**Test Plan:**

**Test Input and Output Organization:**

The tests are organized into separate directories depending on what they do and whether they are an input or an output. For example, we have two directories called “Input” and “Output” and sub-directories for each function, such as “create”, “delete”, “login”, etc.

The test files themselves are named after the transaction they are meant to test and are numbered sequentially according to what our test case list has them as. In terms of how the tests are named, we have our test files named such that the format is the name of the transaction, followed by the number in the sequence, followed by whether it is an input or an output, for example “login1\_input.txt” indicates that it is meant to test login and that it is an input file. For our output files, we have two categories for them, “expout” indicates that the file is our expected output and “expfile\_output” indicates the expected file output for our transaction file. These two output files will be placed in the same directory for the sake of simplicity. For when we run our tests, the test outputs will be placed in the same directory as the expected outputs and they will be named in a similar format as our expected output in which they will have “\_actout” and “\_actfile\_output” to indicate the actual terminal output and actual transaction file output respectively.

**Test Run Plan:**

For these tests, they will be run using shell script commands that will run the Front End for each of our test input files in the directory one at a time. As it runs each test input, it will generate the output that we get from the console and save them onto a text file. It will then compare that text file, the actual output to our expected output text file. The script will then compare the expected output to the actual output and produce what the differences are, if any.

Given that some of these tests require some preexisting data in the front end to work, such as delete, login, sell, and buy, it would be best to have preset files for the current user accounts and available tickets file just made for the purpose of testing, or else we would unable to properly test some of the transactions in the front end since there would be no existing data.

**Requirements Problems:**

Problems and oversights found in the requirements were any numeric inputs for any transaction that required them, for this, we assume that in addition to the stated restrictions for them, that any negative number will be considered an invalid input.

For the buy transaction, there is no explicit constraint against the user entering more than 25 characters for an event title and the user would not normally know of this, so we assume when the user inputs more than 25 characters, they will get an error and in addition, we have a test case for that.

Most services that uses accounts would have a username and password, however, the requirements don’t explicitly state the necessity for one. For this, we assume that they are not part of the system and only have usernames as a way to log in.