**BACKEND SKILLS ASSESSMENT PROJECT**

For this task you will be creating a Java application that can be run from the command line. This application will take various input parameters and store data about transactions. The application should process the parameters and then exit. Please follow good object oriented coding practices with your code.

The application should store data on the file system under the folder where the application is. The methods on the interface for the code that is in charge of storing the data should be written such that the code calling the interface should not have to be changed if you later changed the data storage implementation so that data is stored in a database.

Please be prepared to explain what kind of database you would use if you were to store the data in a database in the future. Be prepared to describe the tables and columns you would use (if you were to choose a relational database), or the collections you would use (if you were to choose a NoSQL database)

Your command line should be able to handle the following command line inputs

***ADD TRANSACTION***

./application <user\_id> add <transaction\_json>

This command should add a transaction to the user specified in <user\_id> using the information specified in transaction\_json. The transaction json will have the following format:

*{ “amount”: 1.23, “description”: “Joes Tacos”, “date”:”2018-12-30”, “user\_id”: 345 }*

This command should print out a version of the transaction added with a unique id for the transaction like this:

*{ “transaction\_id”: “2299ce24-9eaf-417f-82d6-e57f93777dc4”, “amount”: 1.23, “description”: “Joes Tacos”, “date”:”2018-12-30”, “user\_id”: 345 }*

***SHOW TRANSACTION***

./application <user\_id> <transaction\_id>

This command should return the transaction specified in the transaction\_id. If the user\_id is not the user\_id that corresponds with the user\_id for the specified transaction, you should print out

*Transaction not found*

If the transaction does exists, you should print out the information for the transaction like this:

*{ “transaction\_id”: “2299ce24-9eaf-417f-82d6-e57f93777dc4”, “amount”: 1.23, “description”: “Joes Tacos”, “date”:”2018-12-30”, “user\_id”: 345 }*

***LIST TRANSACTIONS***

./application <user\_id> list

This command should print all the transactions associated with the user specified by user\_id. The transactions should be in chronological order. If the user\_id does not exist, then the response should return an empty list. You should print the items in the following format:

*[*

*{ “transaction\_id”: “2299ce24-9eaf-417f-82d6-e57f93777dc4”, “amount”: 1.23, “description”: “Joes Tacos”, “date”:”2018-12-30”, “user\_id”: 345 },*

*{ “transaction\_id”: “5467ce24-9eaf-417f-82d6-e57f4444444”, “amount”: 5.26, “description”: “Freds’s Tacos”, “date”:”2018-12-19”, “user\_id”: 345 }*

*]*

***SUM TRANSACTIONS***

./application <user\_id> sum

This command should sum all the transactions associated with the user specified by user\_id. It should print out the sum in the following format

*{ “user\_id”: 123, “sum”: 234.76 }*

Please write this application in Java. You are free to implement this however you’d like with whatever resources or 3rd party code you’d want. Feel free to ask questions at any time (kevin.davies[@payclip.com](mailto:jake@payclip.com))