Quickstart

This quickstart covers two things:

1. Deploying an AI app to the Galadriel testnet
2. Calling your deployed AI app and viewing the results

Prerequisites

- A Galadriel testnetaccount
- . We recommend creating a new EVM account for development purposes.
- Sometestnet tokens
- on the account you are using.
- node
- andnpm
- installed on your machine.

Deploying a contract on Galadriel testnet

1 Testnet tokens

Get yourself some testnet tokens from thefaucet . 2 Clone repository

Clone the repo that contains Galadriel example contracts and the oracle implementation.

git clone https://github.com/galadriel-ai/contractscd contracts/contracts 3 Setup environment

Use thetemplate.env file to create a.env file:

Starting in repo root

cp template.env .env Then, modify the.env file:

- SetPRIVATE_KEY_CUSTOM
- to the private key of the account you want to use for deploying the contracts to Galadriel testnet.
- SetORACLE_ADDRESS
- to thecurrent testnet oracle address
- provided by Galadriel team:0xACB8a1fcC06f1a199C1782414E39BdB4A8238e69
- . 4 Install dependencies

Install the dependencies using npm:

In repo root -> /contracts

npm install 5 Deploy contract to testnet

Thequickstart contract can be deployed with asimple script :

npm run deployQuickstart The output of the script will show the deployed contract address. Store this and export it in terminal to call the contract later:

Replace with your own contract address

export QUICKSTART_CONTRACT_ADDRESS = 0x.. .

Calling your contract

Prerequisites:

- You’ve deployed the quickstart contract in the previous step, and stored the contract address.

Execute the following command to run a script that calls the deployed contract:

npm run callQuickstart The script will interactively ask for the input (DALL-E prompt: what image should be generated) and then call the contract with the input. The output, when ready, will be printed to the console.

If this step fails, make sure you have set theQUICKSTART_CONTRACT_ADDRESS environment variable to the address of your deployed contract. You’re done! You’ve successfully created and called your own on-chain AI app.

What’s next?

- Continue developing your own contracts. See the contracts readme
- and theSolidity reference
- for more.
- Readhow Galadriel works
- .
- Exploreuse cases
- for inspiration and examples of full dApps including a frontend.