PictureCash Dokumentáció

Ez a dokumentáció leírja a Blockchain Technológiák és Alkalmazások tárgyra készített házifeladatunk leírása. A projekt, egy olyan alkalmazás, ami az ismert GeoCache[1] játék alapjaira épül, és mindez blockchain technológiai alapokon.

A Program Leírása

A program fő célja, hogy GPS koordináták alapján vannak úgynevezett „cache" objektumok, amiket a játékosok megkereshetnek, és feltéve, hogy tényleg a koordináták alapján egy adott „cache" területén helyezkednek el, „kinyithatják" azt, és beleírhatják a nevüket a felfedezők listájára, illetve kicserélik a „cache" -ben lévő objektumot, a sajátjukra.

A cache-ekben, úgynevezett „trackable" objektumok vannak, mindegyikben szigorúan egy darab lehet egy időben, és ha egy játékos „kinyitja", akkor mindenképpen kicserélődik a játékosnál lévőre. Minden játékosnál csak egy ilyen objektum lehet egy időben. Ezek az objektumok, képek, amik úgy jönnek létre, hogy minden játékos regisztráláskor készít egy képet, ez lesz az ő első „trackable" objektuma, amit el is cserél az első alkalommal, amikor „kinyit" egy cache-t. A ládákba, létrehozáskor, a létrehozó rak bele ugyanilyen módon egy trackable objektumot.

Minden cache-ben található egy log fájl, ami az egyes felhasználókat tárolja el, akik kinyitották az adott cache-t valamikor.

Egy felhasználó létrehozhat cache-t, az általa létrehozott cache-eket módosíthatja, illetve bármelyik cache-nél hagyhat egy jelentést, abban az esetben, ha valami problémát talál azzal kapcsolatban. A jelentéseket a létrehozója olvashatja el.

Fontos, hogy a teszt programban, a trackable objektumok, nem képet, hanem csak egy szöveget tartalmaznak, ez a tesztelés, és demózás szempontjából került módosításra, viszont a program eredeti ötlete továbbra is a képeken alapszik, így ez nem lett módosítva, csupán egy későbbi fejlesztési ciklusra eltolva.

[1] https://geocaching.hu/

A Program felépítése

A program a Visual Studio Code, IBM BlockChain Platform[2] nevű kiterjesztésének segítségével készült, a Hyperledger Fabric[3] környezetre épülve. Nyelvként a golang[4]-et választottuk.

A program fő logikai részeit, a „geo-cache-contract.go" programfájl tartalmazza, ehhez tartozik a „geo-cache-contract_test.go", ami leimplementált függvények teszteseteit tartalmazza.

A program fő eleme, a GeoCache „struct", ami a golang megfelelője egy class-nak, bár nem teljesen ugyanaz, ebbe mélyebben belemenni ezen írásban indokolatlan lenne, ezért a továbbiakban csak „class" -ként fogok ezekre hivatkozni. (bővebb írás erről a témáról: https://golangbot.com/structs-instead-of-classes/)

A GeoCache tehát a legfontosabb class a rendszerben, ez tárolja el a cache-ek adatait, ezen objektumok tranzakcióit fogjuk eltárolni. Ez a következő attribútumok találhatóak benne:

- Name – a létrehozó által megadott neve egy objektumnak (string)
- Description – a létrehozó által megadott leírás a cache-hez (string)
- XcoordRange – az x tengelyen értelmezett koordinátái a cache tartományának (két elemű int lista)
- YcoordRange – az y tengelyen értelmezett koordinátái a cache tartományának (két elemű int lista)
- Owner – a cache létrehozója (User objektum)
- Reports – a cache-re feladott reportok (string lista)
- Visitors – a cache látogatóinak feljegyzése (string lista)
- Trackable – a cache jelenlegi trackable objektuma (Trackable objektum)

Emellett a program tartalmaz még három class-t, a Trackable, a User és a Report. Ezeknek struktúrái:

Trackable – az az objektum, amit a felhasználó cserélget a cachek megtalálásakor, minden cache-ben és minden felhasználónál csak egy lehet egy időben. Attribútumai:

[2] https://marketplace.visualstudio.com/items?itemName=IBMBlockchain.ibm-blockchain-platform
[3] https://www.hyperledger.org/use/fabric
[4] https://go.dev/

- Id – egyedi azonosító (string)
- Value – a kép, a teszt programban ez egy szöveg (string)

User – ez az objektum tárolja el a felhasználó adatait. Attribútumai:

- Id – egyedi azonosító (string)
- Name – regisztráláskor megadott név (string)

Report – egy hibajelentést tároló objektum, a felhasználó hozhatja létre, egy cache-ben lesz tárolva. Attribútumai:

- Id – egyedi azonosító (string)
- Message – a hibaüzenet (string)
- Modifier – a feladó felhasználó adatai (User objektum)

A fő programfájl, egy GeoCacheContract class segítségével menedzseli az egy GeoCacheContract objektum-hoz tartozó metódusokat. Ezen metódusok az alábbiak:

- GeoCacheExists – Egyszerűen megnézi, hogy létezik-e egy adott „key” -hez tartozó Geocache objektum
- CreateGeoCache – Létre lehet vele hozni egy GeoCache objektumot
- ReadGeoCache – Le lehet kérni egy GeoCache objektum adatait
- UpdateGeoCache – Módosítani lehet egy GeoCache objektum adatait
- AddVisitorToGeoCache – Egy GeoCache objektum látogatóihoz hozzá lehet adni egy megadott User objektumot
- SwitchTrackable – Ki lehet cserélni vele egy megadott User és egy megadott GeoCache „key” -hez tartozó objektum trackable-jeit
- UpdateCoordGeoCache – Módosítani lehet egy GeoCache objektum koordinátáit
- DeleteGeoCache – Törölni lehet egy GeoCache objektumot
- ReportGeoCache – Hibajelentést lehet rögzíteni egy megadott „key” -hez tartozó GeoCache objektumhoz
- GetReports – Le lehet kérni egy GeoCache objektumhoz tartozó hibaüzeneteket

Az API működése

Az Api, a fentebb leírt metódusokkal működik, egy mobilokon futó kliensalkalmazás backendjeként működik, a felhasználók, csak kliens oldalon vannak eltárolva, a backend külön nem tartalmaz erre funkciót. A felhasználónak van egy egyedi azonosítója, és ennek a hash-ét tárolja el a program a ledger-ön, így csak az a felhasználó fogja tudni a saját adatait módosítani, aki az eredeti azonosítónak a birtokában van. A hash mellett még salting és stretching technikákat is alkalmaztuk a jelszavak titkosításában. Az API lehetőséget ad a Geocache objektumok létrehozására, módosítására, törlésére, és adataik olvasására. Emellett tartalmaz speciális függvényeket, ezek az előző fejezetben olvashatóak részletesen.

A függvények egy alapvető függvénykészletre támaszkodnak, tulajdonképpen ezekben business logic van implementálva, és előre elkészített függvényekkel hajtja végre a tényleges lekéréseket, objektum létrehozásokat, módosításokat stb. Ezek a függvények a GetState, PutState és a DelSate. Mindegyik függvény, ezen függvények hívásával dolgozik.

Implementált tesztesetek:

Alapvetően tíz előre implementált unit tesztet tartalmaz a kód, mindegyik függvényre jut egy teszt, ami az adott függvény alapvető működését validálja.

A tesztek működése egy teszt környezet felépítésével kezdődik, azért, hogy a tesztekben lefuttatott folyamatok ne mentsenek rá a ledger-re ténylegesen, ezért van erre szükség.

A rendszerfüggvények mindegyike egy State függvényt hív meg, így ezen függvényeknek megadva egy tetszőleges statikus visszatérést, elkerülhetjük, hogy tényleges változások történjenek a blockchainen, miközben tudjuk tesztelni a busines logic layert.

Ezért jött létre a configureStub függvény, ami elvégzi a kellő konfigurációkat a tesztesetek környezetében, illetve meghatározza, hogy egyes state függvények milyen paraméterre, milyen visszatérési értéket adjanak vissza.

Ezekre alapozva minden tesztesetben meg van adva egy eset, a különböző state visszatérési értékekre, hogy azoknál a business logic alapján mit kell visszaadnia a meghívott függvénynek.

Abeadott zip fájl tartalmaz továbbá egy .txt fiájl-t, ami tartalmaz néhány előre megírt json bemenetet, amik segítségével kézzel is lehet tesztelni a program működését.