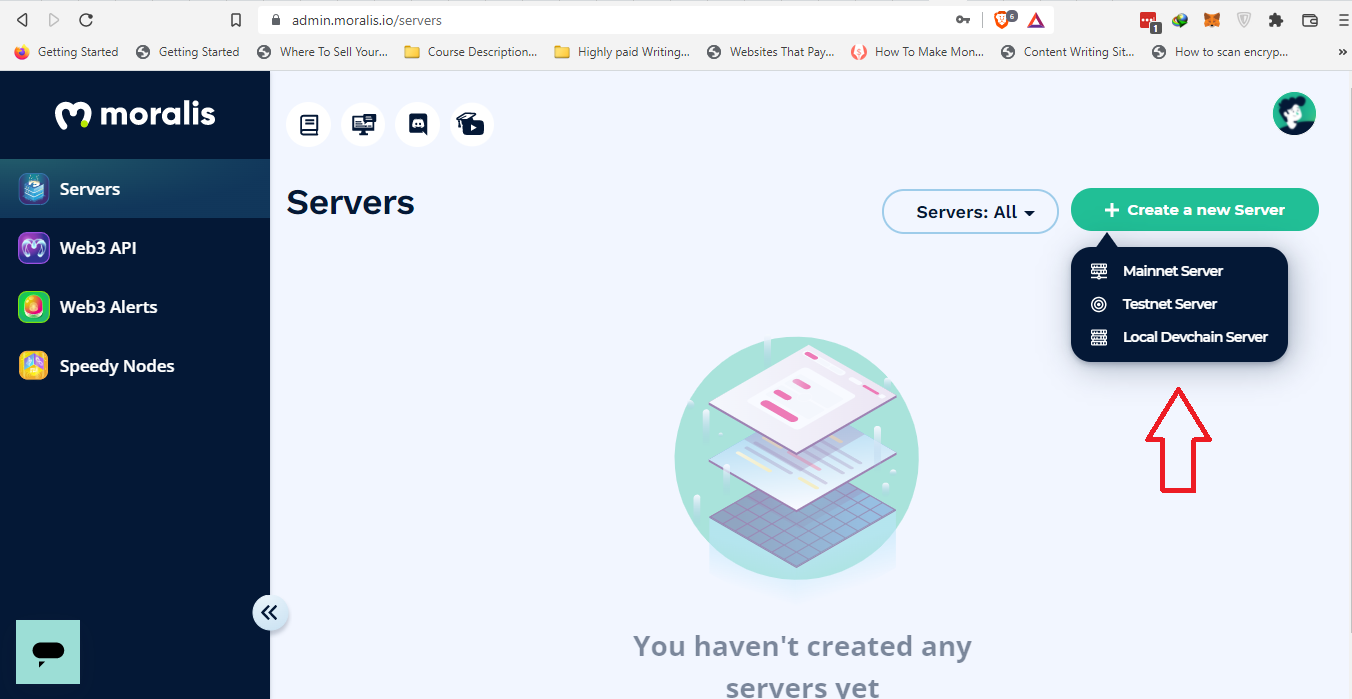
You can launch the Moralis server with one click after signing up with Moralis.

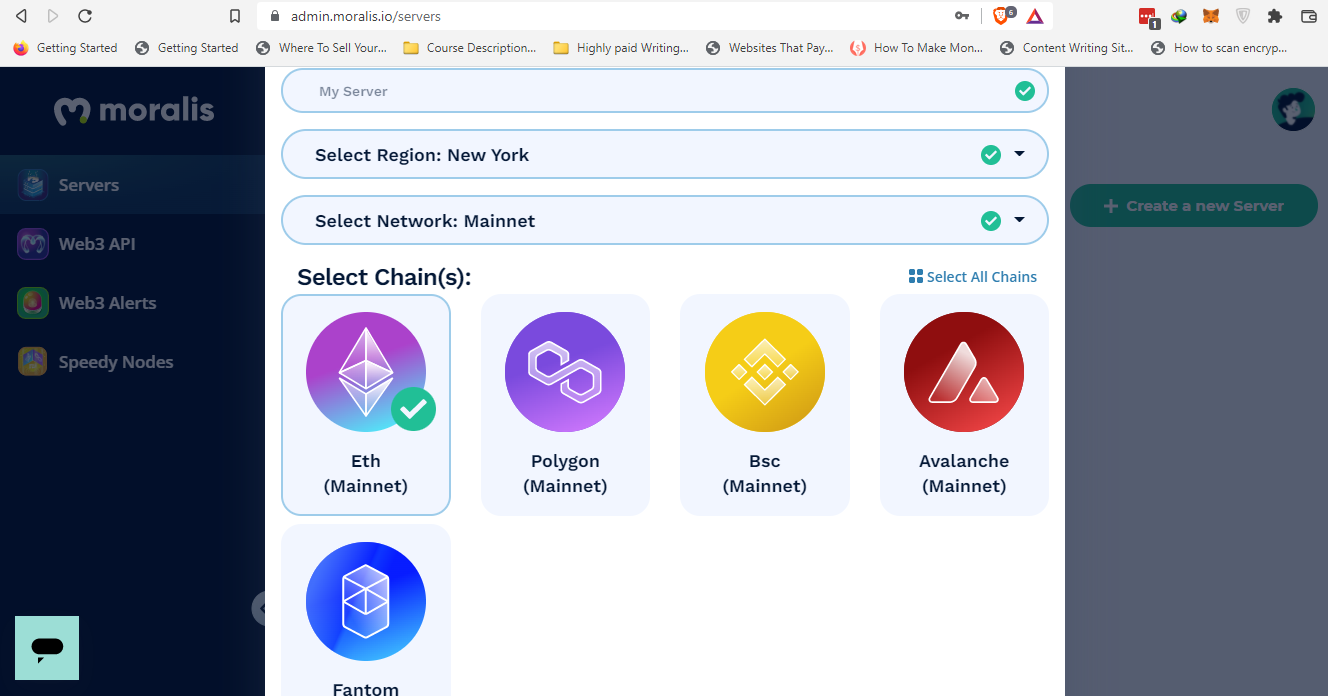
In a nutshell, developing a dApp with Ethereum utilizing Moralis is an easy three-step process.

**Step #1:** **Setup the Moralis Server** - Create a free account with Moralis. Login to your Moralis account. Setup the Moralis server by clicking the "+ Create a new Server” button on the top right of the Moralis dashboard after logging in.

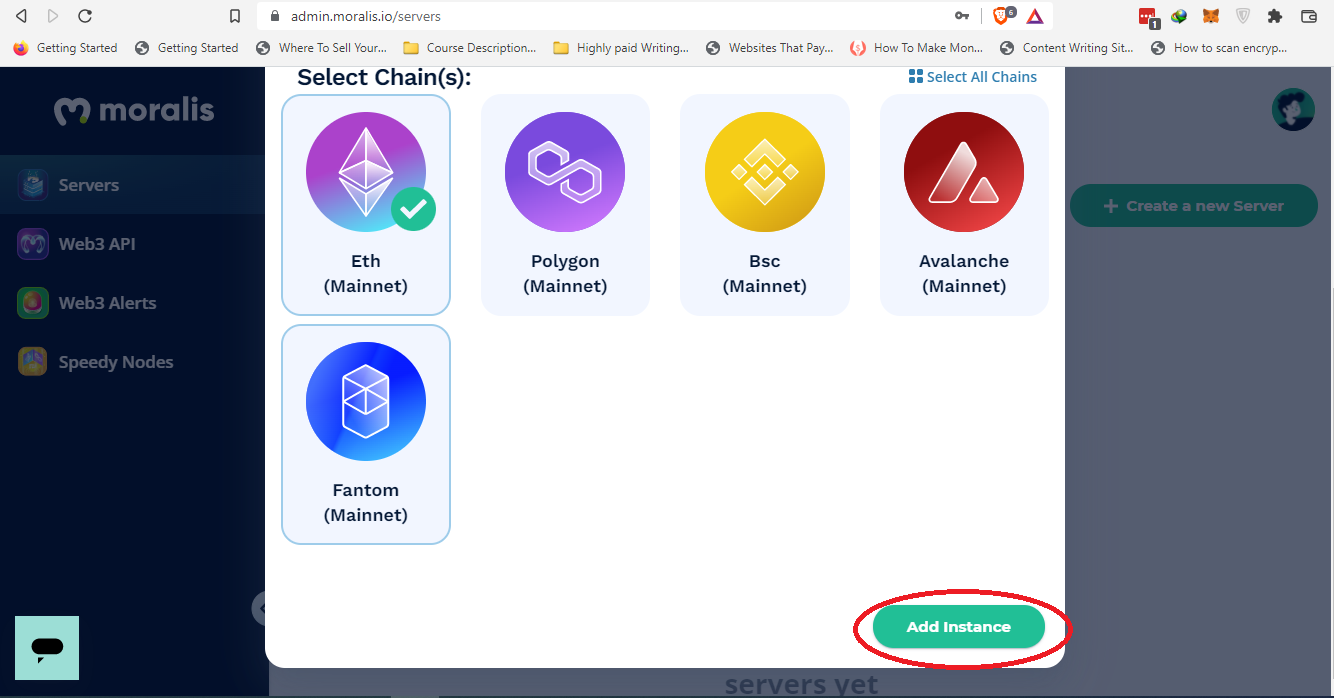


From the drop-down list, select the Mainnet. The three drop-down option menu asks you to specify where you want your dApp to live.



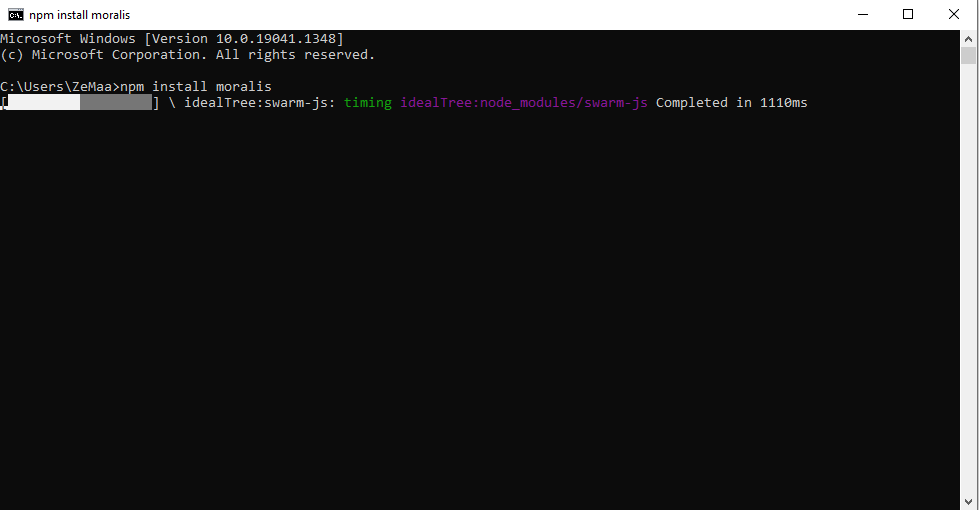
Fill in the short form to create the server in Moralis.

Select the blockchain network that you will use to create your dApp. In this case, we will choose Eth (Mainnet). You can opt for multiple networks since Moralis supports cross-chain compatibility. Once done, click the “Add Instance” button at the bottom right corner of the pop-up.

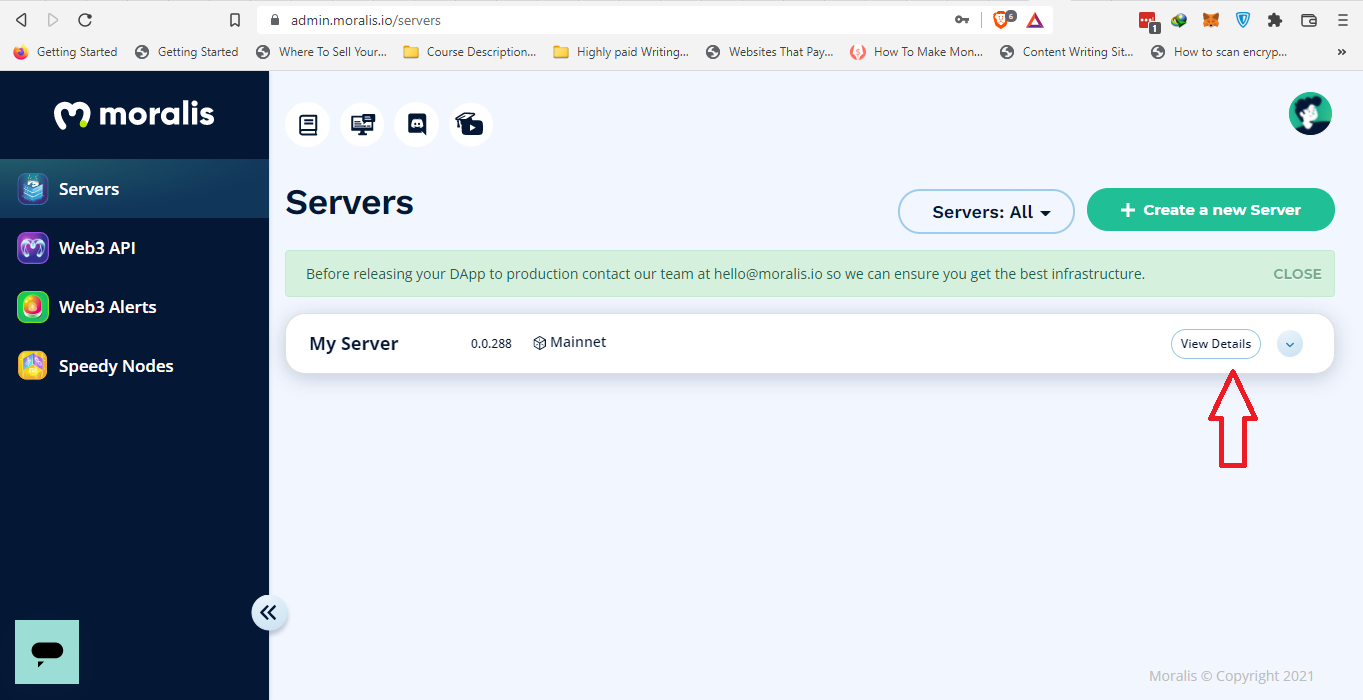


**Step#2: Install Moralis SDK** – Installing Moralis SDK is a piece of cake. There are many ways to install Moralis SDK. We will use npm (Node Package Manager) to install the Moralis SDK. Make sure that you have installed node.js. The command is as follows.

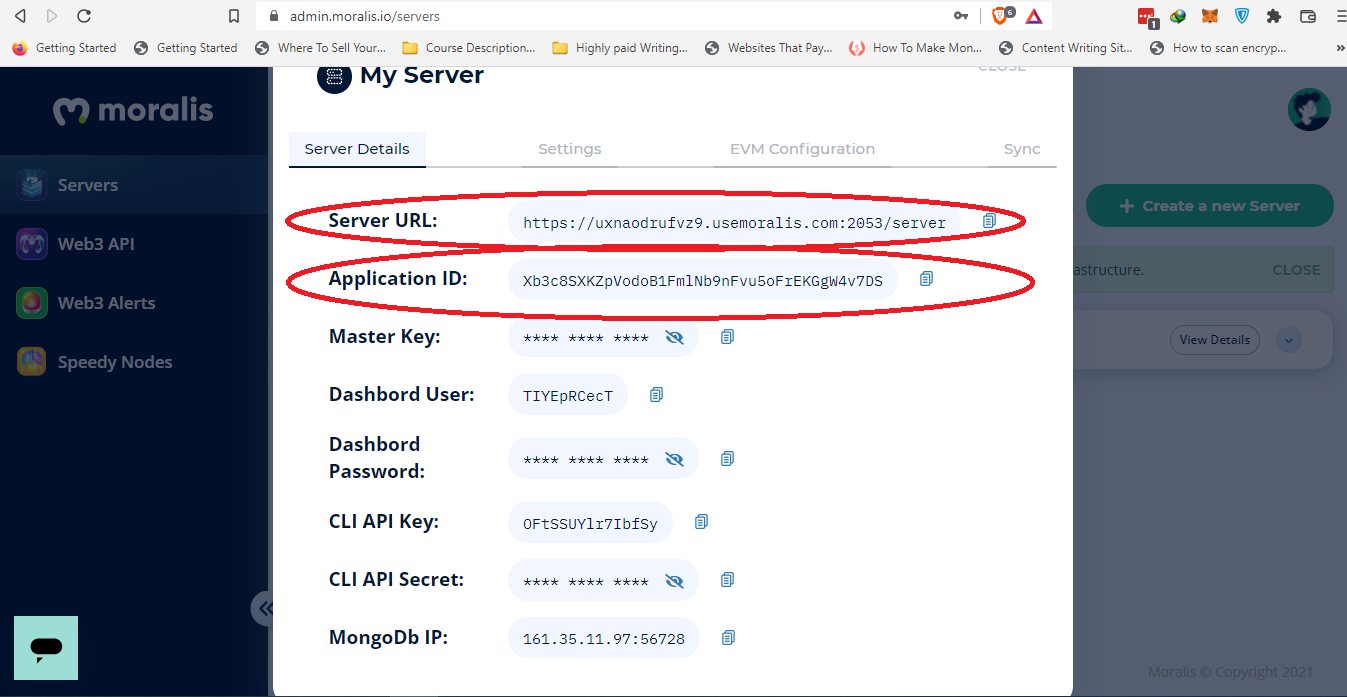
npm install moralis



**Step#3: Initialize Moralis** – The final step is to initialize Moralis. Doing this will enable us to use Moralis in code. To accomplish it, we will need the “app ID” and “server URL." Hit "View Details" on the Moralis dashboard.



The following pop-up will open, and you will get the "Server URL" and "Application ID."



The code to initialize the Moralis server is as follows,

1.Moralis.initialize (”YOUR\_APP\_ID”);

2.Moralis.serverURL = ’YOUR\_SERVER\_ID’

After plugging in the values of app ID and server

**1**.Moralis.initialize (”Xb3c8SXKZpVodoB1FmlNb9nFvu5oFrEKGgW4v7DS”);

**2**.Moralis.serverURL =’https://uxnaodrufvz9.usemoralis.com:2053/server’

Now, you can see that we have laid the foundation of a dApp.

**Moralis SDK Code Examples for Building Ethereum dApps**

After initializing [Moralis SDK](https://moralis.io/exploring-moralis-sdk-the-ultimate-web3-sdk/) after the Moralis server launches, and you start utilizing the SDK. As an example, if you want to allow a user to log in to a dApp account. You need to copy-paste two lines of code as shown in the following.

Moralis.Web3.authenticate().then(**function**(user){

console.log(user.get('ethAddress'))

})

The power of Moralis is that as soon as the user logs in to the dApp, the user information is readily available from the Moralis database server. You do not have to take the hassle of setting up your database server. As you quickly build the backend of your dApp with Moralis, the Angular Framework will take care of the frontend of your dApp.

Moralis SDK is a powerful toolbox that helps you build dApps quickly and easily. In the following, you see how easily you can perform many different kinds of operations.

**Getting transactions:**

//get mainnet transactions for the current user

**const** transactions = **await** Moralis.Web3API.account.getTransactions();

The “await” keyword will pause the program execution. Once the promise of fetching transactions of the user account completes, the code execution continues.

**Getting MATIC/BNB balance:**

//get mainnet native balance for the current user

**const** balance = **await** Moralis.Web3API.account.getNativeBalance();

In this code native balance of the dApp user is fetched and stored in the constant “balance.”

**Fetching ERC-20 balance:**

**const** balance = **await** Moralis.Web3API.account.getTokenBalance();

This code provides access to token balance.

**Getting NFT balance:**

**const** userEthNFTs = **await** Moralis.Web3API.account.getNFTs();

The above code explains some common programming scenarios with Moralis. You can devise any form of query, for instance, filtering, sorting, and more. You can create awesome dApps with Moralis, so make sure you check out the different functionalities of Moralis.

The most notable thing in the mentioned examples of the Moralis code is that if you have tried to gain the same functionality from scratch (without the Moralis' tools, components, and so on). It will take nearly a month, and we made it in a few minutes. It is the power of the Moralis SDK.

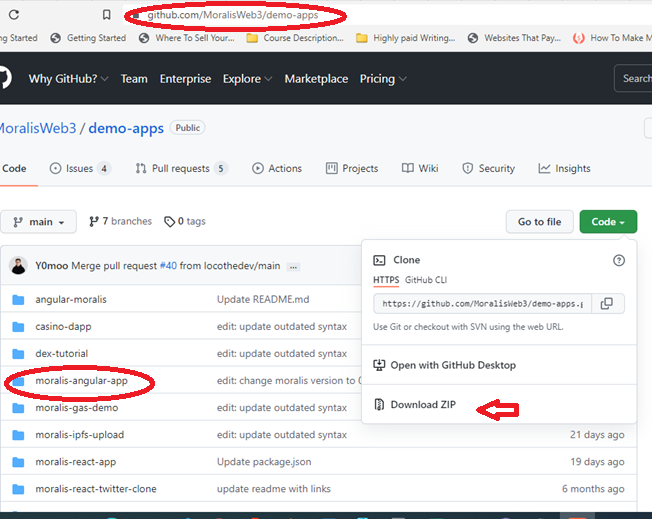
[](https://www.youtube.com/embed/lX9A6yQXZ_8?feature=oembed)

**Ethereum Angular Boilerplate Code to Integrate Angular with Moralis**

**[](https://www.youtube.com/embed/3WU8ple4zsM?feature=oembed)**

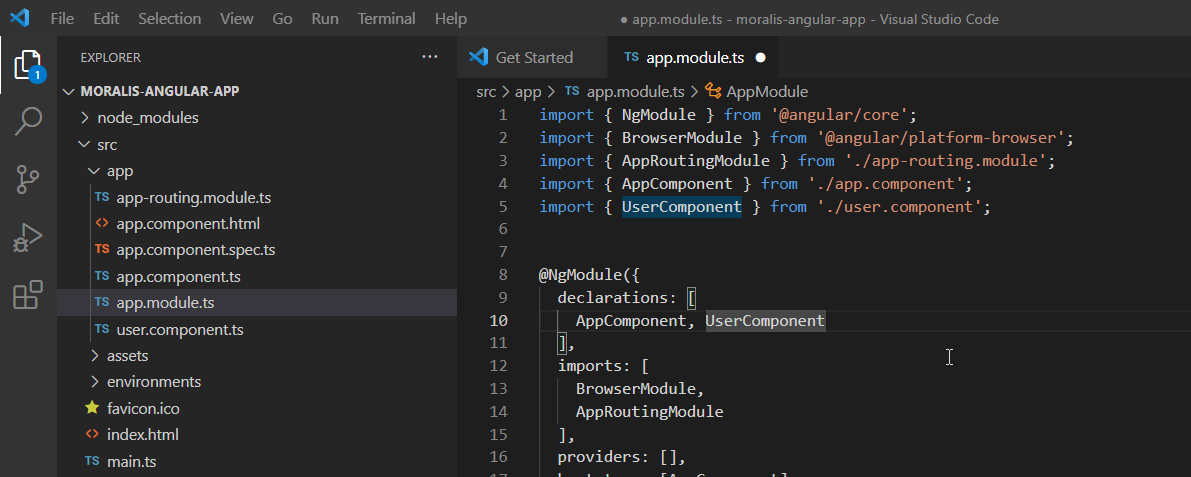
To use Angular framework with the Moralis, download the "demo-app" folder from GitHub. Make sure the demo-app folder contains the "moralis- angular-app" folder. The URL is below:

<https://github.com/MoralisWeb3/demo-apps>

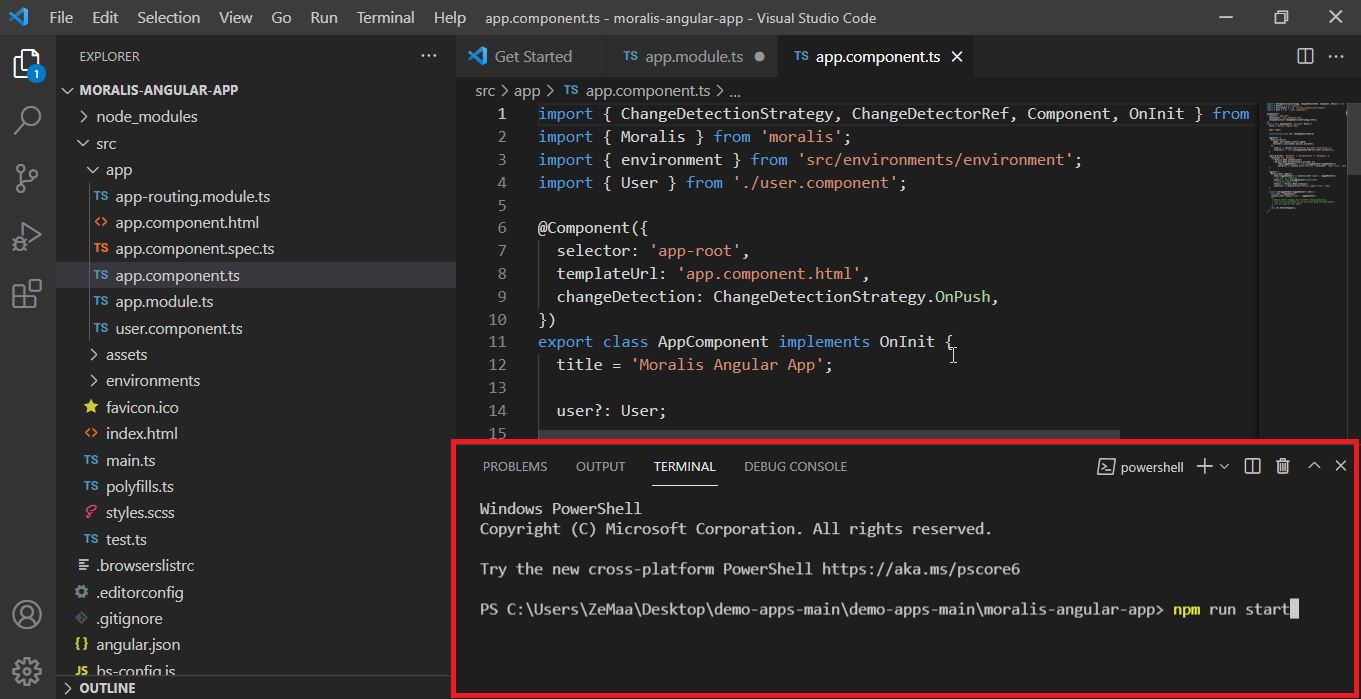


Download the demo-apps folder containing the moralis-angular-app. Open the moralis-angular-app folder into your code editor. We will be using Microsoft Visual Studio Code.

The folder contents will look something like the following.



Open the “Terminal” from the terminal tab in Visual Studio Code.



Run the following command in the command-line terminal,

npm run start

It will launch moralis-angular-app in production mode.

Add another terminal tab by clicking the "+" button on the right-hand bottom side in Visual Studio Code.

Run the following command

npm run demo

It will run molaris-angular-app in development mode.

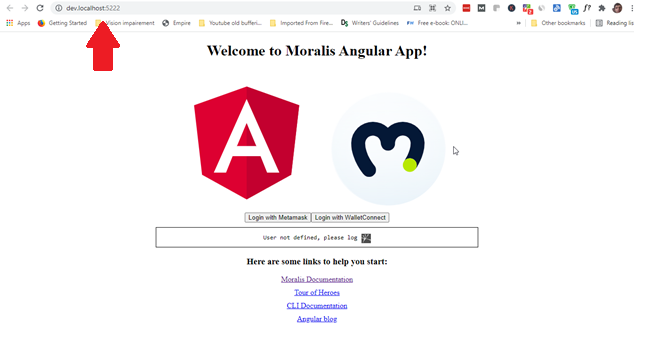
**Login with Angular in Development Mode**

It is the simplest way to login with Angular in MetaMask or WalletConnect. When you use the moralis-angular-app as previously described, you will find a URL in the terminal window of Visual Studio Code. Open that URL. It will be similar to the following.

http://dev.localhost:5222/

Here the “dev” in the URL shows that you are in development mode. The localhost means that the service runs on your local computer, and 5222 is the port number. You will see a similar MetaMask and WalletConnect login screen shown in the following. The only difference is that you will be in development mode. The production mode will open a similar webpage.

It will allow you to login with Angular in MetaMask or WalletConnect to connect your dApp to crypto wallets.



*Note: You may get an error saying, "You seem not be depending on "@angular/core" and "rxjs." This is an error" when you run the commands:*

npm run start

npm run demo

*If this happens, run the following command,*

*npm install*

*It should remove the error.*

**Login with Angular in Production Mode**

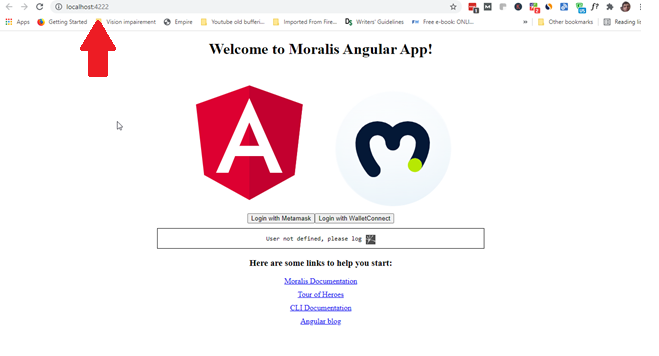
The moralis-angular-app provides you with the capability of utilizing login with Angular to MetaMask and WallectConect. The two are the most popular crypto wallets that connect to dApps to facilitate transactions. All the crypto earnings go into or out of these wallets from your dApp.

Once you execute the "npm run start" command, you need to scroll down the terminal and find a link similar to the following.

http://localhost:4222/

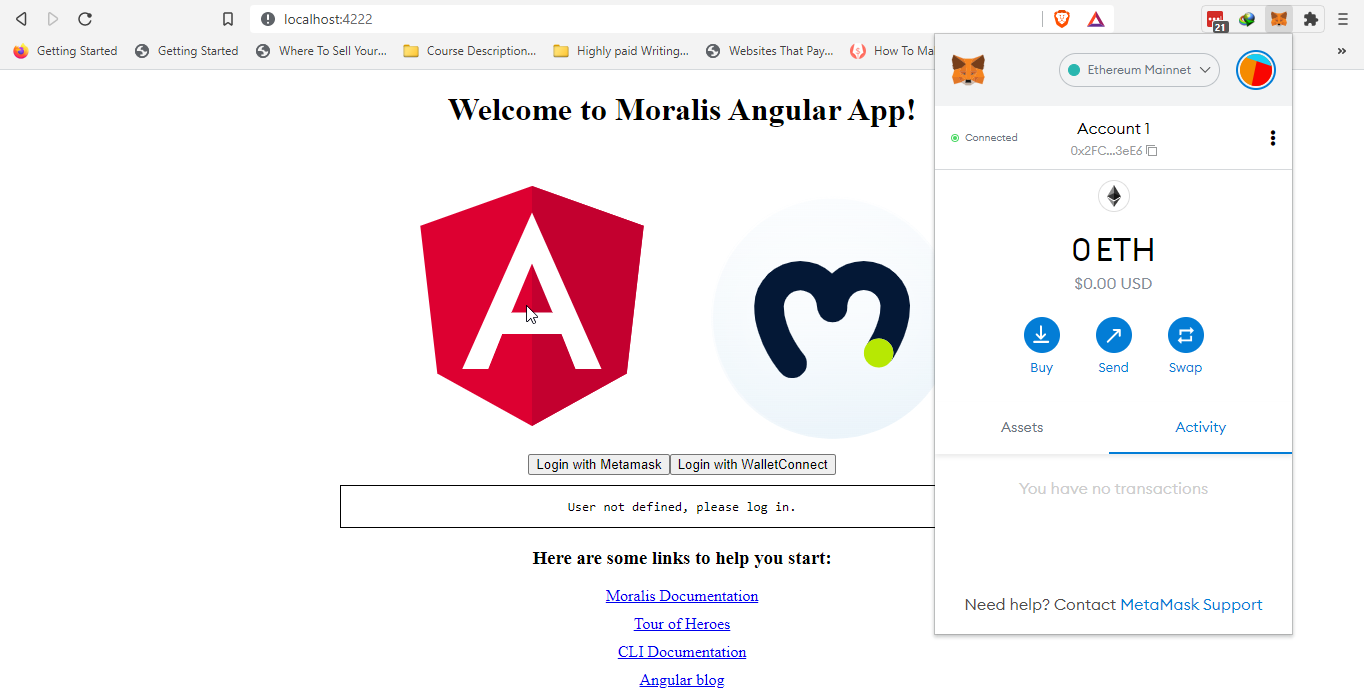
The link shows that the service runs on the local computer. And 4222 is the port number of the service.

You need to open that link, and you should find a page similar to the following.



When you see the above page. Click on “Login with MetaMask." You will log into your MetaMask account. Make sure that you already have the MetaMask browser extension installed in your web browser and you have a MetaMask account.

Similarly, when you click the “Login with WalletConnect," you will see a QR code pop-up. Scan the QR code with your mobile, and you will connect your dApp with WalletConnect.



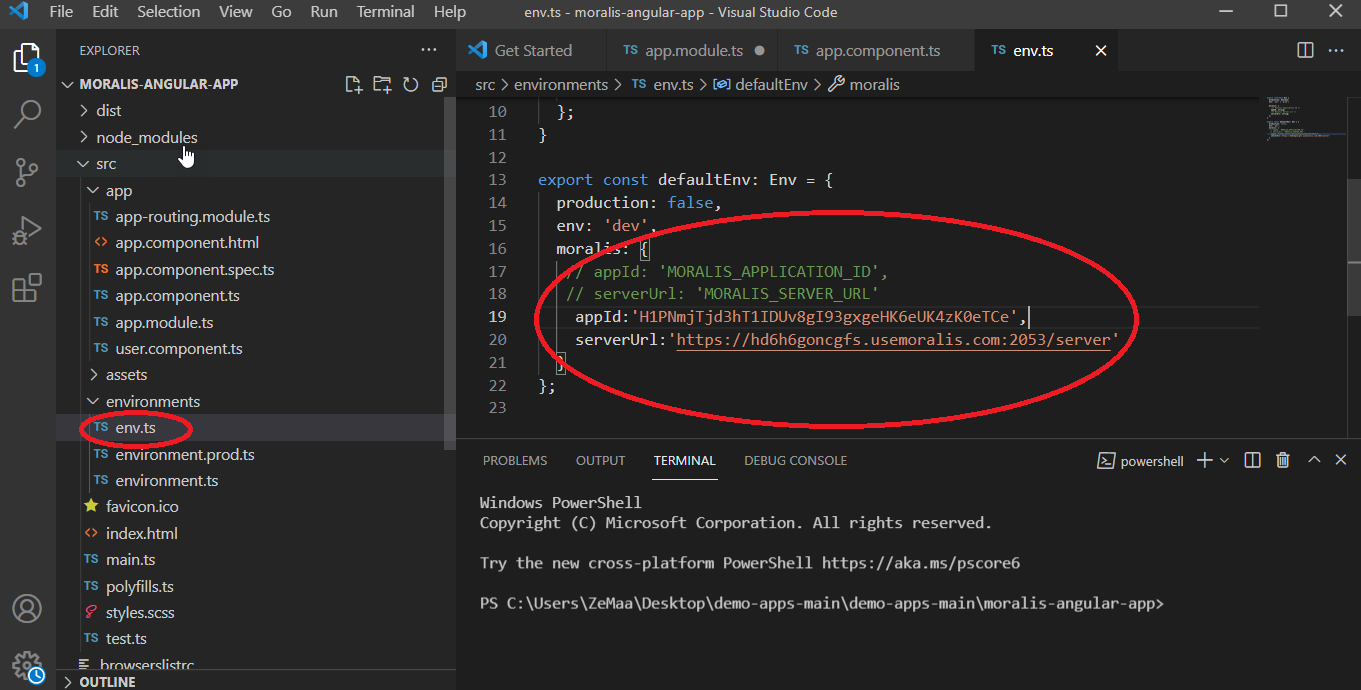
When you click on the Login with MetaMask button and Sign within the MetaMask extension, your dApp will connect with the MetaMask wallet.

You can similarly connect with WalletConnect by clicking the "Login with WalletConnect" button. It is the way your wallet works to make you login with Angular along with Moralis.

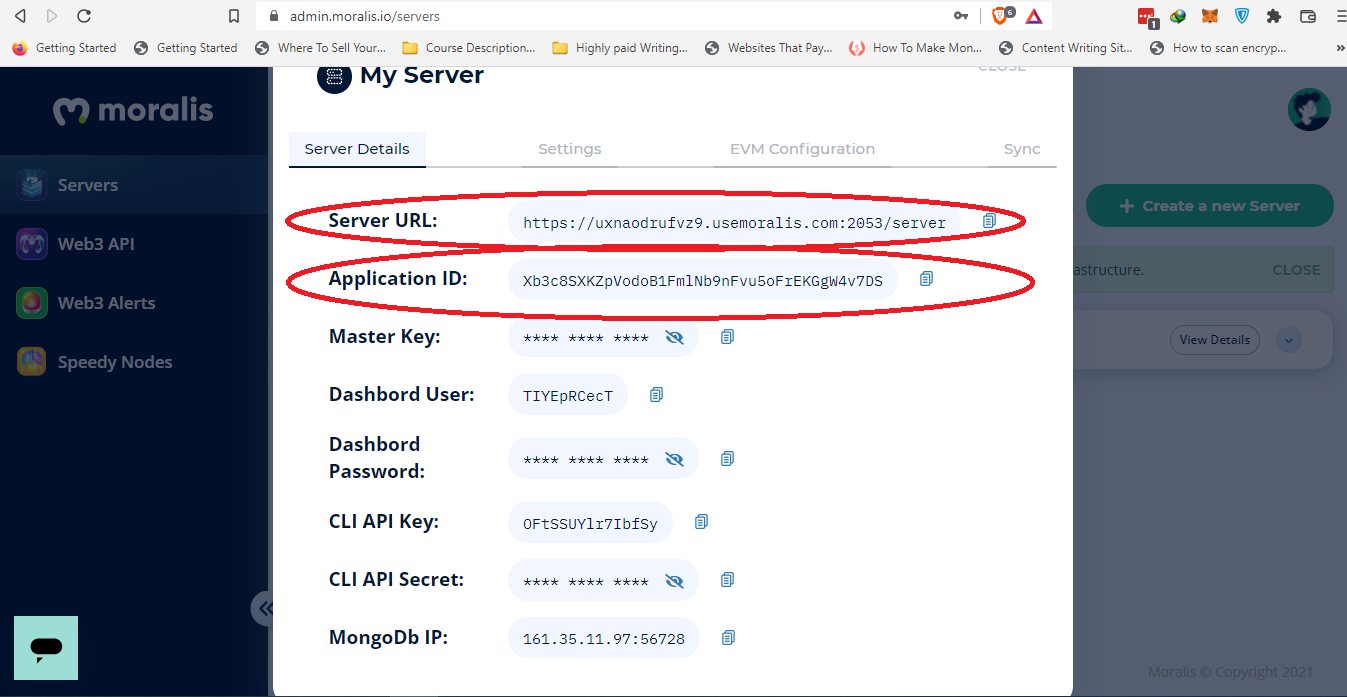
**Integrating Angular with Moralis to Build Ethereum dApps**

To integrate the Angular code with the Moralis server, head over to the env.ts in the "moralis-angular-app."

At the end of the code, comment out or give the values of "appId” and “serverUrl" as we did previously.

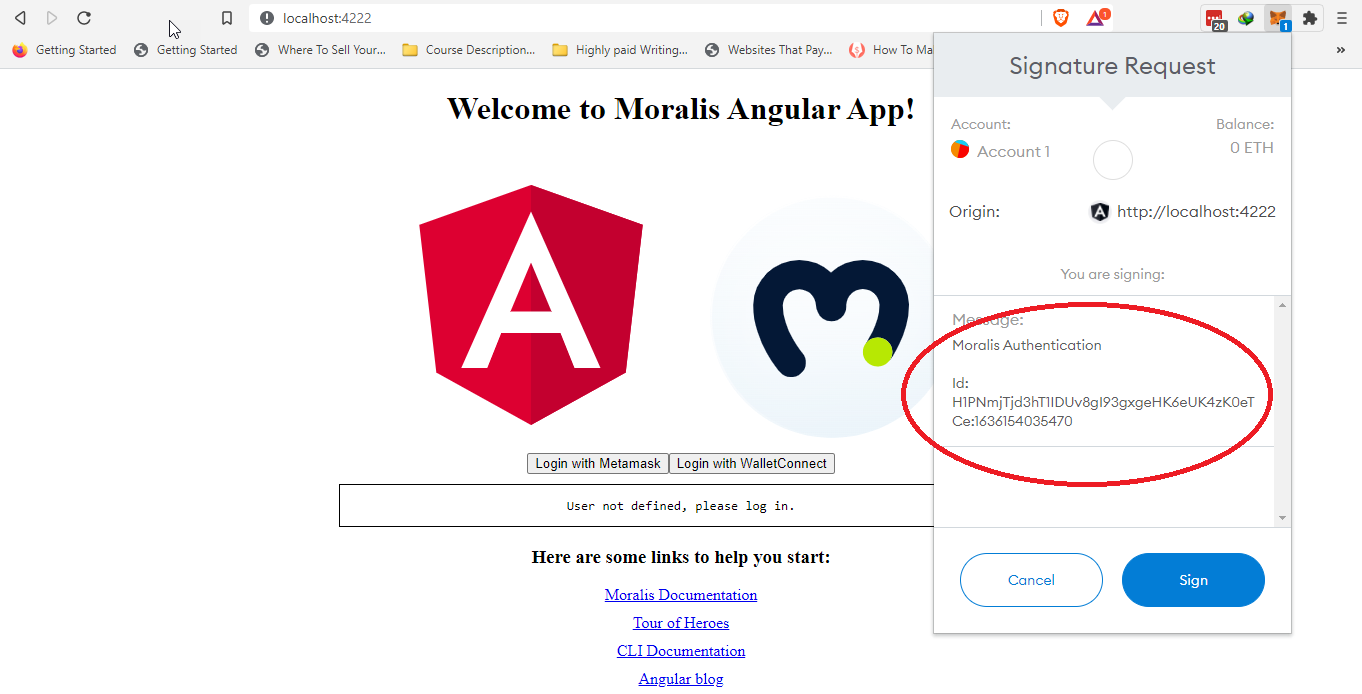


Now, the question arises where do you get the “appId” and the “serverUrl." It is the same procedure we did before, copy-paste the server ID and the app ID from the Moralis dashboard, head over to your moralis.io account and click on "View Details." You will see the following pop-up screen.



Copy-paste the “Application ID” to “appId:” and the “Server URL” to “serverurl” in the TypeScript code.

After integrating Angular code will Moralis, run the "moralis-angular-app" and browse the URL on the localhost as you did before. You will see the following web page as you log in to MetaMask.



Notice you get the Moralis Authentication Id. Now you are ready to create the frontend of your dApp in Angular with Moralis at your backend without the burden of building your server infrastructure. You save months of coding effort and expense for the same functionality. What is more interesting is that your dApp will also have integration with your MetaMask or WalletConnect wallets.

From here, you can let your creativity run wild and develop awesome dApps. For further development of your dApp, you can refer to the [official documentation of Moralis](https://docs.moralis.io/introduction/readme).

**Smart Contracts in Solidity to Build Ethereum dApps in Angular**



When you build Ethereum dApps, you must have an understanding of [the Solidity programming language](https://moralis.io/solidity-explained-what-is-solidity/). Solidity is the programming language in which [dApp builders build smart contracts](https://moralis.io/smart-contracts-explained-what-are-smart-contracts/).

Smart contracts are programs stored on the blockchain like Ethereum, and the execution of the smart contracts takes place when the right conditions become valid. Smart contracts can be thought of as the way you organize code, store data, and write all of your programming logic.

It is a high-level language to create smart contracts, and its creation has influences from JavaScript, C++, and Python. Smart contracts run on Ethereum Virtual Machine (EVM) that forms the Ethereum blockchain.

To create a smart contract, you need an IDE (Integrated Development Environment) called ["Remix](https://moralis.io/remix-explained-what-is-remix/)." You can download Remix to your computer, or you can use it in your browser and start coding from there in Solidity.

Solidity is not hard to learn, and it can be a powerful tool. If you are new to dApp development, you have to know Solidity very well.

**Conclusion**

There is no better solution yet in the blockchain market that allows you to set up your server with all the necessary services for your dApp to thrive. Moralis adds enormous value to your dApp in the form of ease of use and saves you loads of time and effort with many other benefits. You can focus on your dApp frontend coding.

Similarly, when Angular and Moralis integration occurs, it offers a powerful solution for your frontend. Make your dApp creation faster, and you enjoy all the other perks to build Ethereum dApps in Angular, such as one-page coding for the frontend and a remarkable user experience.

So, sign up with Moralis and jump-start your innovative dApp project. One more awesome thing about Moralis is that besides being free, it also allows you to grow your project as you intend to scale your business with time. Moralis also facilitate you to form your private Ethereum dApp infrastructure for your business as it grows.

*Meta Description:*

*Moralis and Angular are two technologies to build Ethereum dApps reducing months of effort and high costs into a few hours of work.*