Guest Book

Our Guest Book example is a simple app composed by two main components:

1. A smart contract that stores messages from users, allowing to attach money to them.
2. A simple web-based frontend that displays the last 10 messages posted.

Obtaining the Guest book Example

You have two options to start the Guest book Example.

1. You can use the app throughGitHub Codespaces
2. , which will open a web-based interactive environment.
3. Clone the repository locally and use it from your computer.

Codespaces Clone locally   https://github.com/near-examples/guest-book-examples

Structure of the Example

The example is divided in two main components:

1. The smart contract, available in two flavors: Rust and JavaScript

2. The frontend, that interacts with an already deployed contract.

3.   JavaScript

4.   Rust

┌── sandbox-ts # sandbox testing | ├── src | | └── main.ava.ts | ├── ava.config.cjs | └── package.json ├── src # contract's code | ├── contract.ts | └── model.ts ├── package.json # package manager ├── README.md └── tsconfig.json # test script ┌── tests # workspaces testing | ├── workspaces.rs ├── src # contract's code | └── lib.rs ├── Cargo.toml # package manager ├── README.md └── rust-toolchain.toml

Frontend

The guest book example includes a frontend that interacts with an already deployed smart contract, allowing user to sign a message.

Running the Frontend

To start the frontend you will need to install the dependencies and start the server.

cd frontend yarn yarn start Go ahead and login with your NEAR account. If you don't have one, you will be able to create one in the moment. Once logged in, you will be able to sign a message in the guest book. You can further send some money alongside your message. If you attach more than 0.01Ⓝ then your message will be marked as "premium".

Understanding the Frontend

The frontend is composed by a single HTML file (/index.html) and uses REACT. Check/App.js and/index.js to understand how components are displayed in the screen.

You will notice in/src/App.jsx the following code:

- JavaScript

frontend/src/App.jsx loading ... [See full example on GitHub](https://github.com) It setups the necessary variables and starts the app.

Smart Contract

The contract presents 3 methods:add_message ,get_message andtotal_messages .

- JavaScript
- Rust

contract-ts/src/contract.ts loading ... [See full example on GitHub](https://github.com)contract-rs/src/lib.rs loading ... [See full example on GitHub](https://github.com)

Testing the Contract

The contract readily includes a set of unit and sandbox testing to validate its functionality. To execute the tests, run the following commands:

- JavaScript
- Rust

cd contract-ts yarn yarn test cd contract-rs cargo test tip The integration tests use a sandbox to create NEAR users and simulate interactions with the contract.

Deploying the Contract to the NEAR network

In order to deploy the contract you will need to create a NEAR account .

- JavaScript
- Rust

Optional - create an account

near create-account --useFaucet

Deploy the contract

cd contract-ts yarn build near deploy ./build/guestbook.wasm

Optional - create an account

near create-account --useFaucet

Deploy the contract

cd contract-rs cargo near build near deploy ./target/wasm32-unknown-unknown/release/guestbook.wasm tip To interact with your contract from the frontend , simply replace the variable CONTRACT_NAME in the index.js file.

CLI: Interacting with the Contract

To interact with the contract through the console, you can use the following commands

Get messages with optional arguments for pagination

near view guestbook.near-examples.testnet get_messages --args='{"from_index": "0","limit": "10"}'

Get total number of messages

near view guestbook.near-examples.testnet total_messages

Add a message

Replace with your account ID

Required a text

Optional deposit to make the message premium

near call guestbook.near-examples.testnet add_message '{"text":"Hello Near"}' --accountId --deposit 0.1 tip If you're using your own account, replace guestbook.near-examples.testnet with your accountId .

Moving Forward

A nice way to learn is by trying to expand a contract. You can modify the guestbook example to incorporate a feature where users can give likes to messages. Additionally, implement a method to toggle the like. Last updatedonMar 25, 2024 bymatiasbenary Was this page helpful? Yes No