University of British Columbia, Department of Computer Science

CPSC 304

Summer 2018

Project Part 3

Project Formal Specification

Group Members:

Name	Student Number	Unix ID	Tutorial Section	Email Address
Wongelawit Teka Zewde	32493141	a8d0b	T1A	gospel.teka@gmail.com
Stephanie Wu	60030137	a5I0b	T1A	stephwu2000@gmail.com
Daniela Shklover	18491143	u5s0b	T1A	danielashklover@gmail.com
Ben Walker	34334169	m3c1b	T1E	bnwlkr@icloud.com

By typing our names and student numbers in the above table, we certify that the work in the attached assignment was performed solely by those whose names and student IDs are included above.

In addition, we indicate that we are fully aware of the rules and consequences of plagiarism, as set forth by the Department of Computer Science and the University of British Columbia.

What Platform Will You Use?

We will be using the CS UGrad Oracle installation and provided PHP/Apache.

We will be using a simple React App for the frontend.

What Functionality Will the Final Application Have?

A user using our system will be able to make accounts, fill the account with their personal information, such as address and telephone number. The user's account will then have a balance of money the user deposits and can also withdraw, the user is also able to query and see only their own balance. The user's account is related to a specific exchange and a user can have single accounts with many exchanges. Through these exchanges a user can buy or sell commodities including stock and e-currency. The user will also be able to see the commodity information and price history by querying specific commodity ids.
In specific these are the queries that will be supported for the user:

Activity/Query	**Input**	**Output**	**Cases**
Create an Account	Name, date of birth, email, credit card, password	Account Created	When user starts using our application
Edit Personal Information	Name, date of birth, email, credit card, password	Info edited	When user moves or changes phone number
Deposit Money	Amount (double)	Money sent to balance.	When user wants to add money to account.
Withdraw Money	Amount (double)	Money sent to bank.	When user wants to withdraw money.
Buy Commodity (e-currency or Stock)	Id of Commodity, Amount of stock to be bought.	Commodity now in user portfolio.	When user wants to buy.

Sell Commodity (e-currency or Stock)	Id of Commodity, Amount of stock to be sold.	Commodity sold and out of portfolio.	When user wants to sell.
Get Fee	Id of Commodity, Amount of commodity to be sold, “Sell” or “Buy.”	Find out fee of buying or selling commodity.	When a user wants to see fees.
See Commodity info and trends	Id of Commodity	See all info of Commodity available.	Research.
Choose to see commodities over or under certain price per unit.	Price	See all commodities below or above threshold price	When user wants to spend a certain amount of money.

What Data Will be in the Final Application?

To aid in the explanation, our schema has been included and annotated line by line. The data that we will have in our final application will include:

User(id: INT, name: CHAR(20), email: CHAR(30))

- The ids (assigned upon registration), names, and emails of registered users

Trade(id: INT, timestamp: INT, price: DOUBLE, ***userID: INT****, bought?: BOOL)*

- The ids (assigned upon performance of trade), time of trade, trade valuation, id of the user who performed the trade, and whether the trade was a buy or a sell.

Exchange(name: CHAR(20), website: CHAR(30))

- The name and website of an exchange.

*Traded_On(****exchange_name: CHAR(20)****,* ***commodity_name: CHAR(20)****)*

- The names of commodities and the exchanges they were traded on

Metric(name: CHAR(20), timestamp: INT, ***tradedon_commodity: CHAR(20)****,*
tradedon_exchange: CHAR(20))

- The superclass type that encompasses various the various metrics of a trade. Data includes the metric name (enumerative?), time the trade was performed, the commodity that was traded, and the exchange that the commodity was traded on.

In addition to the data stored in the superclass type additional subclass attribute data includes:

*Price(**metric_name: CHAR(20)**, buy: DOUBLE, sell: DOUBLE)*
- The buying price and selling prices associated with the trade.

*Volume(**metric_name: CHAR(20)**, daily: INT, hourly: INT)*
- Volumes that the commodity has been traded in on a daily and hourly basis.

*Trend (**metric_name: CHAR(20)**, value: INT, source CHAR(30))*
- The relative upwards/downwards value of a commodity as well as information about the source that the data comes from.

Commodity(name: CHAR(20))
- The superclass type that encompasses the various commodity name

*Coin(**commodity_name: CHAR(20)**, ICO_value: DOUBLE)*
- The value of the initial coin offering if the commodity is a coin.

*Stock(**commodity_name: CHAR(20)**, company_site: CHAR(30))*
- The company site if the commodity is a stock.

What Will the Division of Labour be Within Your Team?

Daniela Shklover and Ben Walker: Are mainly in charge implementing the db query.
Wongelawit Teka Zewde and Stephanie Wu: we will implement the GUI.
We will be working together to populate the data.

What Would You Like Feedback On?

We are good for now but we will reach out for help in the future. Thank you.