Kreirati web aplikaciju koja će omogućiti komunikaciju sa Hyperledger Fabric mrežom, kroz pozive chaincode funkcija.

Aplikacija će podržati funkcionalnosti:

- enroll / login korisnika
- upit (query) chaincode-a
- izazivanje (invoke) chaincode-a

Web aplikacija mora da koristi SDK (NodeSDK, JavaSDK, GoSDK…) za rad sa chaincode-om.

Prilikom izrade projekta je neophodno koristiti 2.2.6 verziju svih modula Hyperleder fabric mreže.

Projekat je potrebno raditi u grupama od po troje. Moguće je raditi i u manjim grupama ali obim posla ostaj isti. Tabela za raspored studenata po grupama se nalazi na linku. Studenti se sami raspoređuju u grupe i popunjavaju tabelu. Krajnji rok za formiranje grupa i postavljanje github repozitorijuma u tabelu je 20.01.2024.

Potrebno je da Fabric blockchain obezbedi praćenje i poslovnu logiku kojom se obezbeđuju funkcionalnosti opisane u daljem tekstu.

U sistemu inicijalno postoji nekoliko banaka koje su modelovane na sledeći način:

- idBanke(jedinstveni identifikator)
- sedište
- godina osnivanja
- PIB
- liste/a korisnika i računa (studenti odlučuju kako će modelovati ovu vezu)

Pored banki u sistemu se čuvaju i podaci o korisnicima i računima. Nabrajanja koja su data u specifikaciji projekta su minimalni zahtevi za polja stuktura. Dozvoljeno i poželjno je proširivanje specifikacije.

Korisnik ima:

- idKorisnika(jedinstveni identifikator)
- ime
- prezime
- adresu elektronske pošte
- računi koje korisnik ima

Račun ima:

- količina novca
- valuta
- lista kartica vezanih za račun

Prethodno opisane strukture se upisuju u WorldState, prilikom INIT funkcije chaincode-a, ili neke druge invoke funkcije koja će postaviti početno stanje tako da sadrži minimum 4 banke i svaka ima po bar 3 korisnika sa po jednim ili dva računa.

Funkcionalnosti koje će chaincode obezbediti jesu:

- unos novog korisnika
- kreiranje jednog ili više računa za korisnika
- prenos sredstava sa jednog računa na drugi u okviru jedne ili više banaka
- uplata novca na račun
- dizanje novca sa računa

Zahtevi za projektni zadatak:

- ❏ Aplikacija za interakciju sa chaincode-om mora imati opciju rada sa bar četiri sertifikata. Tj. kroz aplikaciju je moguće logovanje kao četiri različita korisnika Hyperledger mreže koji pripadaju različitim organizacijama.
- ❏ Mapiranje organizacija na specifikaciju projekta je slobodno za interpretaciju. Tj. nije obavezno praviti novog korisnika u Hyperledger fabriku za svaku novu registraciju klijenta banke na web aplikaciju. Dozvoljeno je raditi sa više klijenata **iste** banke koristeći jedan Hyperledger Fabric sertifikat.
- ❏ Kreiranje Fabric mreže sa dva kanala
    - specifikacija uključuje četiri organizacije sa po četiri peer-a
        - ➔ generisanje potrebnih kripto materijala
        - ➔ specifikacija docker kontejnera za elemente Fabric mreže
    - kanalu mora biti pridružen bar 1 orderer
    - svi peer-ovi se nalaze na oba kanala
    - kreiranje svih preostalih neophodnih učesnika u mreži kao i njihove kriptomaterijale (CA, Orderer)
        - ➔ obavezno je koristi CouchDB
        - ➔ Neophodno je napisati par dodatnih bogatih upita koji pokazuju prednosti CouchDB-a i objasniti na odbrani zašto su bas ti upiti zanimljivi i kako bi se ista stvar uradila u levelDB.
        - ➔ svi peer-ovi imaju ulogu endorser-a i commiter-a

- ❏ Chaincode funkcionalnosti
    - Transfer novca
        - ➔ Transfer novca je moguć samo ukoliko korisnik koji prebacuje novac ima dovoljno sredstava na računu. Ukoliko računi nemaju iste valute, pitati korisnika da li želi da izvrši transakciju koristeći srednji kurs za konverziju

koji propisuje Narodna Banka Srbije. Ukoliko korisnik potvrdi da i dalje želi da izvrši transakciju, odraditi konverziju.

- Uplata novca
  - ➔ Može da se izvrši samo u valuti u kojoj je račun napravljen
- query funkcije koje omogućuju pretragu po imenu, prezimenu, broju računa, po prezimenu i adresi istovremeno
- potrebno je testirati sve implementirane fukncionalnosti (kroz pozive funkcije ili REST pozive)
- potrebno je obraditi sve moguće greške (ukoliko ne postoji korisnik ili račun sa zadatim ključem u WorldState-u, nedovoljna količina sredstava i drugo)

❑ U world state-u se čuvaju objekti u JSON obliku, web aplikacija takođe vraća JSON podatke.

- aplikacija koja koristi SDK treba da omogući:
  - ➔ enroll – login
  - ➔ query
  - ➔ invoke

Napomene:

- Preporučuje se pisanje chaincode-a u Golang-u,
- OrderingService koristi RAFT konsenzus algoritam. Potrebno je da commit bloka radi tako da se bar 2 transakcije nađu u bloku ili na svaki sekund, šta god se desi prvo,
- Obavezno je koristiti CouchDB
- Imenovanje elemenata mreže (MSP, Org, Peer, Channel, Orderer itd) je ostavljeno studentu, preporučuje se imenovanje koje je pokazano na vežbama
- Specificiranje pravila prihvatanja transakcije, odnosno *endorsement policy* za chaincode je ostavljen studentu da specificira kako želi
- **SVE ŠTO NIJE EKSPLICITNO NAPISANO U SPECIFIKACIJI PROJEKTA STUDENT MOŽE IMPLEMENTIRATI PO SOPSTVENOJ INTERPRETACIJI!**
- **Obavezno je koristi CA (Certificate Authority) i obavezno je testirati sve funkcionalnosti**
- **.sh** skripte za sve potrebne operacije, koje omogućuju testiranje i pokretanje mreže su obavezne
- Neophodno je obezbediti gitHub pristup asistentu (GitHub username NebojsaHorvat). Takođe je obavezna ravnomerna raspodela posla koja se vidi kroz gitHub istoriju.