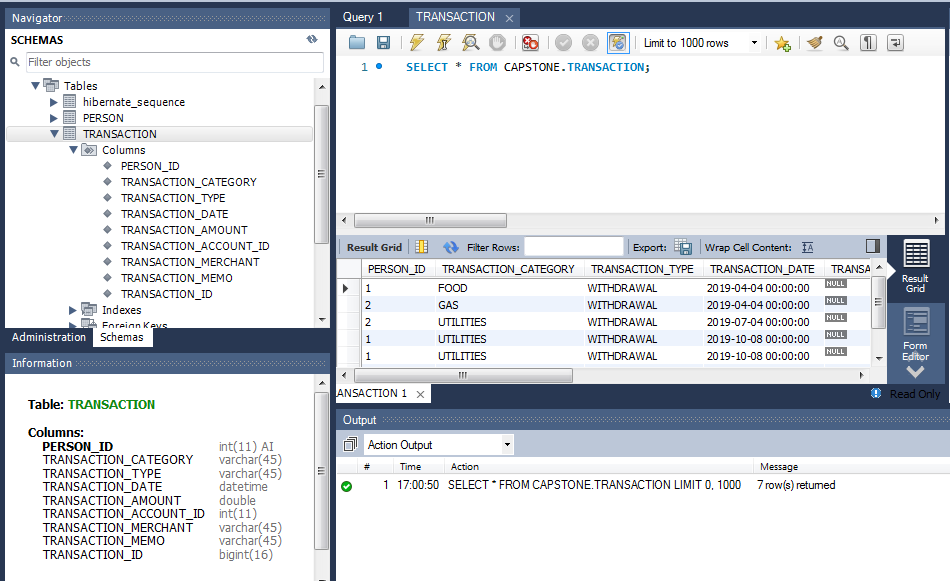
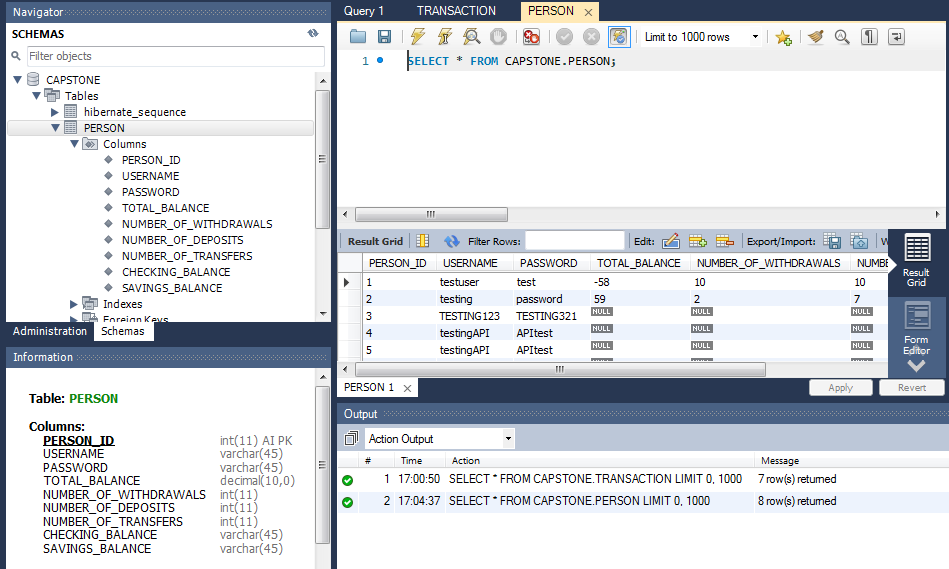
**User Guide and Testing**

**SQL Database**

* Consists of two tables – TRANSACTION and PERSON.
* The Transaction table stores all transactions for all users and includes the following fields: PERSON\_ID, TRANSACTION\_CATEGORY, TRANASACTION\_TYPE, TRANSACTION\_DATE, TRANSACTION\_AMOUNT, TRANSACTION\_ACCOUNT\_ID, TRANSACTION\_MERCHANT, TRANSACTION\_MEMO, TRANSACTION\_ID.
* The transaction table is linked to the person table via PERSON\_ID, which is forced to be unique.

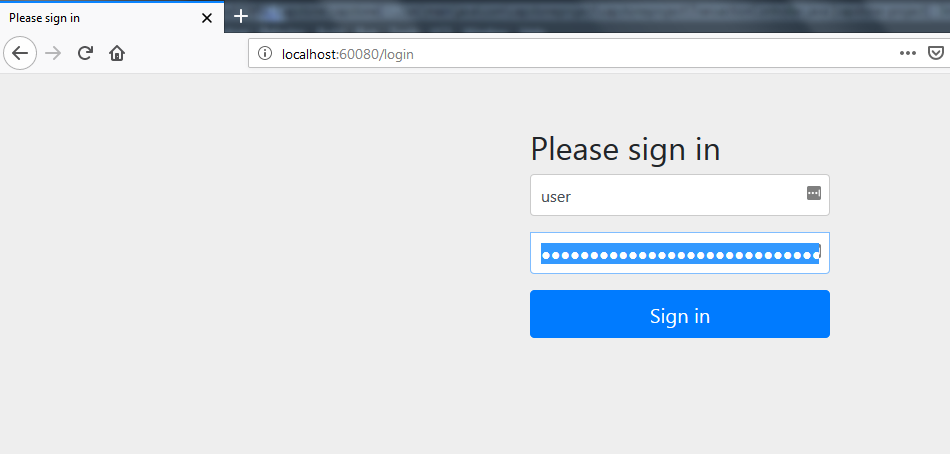


* The person table stores all user information and includes the following fields: PERSON\_ID, USERNAME, PASSWORD, TOTAL\_BALANCE, NUMBER\_OF\_WITHDRAWALS, NUMBER\_OF\_DEPOSITS, NUMBER\_OF\_TRANSFERS, CHECKING\_BALANCE, SAVINGS\_BALANCE.



**Backend Design**

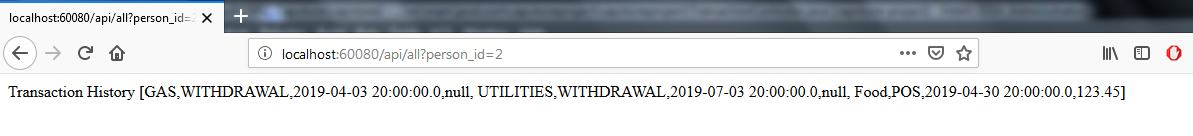
* Data is managed in the database utilizing Java Spring Boot in our backend code and a custom REST API that we developed for access from our frontend code.
* We created two repositories to access and edit the database. The first repository is meant to handle transactions, while the second repository handles user access and account creation.
* The backend application/API can be accessed via <http://localhost:60080/api>.



* Accessing endpoints –

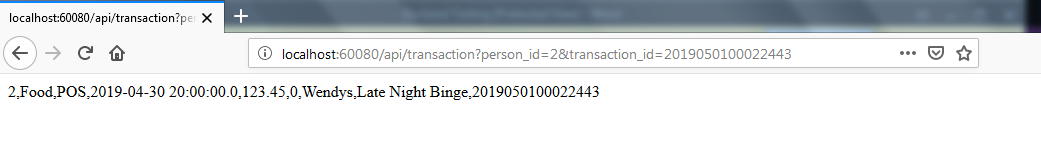
1. /all – Returns all transactions for a specific person. The PERSON\_ID must be sent as a parameter in the url request.

Example request: <http://localhost:60080/api/all?person_id=2>



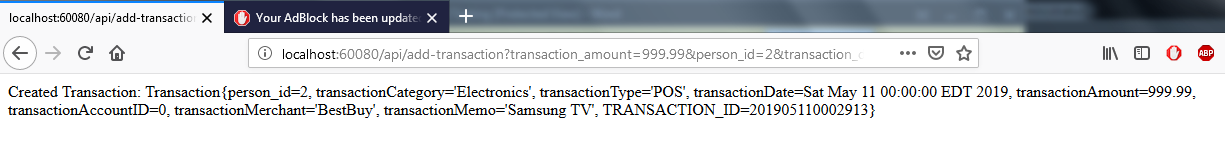
1. /transaction – Returns a specific transaction. A person\_id and transaction\_id must be sent as parameters in the url request.

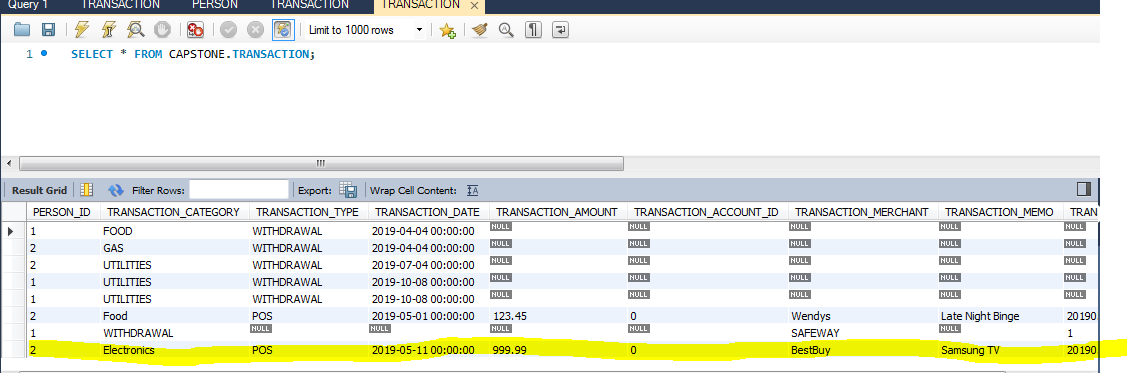
Example request: <http://localhost:60080/api/transaction?person_id=2&transaction_id=2019050100022443>



1. /add-transaction – Adds a transaction to the database. All fields for the transaction, with the exception of transaction ID, must be sent as parameters. The transaction ID is dynamically generated by the backend to ensure uniqueness.

Example request: <http://127.0.0.1:60080/api/add-transaction?transaction_amount=999.99&person_id=2&transaction_date=2019-05-11&transaction_type=POS&transaction_category=Electronics&transaction_memo=Samsung%20TV&transaction_merchant=BestBuy&transaction_account_id=0>





1. /remove-transaction – Removes a transaction from the database. Person\_ID and Transaction\_ID must be sent as parameters in the URL request.

Example request: <http://localhost:60080/api/remove-transaction?person_id=2&transaction_id=201905110002913>

