CSCI 3060U- Phase 1: Front End Requirement

Group: Quinton Belcastro and Alex Zheng

Test Cases

login	login1	Try to login with no username (0 chars too short)
	login2	Try to login with username that does not exist in user accounts file
	login3	Try to login with a real user (should pass)
	login4	Try the “login” transaction once a successful login occurs (should fail)
logout	logout1	Try and use logout when logged in transaction (should work)
	logout2	Try and use logout command again after logging out (should fail)
	logout3	Try and use the create transaction after logging out (should fail)
	logout4	Try and use the delete transaction after logging out (should fail)
	logout5	Try using the sell transaction after logging out (should fail)
	logout6	Try using the buy command after logging out (should fail)
	logout7	Try using the refund transaction after logging out (should fail)
	logout8	Try using the addcredit transaction after logging out (should fail)
create (UNames)	create1	Try creating a user with a name that is too long (15 char

		max)
	create2	Try creating a user with a blank name (should fail)
	create3	Try creating duplicate users with the same name (not allowed)
create (Type)	create4	Try creating a proper username with ‘Admin’ type
	create5	Try creating a proper username with ‘full-standard’ type
	create6	Try creating a proper username with ‘buy-standard’ type
	create7	Try creating a proper username with ‘sell-standard’ type
create (running perms)	create8	Try to run the create command as a non-privileged user (should fail)
create (values)	create9	Try and create a user with a credit larger than 999,999.99 (this is the proper max)
	create10	Try creating a user with a type that doesn’t exist.
	create11	Try creating a user with a credit with not a number
	create12	Try setting the initial credit value to a negative number
	create13	Try and create a user with “ “ a space as a username (should fail)
delete	delete1	Try the command as an unprivileged user (should fail)

	delete2	Try to delete the currently logged in user (should fail and output error message)
	delete3	Try to delete a username that doesnt exist (should fail and output appropriate error message)
	delete4	Try to delete a username that already exists as a privileged user (should pass)
Delete (transactions on deleted users)	delete5	Try to perform a refund transaction on deleted users
	delete6	Try to perform sell transaction on deleted user
	delete7	Try to perform the buy transaction on a deleted user
	delete8	Try to perform the refund transaction on a deleted user
sell	sell1	Try to sell ticket with empty event title
	sell2	Try to enter an event title with more than 25 characters
	sell3	Try to enter a negative price
	sell4	Try to enter a price > $999.99
	sell5	Try to enter a number with more than 1 cent of precision (e.g. 123.456)
	sell6	Try to enter amount < 0 for # of tickets to sell
	sell7	Try to enter amount = 0 for # of tickets to sell
	sell8	Try to enter amount > 100 for # of tickets to sell
	sell9	Try to enter non-numeric

		characters (excluding period) for price
	sell10	Try to enter more than one period for price
	sell11	Try to enter non-numeric characters for # of tickets to sell
	sell12	Try to enter period (for non-integer values) for # of tickets to sell
	sell13	Try to sell tickets on “standard-buy” account
buy	buy1	Try to purchase a ticket(s) with event title blank
	buy2	Try to purchase a ticket(s) with # of tickets to buy blank
	buy3	Try to purchase a ticket(s) with seller username blank
	buy4	Try to enter event title with > 25 characters (limit)
	buy5	Try to enter non-numeric characters into # of tickets to buy
	buy6	Try to purchase 5 or more tickets as non-privileged user/account
	buy7	Try to purchase more than available amount of tickets to an event
	buy8	Try to purchase a ticket(s) with “standard-sell” account
	buy9	Try to purchase a negative amount of tickets
	buy10	Try to purchase tickets with insufficient amount of credit

		on account
refund	refund1	Try to issue a refund as a non-privileged user/account
	refund2	Try to leave buyer username blank
	refund3	Try to leave seller username blank
	refund4	Try to leave amount of credit to transfer blank
	refund5	Try to enter non-numeric characters (excluding period) for amount of credit to transfer
	refund6	Try to enter more than one period for amount of credit to transfer
	refund7	Try to issue a refund with credit that has more than 1 cent of precision (e.g. $123.456)
	refund8	Try to issue refund to deleted buyer from existing seller
	refund9	Try to issue refund to existing buyer from deleted seller
addcredit	addcredit1	Try to issue an addcredit on non-admin/non-privileged account/user
	addcredit2	Try to leave amount of credit to add blank
	addcredit3	Try to leave username of account to add credit to blank
	addcredit4	Try to add < $0.01 credit to an account during a session
	addcredit5	Try to add > $1000.00 credit to an account during a

		session

“Current user” - Existing (not deleted) users

Test Plan:

Test Input and Output Organization:

We will use the format of [transaction][number].inp for input files [transaction][number].out for output files and [transaction][number].tout for the daily transaction file outputs. Each different transactions tests will be saved in their own directories and the output files will be seperated in subdirectories. In the case that the results fail and there should be no updates to the daily transaction file. If the file passes the test and the output matches the output file the results are saved to a master results file which has the results of subsections of tests (one for each type of transaction) ie one result file for login, logout create etc. In the case where the transaction file would be blank, ie when there are no successfully ran transactions the transaction file is omitted.

Test Run Plan:

Since the naming scheme is consistent for all test cases a batch file or shell script will be made to iteratively run all of the test cases. The transaction tests will be run in the order: login, logout, create, delete, addcredit, sell, buy, refund in order to make sure that no tests rely on transactions that have not been tested. The output from the tests will be checked against the output files to confirm correctness. In the event that a test fails that a later test relies on that output will be recorded and the running sequence will stop.

Other Requirements Integrated:

Through discussion with the customer we were able to identify many additional front-end requirements for this project such as:

- minimum amount for addcredit cant be below 0
- " " is not a valid name
- Max credit is actually 999999.99
- On create the user should be prompted to enter the starting credit amount
- Min username length is 1
- full-standard can buy and sell but can't create, delete, or refund
- For any and all transactions that encounter errors such as those, there should be error messages
- A seller cannot buy their own tickets.
- no further transactions should be accepted on a new ticket
  for sale until the next system cycle. (i.e. when the back end is run)

- You should be able to handle anything unicode (like Üsername) but not anything really weird (like u͓̽s͇̩̩̩͕ern̤̥a̢̙͕̬m̷͔̍e̳̫)
- user accounts must have different usernames
- Order should be Name, Type, Credit for creating users
- Logout should still work even if all the other transactions fail, so it won't be blank.