We presently are minting an NFT that represents ownership of the AI model, and is linked to the smart contract so that the owner can prove their ownership of the model on the blockchain.

We want to use smart contracts to register and control access to our data science model. The smart contract should be programmed to require permission from the owner before allowing access to the model, and can also send notifications to the owner each time the contract is accessed.

We need to create a smart contract that allows investors to purchase tokens representing ownership in the AI model. These tokens can be minted as ERC-20. The smart contract should be programmed to distribute royalties to token holders whenever someone pays to use the model. The contract should be programmed to automatically calculate and distribute the royalties based on the number of tokens held by each investor and the amount of revenue generated by model usage.

In order to implement this, we need to have a way to track the usage of the model and the associated revenue, it can be done by through an oracle that tracks the usage and sends the information to the smart contract. We should use decentralized oracles (such as Chainlink) because they use a network of multiple independent nodes to provide a more robust solution.

The oracle would be responsible for monitoring the usage of the model and collecting the relevant data (e.g number of requests, revenue generated) and then sending that data to the smart contract.

The oracle architecture would be:

* Off-chain system: This is the external data source or API that the oracle is monitoring. In this case, it would be the system that the AI model is deployed on and the usage of the model is being tracked.
* Oracle contract: This is a smart contract deployed on the blockchain that acts as an intermediary between the off-chain system and the main contract. It contains the logic to communicate with the off-chain system, validate the data received, and then send it to the main contract.
* Main contract: This is the smart contract that is responsible for handling the investment and royalty distribution. It contains the logic to receive the data sent by the oracle contract and use it to calculate and distribute royalties to the investors.

A sample architecture for an oracle that tracks the usage and sends the information to the smart contract could look like this:

* Off-chain system: A server running the AI model and a database to store usage data.
* Oracle contract: A smart contract deployed on blockchain that communicates with the off-chain system to retrieve the usage data, validate it, and then sends it to the main contract.
* Main contract: A smart contract deployed on blockchain that receives the usage data sent by the oracle contract, calculates the royalties based on the number of tokens held by each investor and the revenue generated by the model, and then distributes the royalties to the investors.

We limit the number of investors by setting a maximum supply of tokens representing ownership in the AI model in the smart contract. For example, if we limit the number of investors to 20, we set the maximum supply of tokens to 20. Once all the tokens are sold, the smart contract will no longer allow any more purchases.

We should also include a clause in the smart contract that checks the number of investors, and only allows purchases if the number of investors is less than the maximum limit.

We should also set a restriction on the minimum investment amount to limit the number of investors, for example, instead of setting a fixed number of investors, We say that first 20 investors who invest more than 10 ETH are allowed to invest.