Test Plan Approach (Correctness):

We believe our test plan adequately demonstrates the correctness of our backend implementations because heavily emphasized testing all functions in relation to each other during a single instantiation. Although we leaned towards glass box testing, we included a variety of different types of testing to incorporate the vast majority of functionalities utilizes as well as the combination of functionality utilizes within our functions. This was prevalent for the majority of Account.ml, and the entirety of Transaction.ml and Blockchain.ml. This is why we believe our testing ensures correctness in the backend.

For other parts of our program in Accounts.ml that aren't tested in Ounit, and files executable menu.ml, we manually tested our code. This was because of database threading issues with testing concurrently in our testing file, furthermore, because the bulk of menu.ml was our GUI implementation, testing it by running was the best approach towards ensuring correctness and personal preference for its design. Therefore, this is why we believe our testing ensures correctness in the frontend.

How developed and tested (manually or automatically):

These three files (Account.ml, Transaction.ml, Blockchain.ml) were tested automatically with Ounit2. These tests were developed as glass box tests and black box tests, although admittedly we heavily favored glass box testing through the three modules. They were developed this way given our integration issues with the GUI, and our continuous integration which was strenuous and required a great deal of testing clarity.

Functions in accounts.ml that interacted with databases and GUI implementations in menu.ml were tested manually, and done so as we continuously improved functionalities for our GUI with greater complications from the backend. These were done manually via utop or running our executable menu, given we thought it was the best approach to accommodate integration and efficiency.