README¶

PEER-AI¶

- About - Peerism API built with Truffle, Node.js, Express.js, Mongoose, MongoDB, Solidity, and Ganache CLI (TestRPC).
- Instructions - Setup instructions (for macOS) are provided in theQuick Start Guide
- shown below.
- Usage Capabilities - Use cURL to simulate a HTTP POST request to a Peerism API endpoint (instead of sending a request from the Peerism React Native ÐApp https://github.com/peerism/peer.ai). Alternatively allows triggering Middleware functions individually. API routes use an Express.js Middleware Chain that allow a request to compile the Peerism Smart Contract using Solidity Compiler (Solc) and Ether Pudding, then deploy it to the Ethereum TestRPC test network blockchain and responds with the contract address.

Table of Contents¶

- Quick Start Guide
- Log
- FAQ
- TODO
- References

Quick Start Guide

¶

- Terminal Tab #1 - Install dependencies including Ganache CLI (TestRPC from the Truffle Suite)
- nvm install v9.3.0;
- nvm use;
- nvm use v9.3.0;
- yarn install;
- npm install -g ganache-cli
- Terminal Tab #1 - Run MongoDB Server in a separate Terminal tab
- mongod
- Terminal Tab #2 - Run Ethereum Client using Ganache CLI (TestRPC).

rm -rf ./db; mkdir -p db/chaindb; ganache-cli --account '0x0000000000000000000000000000000000000000000000000000000000000001, 10002471238800000000000' \ --account '0x0000000000000000000000000000000000000000000000000000000000000002, 10004471238800000000000' \ --unlock '0x0000000000000000000000000000000000000000000000000000000000000001' \ --unlock '0x0000000000000000000000000000000000000000000000000000000000000002' \ --unlock '0x7e5f4552091a69125d5dfcb7b8c2659029395bdf' \ --unlock '0x2b5ad5c4795c026514f8317c7a215e218dccd6cf' \ --blocktime 0 \ --deterministic true \ --port 8545 \ --hostname localhost \ --gasPrice 20000000000 \ --gasLimit 0x8000000 \ --debug true \ --mem true \ --networkId 1337 \ --db './db/chaindb' * Terminal Tab #3 - Compile and Deploy Smart Contracts to TestRPC blockchain * Compile Smart Contracts * * Option 1: Generates build/contracts/Peerism.sol.js * * node lib/compileContract.js Peerism * * Option 2: Genertes build/contracts/Peerism.json (DEPRECATED) * * truffle compile --compile-all; * Deploy Smart Contracts * * Option 1: Deploy without Truffle * * Modify Bitcore dependency before running the next command to avoid errorError: More than one instance of bitcore-lib found. Please make sure to require bitcore-lib and check that submodules do not also include their own bitcore-lib dependency. * * , as described here: https://github.com/bitpay/bitcore/issues/1454, by opening node_modules/bitcore-mnemonic/node_modules/bitcore-lib/index.js and commented out the following lines of code to avoid an error. * * bitcore.versionGuard = function(version) { * * // if (version !== undefined) { * * // var message = 'More than one instance of bitcore-lib found. ' + * * // 'Please make sure to require bitcore-lib and check that submodules do' + * * // ' not also include their own bitcore-lib dependency.'; * * // throw new Error(message); * * // } * * }; * node lib/deployContract.js Peerism * * Option 2: Deploy with Truffle (DEPRECATED) * * truffle migrate --reset --network development; * * Note: Watch the deployment transactions being send to the blockchain in Terminal Tab #2 * Terminal Tab #3 - Run Tests * truffle test; * Terminal Tab #4 - Drop the server. Run server, then try cURL requests * yarn run drop; yarn run dev; * Terminal Tab #4 - Send request to server and receive response for authentication and authorisation to access specific API endpoints. * cURL * * Register with email/password. JWT provided in response (i.e.{"token":"xyz"} * *)curl -v -X POST http://localhost:7000/users/auth/register -d "email=luke@schoen.com&password=123456&name=Luke" -H "Content-Type: application/x-www-form-urlencoded" * * curl -v -X POST http://localhost:7000/users/auth/register -d '{"email":"gavin@wood.com", "password":"123456", "name":"Gavin"}' -H "Content-Type: application/json" * * Sign in with email/password. JWT provided in response (i.e.{"token":"xyz"} * *)curl -v -X POST http://localhost:7000/users/auth -d "email=luke@schoen.com&password=123456" -H "Content-Type: application/x-www-form-urlencoded" * * curl -v -X POST http://localhost:7000/users/auth -d '{"email":"gavin@wood.com", "password":"123456"}' -H "Content-Type: application/json" * * Access a restricted endpoint by providing JWTcurl -v -X GET http://localhost:7000/users -H "Content-Type: application/json" -H "Authorization: Bearer " * * Create user by providing JWTcurl -v -X POST http://localhost:7000/users/create --data '[{"email":"test@fake.com", "name":"Test"}]' -H "Content-Type: application/json" -H "Authorization: JWT " * * curl -v -X POST http://localhost:7000/users/create -d "email=test2@fake.com&name=Test2" -H "Content-Type: application/x-www-form-urlencoded" -H "Authorization: JWT " * Terminal Tab #4 - Send request to server with Smart Contract Name to be Compiled and Deployed to the Ethereum TestRPC and receive response with the Contract Address. * cURL * curl -v -X POST http://localhost:7000/contracts/generate -d '{"contractName":"Peerism"}' -H "Content-Type: application/json" * Terminal Tab #4 - Experiment in REPL * Use Truffle Console * * Run Truffle Console * * truffle console --network development; * * Run commands * * web3 * * web3.currentProvider * * web3.eth.getBalance('0x7e5f4552091a69125d5dfcb7b8c2659029395bdf') * Attach to EthereumJS TestRPC using Go Ethereum (Geth) * * Install Geth * * Start Geth JavaScript console * geth attach rpc:http://localhost:8545 * * Run commands * web3 * web3.currentProvider * web3.eth.getBalance('0x7e5f4552091a69125d5dfcb7b8c2659029395bdf') * eth.accounts * Optional: Try to perform RPC calls to Ganache TestRPC using cURL. https://github.com/trufflesuite/ganache-cli/issues/383 * Terminal Tab #5 - Run Tests on port 7111 * yarn run drop; yarn run test-watch * Terminal Tab #1 - Drop the database. Seed the database * yarn run drop; * yarn run seed;

Log

¶

- Initial setupgit init; touch README.md; touch .gitignore;
- code .;
- Add boilerplate contents to .gitignore for Node.js
- Setup API
- yarn init -y;
- yarn add express body-parser;
- yarn add nodemon --dev;
- touch server.js;
- Add boilerplate contents to server.js
- Add "dev" in "scripts" section of package.json
- Add Mongoose
- yarn add mongoose;
- mkdir models; touch models/init.js;
- touch models/User.js;
- touch models/seeds.js;
- touch models/drop.js
- Create Models for Mongoose
- Add boilerplate contents to models
- Add scripts to package.json
- Run MongoDB Server
- mongod
- MongoDB Client
- mongo
- show dbs
- use peerai
- show collections
- db.users.find({})
- db.skills.find({})
- Create routes
- mkdir routes
- Modify server.js. Add routes/users.js
- Add authentication withPassport, Passport-Local, and Passport-Local-Mongoose
- :
- yarn add passport passport-local passport-local-mongoose
- Rename Person and people to User and users
- Add User Registration route
- Add User Sign in route
- Add JWT library to return a token instead of a useryarn add jsonwebtoken;
- Add Passport JWT libraryyarn add passport-jwt
- Add restricted endpoint that requires valid JWT to access
- Add Controllers https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/routes
- Add Route Testsyarn add mocha chai chai-http --dev;

- mkdir -p test/routes;
- touch test/routes/users_test.js;
- Add Model Testsmkdir test/models;
- touch test/models/users_test.js
- Add Dotenv library to use different database in development and testingyarn add lodash;
- yarn add dotenv --dev;
- touch .sample-env;
- echo 'NODE_ENV=development' >> ./.sample-env;
- Add Ethereum dependencies including TestRPCyarn add web3@0.19 ethereumjs-util@4.4 ethereumjs-tx@1.3 eth-lightwallet@2.5;
- yarn add ethereumjs-testrpc --dev;
- yarn add solc ether-pudding --dev;
- yarn add truffle-artifactor --dev;
- References
-
    - https://medium.com/@codetractio/try-out-ethereum-using-only-nodejs-and-npm-eabaaaf97c80
-
    - Smart Contracts without Truffle - https://medium.com/@doart3/ethereum-dapps-without-truffle-compile-deploy-use-it-e6daeefcf919
-
    - EthereumJS Util - Library for cryptographic hashes for Ethereum addresses - https://github.com/ethereumjs/ethereumjs-util
-
    - EthereumJS Tx - library to create, edit, and sign Ethereum transactions - https://github.com/ethereumjs/ethereumjs-tx
-
    - EthereumJS LightWallet - https://github.com/ConsenSys/eth-lightwallet
-
    - Solc - Compile Solidity Contract - https://www.npmjs.com/package/solc
-
    - Ether Pudding - Manage Solidity Contracts and Packages - https://www.npmjs.com/package/ether-pudding
-
    - Truffle Artifactor - replaces Ether Pudding - https://github.com/trufflesuite/truffle-artifactor
-
    - Reading from JSON files - https://www.codementor.io/codementorteam/how-to-use-json-files-in-node-js-85hndqt32
- Problem: Tried to manually compile using Solc withnode lib/compileContract.js ConvertLib
- , which generates ConvertLib.solc.js in build/contracts. However it does not compile MetaCoin.sol, as it returns error1:27: ParserError: Source "ConvertLib.sol" not found: File not supplied initially.\n ... import "./ConvertLib.sol"
- .
- Solution: Use Truffle to compile Solidity contracts withtruffle compile --compile-all
- Run shell script in new Terminal tab (copy from https://github.com/ltfschoen/solidity_test/blob/master/testrpc.sh)
- rm -rf ./db;
- mkdir db && mkdir db/chaindb;
- cd ~/code/blockchain/solidity_test; testrpc --account '0x0000000000000000000000000000000000000000000000000000000000000001, 10002471238800000000000' \
- --account '0x0000000000000000000000000000000000000000000000000000000000000002, 10004471238800000000000' \
- --account '0x0000000000000000000000000000000000000000000000000000000000000003, 10004471238800000000000' \
- --account '0x0000000000000000000000000000000000000000000000000000000000000004, 10004471238800000000000' \
- --account '0x0000000000000000000000000000000000000000000000000000000000000005, 10004471238800000000000' \
- --account '0x0000000000000000000000000000000000000000000000000000000000000006, 10004471238800000000000' \
- --account '0x0000000000000000000000000000000000000000000000000000000000000007, 10004471238800000000000' \
- --unlock '0x0000000000000000000000000000000000000000000000000000000000000001' \
- --unlock '0x0000000000000000000000000000000000000000000000000000000000000002' \
- --unlock '0x0000000000000000000000000000000000000000000000000000000000000003' \
- --unlock '0x0000000000000000000000000000000000000000000000000000000000000004' \
- --unlock '0x0000000000000000000000000000000000000000000000000000000000000005' \
- --unlock '0x0000000000000000000000000000000000000000000000000000000000000006' \
- --unlock '0x0000000000000000000000000000000000000000000000000000000000000007' \
- --unlock '0x7e5f4552091a69125d5dfcb7b8c2659029395bdf' \
- --unlock '0x2b5ad5c4795c026514f8317c7a215e218dccd6cf' \
- --blocktime 0 \
- --deterministic true \
- --port 8545 \
- --hostname localhost \
- --gasPrice 20000000000 \
- --gasLimit 1000000 \
- --debug true \
- --mem true \
- --db './db/chaindb'
- Install Trufflenpm install -g truffle;
- truffle init;
- Run Truffle Unbox in separate directory to get template Metacoin example and move relevant boilerplate contracts and tests into the the root folder
- Update package.json tests script to run tests for Smart Contracts and API tests:"test": "truffle test; NODE_ENV=testing mocha --recursive test//*_test.js",
- Remove truffle-config.js and add the following to truffle.js:module.exports = {
- // http://truffleframework.com/docs/advanced/configuration
- networks: {
- development: {
- host: "localhost",
- port: 8545,
- network_id: "*" // Match any network id
- }
- }
- };
- Add ethpm.json for EthPM Package Management{
- "package_name": "truffle-box-peerism-api-node-express",
- "version": "0.0.1",
- "description": "Truffle Box of Peerism API built with Truffle, Node.js, Express.js,
- Solidity, Ether Pudding, and Ethereum TestRPC",
- "authors": [
- "Luke Schoen ltfschoen@gmail.com"
-],
- "keywords": [
- "ethereum",
- "express.js",
- "node.js",
- "middleware",
- "api"
-],
- "license": "MIT"
- }
- References:
-
    - http://truffleframework.com/docs/getting_started/packages-ethpm
- Open node_modules/bitcore-mnemonic/node_modules/bitcore-lib/index.js and commented out the following lines of code to avoid an error.
- bitcore.versionGuard = function(version) {
- // if (version !== undefined) {
- // var message = 'More than one instance of bitcore-lib found. ' +
- // 'Please make sure to require bitcore-lib and check that submodules do' +
- // ' not also include their own bitcore-lib dependency.';
- // throw new Error(message);
- // }
- };
- Run Truffle Console experimentation
- truffle console --network development;
- Build script for Smart Contract (generates .sol.js file in build/contracts/)

- mkdir lib;
- node lib/compileContract.js Peerism
- Alternatively compile with Truffle
- Deployment script for Smart Contract
- Reference: https://medium.com/@codetractio/try-out-ethereum-using-only-nodejs-and-npm-eabaaaf97c80touch lib/deployContract.js;
- node lib/deployContract.js Peerism;
- References:* http://truffleframework.com/docs/getting_started/contracts
-
    - Gas Limits - https://bitcoin.stackexchange.com/questions/39132/what-is-gas-limit-in-ethereum

FAQ

¶

- How to understand how to use Passport JWT library?
- Refer to the library codebase on Github or in node_modules/jsonwebtoken/ i.everify.js
- Use breakpoints
- Experiment using Node. i.e. Runnode
- thennpm install jsonwebtoken
- const JWT = require('jsonwebtoken');
- JWT.decode("eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJlbWFpbCI6Imx0ZnNjaG9lbkBnbWFpbC5jb20iLCJpYXQiOjE1MTMwNjY3NTEsImV4cCI6MTUxMzY3MTU1MSwic3ViIjoiNWEyZjkwZmZi
  UPAQDWTcooOiO69saUVMI")

References

¶

- Express.js server API with JWT authorisation
- Express.js Routes

TODO

¶

- [ ] Integrate withPeerism React Native app
- [X] Integrate Solidity smart contract using TestRPC
- [ ] Create a Truffle Box
- https://github.com/trufflesuite/truffle/issues/433
- http://truffleframework.com/boxes/
- [ ] Upgrade to latest Web3 1.0.0 Beta-27 that has been successfully used in https://github.com/ltfschoen/geth-node to deploy a FixedSupplyToken.sol smart contract to a Private Network with Geth