Back-End Code Challenge

Write a backe-end service that will connect to the KOVAN network and “communicate” (read) with GoodGhosting’s DEMO game on Kovan network.

Requirements: The backend service/repo should have the following features / endpoints:

* Backend service to implement a public endpoint that returns the information of a player in the game. The endpoint should receive the player address as a parameter and return the data of the player that is available in the smart contract. If the player address does not exist (it’s not a player), the endpoint should return a different type of response to alert the client consuming it (up to you to decide if that’s done via http status, error code, error message, etc).
  + At the smart contract level, there’s a Player Struct which stores some information about the player. Please check the smart contract on block explorer to understand what information is returned.
  + If a given address is not a player in the game, the property “addr” in the player’s struct will return a ‘0x0…’ (zero) address
* Backend service to implement a public endpoint that returns the current segment of the game. HOWEVER, this calculation must be done inside the backend service by using (reading) the variables from the smart contract required for such calculation. The data you need from the smart contract is when the game started (variable “firstSegmentStart” and the duration of each round (variable “segmentLength”).
  + TIP: the smart contract implements a function that performs this calculation, but for the purposes of this code challenge, we’d like you to “duplicate” this logic in the backend service itself instead of simply calling the smart contract’s function.
* Write a unit tests for the endpoints implemented above
* Readme file w/ details on how to setup and run the service locally, unit tests, and any additional documentation you see fit

Expected Tech Stack

* Typescript
* Web3.js or ethers.js
* Unit test framework of your choice (at GoodGhosting we use Jest)
* Node.js services (like express.js, fastify.js, etc). At GoodGhosting we use NestJS, but feel free to use the one you’re most comfortable with.

What you’ll need:

* Address of GoodGhosting Demo Smart Contract on Kovan: *0xc69a569405EAE312Ca13C2eD85a256FbE4992A35*
  + contract available here: <https://kovan.etherscan.io/address/0xc69a569405EAE312Ca13C2eD85a256FbE4992A35>)
  + The name of the contract is “GoodGhosting”
* GoodGhosting Smart Contract ABI
  + File “ABI-GoodGhostingWhitelisted.js” attached

You can have a sneak peek of how our DEMO game works on KOVAN at this url:

<https://goodghosting.com/#/pools/99999>

If you want to use the Game Demo on Kovan, you’ll need:

* Some Kovan test ETH to pay for gas.
  + Get some at Kovan faucet: <https://app.mycrypto.com/faucet>
* At least 1 DAI (Kovan) to make the deposits.
  + Get some at DAI (kovan) faucet: <https://testnet.aave.com/faucet/DAI-0xff795577d9ac8bd7d90ee22b6c1703490b6512fd0x88757f2f99175387ab4c6a4b3067c77a695b0349>

For reference: GoodGhosting Game Mechanics + Docs

<https://docs.goodghosting.com/goodghosting/game-mechanics-and-technical>