**The Bored Ape Yacht Club**

[The Bored Ape Yacht Club](https://boredapeyachtclub.com/) is a collection of NFTs on the Ethereum Blockchain.  
The Bored Apes are an [ERC-721 token](https://ethereum.org/en/developers/docs/standards/tokens/erc-721/), controlled by the [0xbc4ca0eda7647a8ab7c2061c2e118a18a936f13d](https://etherscan.io/token/0xbc4ca0eda7647a8ab7c2061c2e118a18a936f13d) contract,

There are 10,000 Bored Apes (all controlled by the single contract). The script [bayc\_supply.py](https://risksalon-heliumlake.codio.io/bayc_supply.py) shows how you can get the “supply” of apes by calling the contract using the Python web3 library.

The Apes are created from a jumble of “attributes,” e.g. background, clothing, eyes etc. The [Bored Ape Gallery](https://boredapeyachtclub.com/#/gallery) makes it easy to filter on different attributes.

For most NFTs, all the “metadata” is stored off-chain. For the Bored Apes, you can access the metadata for an Ape by calling the “tokenURI” function, and providing the ID of the APE (i.e., a number between 0 and 9,999).

**IPFS**

The Bored Apes contract uses IPFS to store its metadata, and if you query the tokenURI(1) on the Bored Apes contract, you will receive the URI for the first ape in the collection. In this case:

ipfs://QmeSjSinHpPnmXmspMjwiXyN6zS4E9zccariGR3jxcaWtq/1

To access data stored on IPFS, you need either an IPFS node, or an IPFS gateway. For this assignment, you are welcome to use either.  
You’ll probably want to use a gateway, and there are several, including Infura, [Pinata](https://docs.pinata.cloud/retrieving-content) and [ipfs.io](https://docs.ipfs.io/concepts/ipfs-gateway/#gateway-providers).

So, for example if you wanted to access the metadata for the first ape, you can access it through

1. <https://ipfs.infura.io:5001/api/v0/cat?arg=QmeSjSinHpPnmXmspMjwiXyN6zS4E9zccariGR3jxcaWtq/1> (Note that Infura doesn’t allow GET requests, only POST, but you can easily connect using [requests.post](https://requests.readthedocs.io/en/latest/user/quickstart/" \l "make-a-request))
2. <https://gateway.pinata.cloud/ipfs/QmeSjSinHpPnmXmspMjwiXyN6zS4E9zccariGR3jxcaWtq/1>
3. <https://ipfs.io/ipfs/QmeSjSinHpPnmXmspMjwiXyN6zS4E9zccariGR3jxcaWtq/1>

**Assignment**

Using the template [get\_ape\_info.py](https://risksalon-heliumlake.codio.io/get_ape_info.py), write a function “get\_ape\_info(apeID),” that takes as input the ape ID (between 0 and 9,999) and outputs a python dict with the three fields “owner” and “image” and “eyes.” The owner field should hold the address of the (current) owner, and the image field should hold the URI of the image, and the eyes field should have the type of eyes of the ape.

Thus get\_ape\_info(1) should return something like:

'owner': "0x46EFbAedc92067E6d60E84ED6395099723252496",

'image': "ipfs://QmPbxeGcXhYQQNgsC6a36dDyYUcHgMLnGKnF8pVFmGsvqi",

'eyes': "Blue Beams"

In order to complete this assignment, you will need to

1. Use web3 to connect to an Ethereum node (to get the owner and tokenURI)
2. Pull image data from IPFS (this is *not* on the blockchain)

You can get feedback on your code by clicking “Run Test Cases”.  
Note that the test cases are designed to look for common errors and only run a subset of the tests performed by the grading script. So if the test cases pass you are not guaranteed to get a perfect score. On the other hand, any errors identified by the test cases would result in lost points if the code were to be submitted as-is.