Progetto Sistemi Distribuiti 2021-2022 (traccia 2)

Nome: Stefano Yecheng

Cognome: Hu

Matricola: 870084

Linguaggio: Python 3.10.5

Il progetto utilizza Python nel backend e Javascript+HTML nel frontend. Inoltre, nel frontend è presente anche un po' di Jinja2 (https://flask.palletsprojects.com/en/2.1.x/templating/), cioè il template engine standard di Flask e permette l'utilizzo di linguaggio Python all'interno di pagine HTML.

Per permettere il caricamento dei risultati senza un refresh è stato utilizzato il Fetch API di Javascript (https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API) che è una alternativa al XMLHttpRequest visto nel corso di "Sistemi Distribuiti".

Il file "main.py" contiene il codice del backend scritto in Python.

La cartella "templates" contiene i file HTML del frontend.

La cartella "static" contiene i file di JQuery e il plugin per il mask.

Informazioni ulteriori alle specifiche della consegna (i metodi senza informazioni importanti non vengono citati):

Database

Sono presenti 2 database creati e gestiti tramite Flask-SQLAlchemy (https://flask-sqlalchemy.palletsprojects.com/en/2.x/).

accounts.db ha i campi: accountId, name, surname, total.

transactions.db ha i campi: uuid, senderId, receiverId, date, type, divert, amount.

/api/account

- GET: restituisce tutti i risultati di accounts.db.

- DELETE: elimina l'account da accounts.db. Le transazioni effettuate non vengono eliminate.

/api/account/{accountId}

- GET: restituisce nome, cognome, e totale presi da accounts.db e le transazioni da transactions.db

- POST: aggiunge la transazione a transactions.db se valida. Il campo type è "Withdraw" in caso di amount negativo, "Deposit" in caso di amount positivo e "Useless" in caso sia zero. Restituisce la transazione effettuata e il nuovo totale.

La boolean divert è inserita come True perché non è una transazione stornabile.

Se è un prelievo, il senderId è l'accountId. Il receiverId è vuoto perché potrebbe essere l'account di un'altra banca o una banca che fornisce contanti all'utente.

Se è un deposito, il receiverId è l'accountId. Il senderId è vuoto perché potrebbe essere l'account di un'altra banca o l'utente che deposita contanti tramite una banca od un ATM.

Se l'amount è zero, il senderId è l'accountId indipendentemente da quale operazione è stata scelta perché è solo un modo per tenere traccia delle transazioni che nel mondo reale non vengono

- PATCH: modifica nome o cognome. Se il body della richiesta ha tutti e due, allora un messaggio invita l'utente ad usare il metodo PUT.

/api/transfer

- POST: se amount è negativo, genera un messaggio di errore. Inoltre, controlla se gli accountId inseriti sono presenti in accounts.db e se amount è un float.

Genera 2 transazioni con uuid diverso (una che toglie saldo al sender e una che ne aggiunge al receiver).

Aggiunge le transazioni a transactions.db.

Aggiorna i saldi totali dei 2 account coinvolti.

/api/divert

- POST: permette l'operazione solo se l'uuid della transazione è valida, è di type "Send" e non è stata già stornata (divert = False).

Di solito, è l'utente che invia denaro a chiedere alla banca di stornare la transazione.

Genera 2 transazioni con uuid diverso e type "Divert Send" e type “Divert Receive” (una che toglie saldo al receiver e una che ne aggiunge al sender).

Le transazioni "Divert" hanno boolean divert uguale a True perché non sono stornabili.

Il vecchio receiver diventa il nuovo sender e il vecchio sender diventa il nuovo receiver.

Modifica la boolean divert della transazione "Send" a True in modo che non sia stornabile un'altra volta.

Un divert di una transazione “Send” che è presente nel sistema ma non è associata a nessun account (eliminato/i) non è possibile.

Aggiorna i saldi totali dei 2 account coinvolti.

/

Mostra un campo di input e un bottone (oltre al reset) che funziona solo se l'utente inserisce 20 caratteri (lunghezza di un account id).

Il bottone manda una richiesta GET a "/api/account/{accountId}" e stampa i risultati nella stessa pagina.

/transfer

Mostra 3 campi ed un bottone (oltre al reset) che funziona solo se gli account id sono di 20 caratteri e l'amount è un numero positivo.

Possono essere inseriti solo 20 caratteri nel campo di input dell'account id.

Il bottone manda una richiesta POST a "/api/transfer" e stampa i risultati nella stessa pagina.

/account

Mostra 2 campi di input e un bottone (oltre al reset) che funziona solo se i due campi sono inseriti.

Il bottone manda una richiesta POST a "/api/account" e stampa le informazioni del nuovo account nella stessa pagina.

/accounts

Mostra una tabella con tutti gli account presenti in accounts.db

/divert

Mostra un campo di input ed un bottone (oltre al reset) che funziona solo se vengono inseriti 32 caratteri (esclusi trattini). È presente una mask, che utilizza JQuery ed un plugin (https://igorescobar.github.io/jQuery-Mask-Plugin/), per il campo dell'uuid al fine di inserire automaticamente i trattini.

Il bottone manda una richiesta POST a "/api/divert" e stampa il risultato nella stessa pagina.

/cash

Mostra 2 radio buttons, 2 campi di input e un bottone (oltre al reset) di conferma. Il bottone funziona solo se l'account id ha 20 caratteri e l'amount è un numero positivo.

Le transazioni con amount 0 vengono mostrate con la tipologia "Deposit" o "Withdraw" a seconda del valore del radio button, ma vengono salvate nel database come "Useless" perché sono, appunto, inutili. "/search" le visualizza con type "Useless".

Deposit è l'opzione di default.

Il bottone manda una richiesta POST al "/api/account/{accountId}" e stampa la transazione nella stessa pagina.

/delete

Mostra un campo di input per l'account id in cui possono essere inseriti solo 20 caratteri. L'account viene eliminato da accounts.db se è presente al suo interno.

Note:

- Una transazione è possibile verso sé stessi anche se è zero.
- I depositi e i prelievi con amount 0 sono possibili.
- Il tasto reset nelle pagine resetta sia i campi di input che i risultati precedenti.