Software Quality

Swift Ticket

Software Requirements Specification

Created by

**Daniel Smullen**

**Jonathan Gillett**

**Rayan Alfaheid**

Document Revision 2.0

22/03/2013

Table of Contents

[Software Requirements Specification 3](#_Toc351733451)

[Front End Requirements 3](#_Toc351733452)

[Back End Requirements 28](#_Toc351733453)

# Software Requirements Specification

Where assumptions are not listed, citations are given to provide ample evidence of requirements. Please refer to Appendix A.

## Front End Requirements

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Item #** | **Requirement Name** | **Requirement Category** | **Requirement Description** | **Assumptions** | **Citation** |
| 1 | Front-end Language | Non-functional | The front-end must be written in C++. |  | Project Description, Page 2 |
| 2 | Console Application | Functional, Interface | The program must be run from the command line, and accept input/output from text files. |  | Project Description, Page 2  Appendix A Table 1 Item 3 |
| 3 | Front-End Must Never Crash | Non-Functional | The front-end for the application must never crash or stop except as directed by transactions. |  | Project Description, Page 6 |
| 4 | Gracefully Handle Bad Input | Non-Functional | The front-end must not depend on valid input, and must gracefully handle bad inputs of any type. |  | Project Description, Page 6 |
| 5 | Login Transaction | Functional | Entering “login” into the input must start a front-end session. |  | Project Description, Page 4 |
| 5.1 | Login Accounts File | Functional | Before processing a login transaction, the front end must read in the current user accounts file.  An error must occur and the program must not crash if there are invalid entries in the user accounts file. |  | Project Description, Page 4  Appendix A Table 1 Item 21 |
| 5.2 | Login User Authentication | Functional | The login transaction must ask for a username. |  | Project Description, Page 4 |
| 5.2.1 | Login User Authentication Error | Functional | If the login entered is invalid, an error must occur. |  | Project Description, Page 4 |
| 5.3 | Read Available Tickets File | Functional | After the username is accepted, the frontend must read in the available tickets file and begin accepting transactions. |  | Project Description, Page 4 |
| 5.3.1 | Available Tickets File Error | Functional | Error must occur if there are invalid entries in the available ticket file. |  | Project Description, Page 4 |
| 5.4 | Login Sequence of Events | Functional | No transaction other than login must be accepted before a login has occurred. |  | Project Description, Page 4 |
| 5.5 | Subsequent Logins | Functional | No subsequent login must be accepted after a login, until after a logout. |  | Project Description, Page 4 |
| 5.6 | Unprivileged Login | Functional | After a non-admin login, only unprivileged transactions must be accepted |  | Project Description, Page 4 |
| 5.7 | Privileged Login | Functional | After an admin login, all transactions must be accepted (where applicable). |  | Project Description, Page 4 |
| 6 | Logout Transaction | Functional | Entering “logout” into the console session must end the front-end session. |  | Project Description, Page 4 |
| 6.1 | Write Daily Transaction File | Functional | The logout transaction must write the daily transaction file to the file system and end the session. |  | Project Description, Page 4 |
| 7 | Create Transaction | Functional | Entering “create” into the input must begin the process of adding a new user. |  | Project Description, Page 4 |
| 7.1 | Creating a User | Functional | After executing the create transaction, the user must be prompted to enter a username, and user type. |  | Project Description, Page 4 |
| 7.1.1 | Username Length Constraint | Functional | A new username must be at least 1 character, and no longer than 15 characters. |  | Project Description, Page 4  Appendix A Table 1 Item 24 |
| 7.1.2 | Username Naming Constraint | Functional | The username must not contain the string “END “(with a space). |  | Appendix A Table 1 Item 4 |
| 7.1.3 | Username Uniqueness Constraint | Functional | New usernames must be different from all other current users. |  | Project Description, Page 4 |
| 7.2 | User Type Input | Functional | After executing the create transaction and entering the username, the user must be prompted to enter a user type. |  | Project Description, Page 4 |
| 7.2.1 | Valid User Types | Functional | The only accepted user types must be:  - “admin”  - “full-standard”  - “buy-standard”  - “sell-standard” |  | Appendix A Table 1 Item 10 |
| 7.3 | Create Transaction Privileges | Functional | The “create” transaction is a privileged transaction, and must only be accepted when logged in as admin user. |  | Project Description, Page 4 |
| 7.4 | Save Create Transaction to the Daily Transaction File | Functional | After the create transaction has been executed by the user, the create transaction must save the user creation information to the daily transaction file. |  | Appendix A Table 1 Item 7 |
| 8 | Delete Transaction | Functional | Entering “delete” into the input must begin the process of removing a user account. |  | Project Description, Page 5 |
| 8.1 | Delete Transaction Privileges | Functional | The delete transaction is a privileged transaction and must only be accepted when logged in as an admin user. |  | Project Description, Page 5 |
| 8.2 | Delete User Input Constraint | Functional | The delete transaction must prompt for a username to delete. |  | Project Description, Page 5 |
| 8.2.1 | Delete Existing User Constraint | Functional | The username specified in the delete transaction must be the name of an existing user. |  | Project Description, Page 5 |
| 8.2.2 | Delete Current User Constraint | Functional | The username must not be the name of the current active user session. |  | Project Description, Page 5 |
| 8.3 | Delete User Ticket Cancellation | Functional | The delete transaction must cancel any outstanding tickets for purchase or sale for the specified user. | This will be completed by the backend and is not handled by the frontend. | Project Description, Page 5  Appendix A Table 1 Item 7 |
| 8.4 | Save Delete Transaction to the Daily Transaction File | Functional | After the delete transaction has been executed by the user. The delete transaction must save this information to the daily transaction file. |  | Appendix A Table 1 Item 7 |
| 8.5 | Delete User from Current User Accounts File | Functional | After the delete transaction has been executed by the user, the user must be removed from the current user accounts file. |  | Appendix A Table 1 Item 7 |
| 9 | Sell Transaction | Functional | Entering “sell” into the input must begin the process of selling a ticket or tickets to an event. |  | Project Description, Page 5 |
| 9.1 | Sell Transaction Privileges | Functional | The sell transaction is a semi-privileged transaction that is only accepted when logged in as any type of account except buy-standard. |  | Project Description, Page 5 |
| 9.2 | Sale Event Title Input | Functional | After executing the sell transaction, the user must be prompted for the event title. |  | Project Description, Page 5 |
| 9.2.1 | Event Title Length Constraint | Functional | The maximum length of an event title must not contain more than 19 characters. |  | Project Description, Page 5  Appendix A Table 1 Item 15 |
| 9.2.2 | Event Name Constraint | Functional | The event title must not contain the string “END “(with a space). |  | Appendix A Table 1 Item 4 |
| 9.3 | Ticket Price Input | Functional | After the event title has been entered, the user must be prompted for the ticket price in dollars. |  | Project Description, Page 5 |
| 9.4 | Ticket Sale Price Input Format | Functional | The ticket sale price input format must be a double value. |  | Project Description, Page 5  Appendix A Table 1 Item 14 |
| 9.4.1 | Valid Dollars Format Constraint | Functional | The dollar amount for the sale price must have the correct format - it must contain at least one digit followed by a decimal and must contain only two digits after the decimal. |  | Appendix A Table 1 Item 14 |
| 9.4.2 | Ticket Price Constraint | Functional | The minimum price for a ticket must not be less than 0 dollars, and the maximum sale price for a ticket must not exceed 999.99. |  | Project Description, Page 5  Appendix A Table 1 Item 20 |
| 9.5 | Ticket Volume Input | Functional | After entering the ticket sale price the user must be prompted for the number of tickets to make available for sale. |  | Project Description, Page 5 |
| 9.5.1 | Ticket Volume Input Constraint | Functional | The maximum number of tickets for sale input must be an integer value. |  | Project Description, Page 5  Appendix A Table 1 Item 6 |
| 9.5.2 | Ticket Volume Constraint | Functional | The maximum number of tickets for sale must not exceed 100. |  | Project Description, Page 5  Appendix A Table 1 Item 6 |
| 9.6 | Save Sell Transaction to the Daily Transaction File | Functional | After the sell transaction has been executed by the user. The sell transaction must save this information to the daily transaction file. |  | Appendix A Table 1 Item 8 |
| 9.7 | One Sale Transaction per Session Constraint | Functional | No further transaction will be accepted on new tickets for sale until the next session. |  | Appendix A Table 1 Item 9 |
| 10 | Buy Transaction | Functional | Entering “buy” into the input must begin the process of buying a ticket or tickets to an event. |  |  |
| 10.1 | Buy Transaction Privileges | Functional | The buy transaction is a semi-privileged transaction and must only accepted when logged in as any type of account except standard-sell. |  | Project Description, Page 5 |
| 10.2 | Buy Event Title Input | Functional | After executing the sell transaction the user must be prompted for the event title. |  | Project Description, Page 5 |
| 10.2.1 | Buy Event Title Exists Constraint | Functional | The event title must be a current title that exists in the available tickets file. |  | Project Description, Page 5 |
| 10.2.2 | Buy Event Name Constraint | Functional | The event title must not contain the string “END “(with a space). |  | Appendix A Table 1 Item 4 |
| 10.3 | Tickets Purchase Input | Functional | After entering the event title, the user must be prompted for the number of tickets to purchase. |  | Project Description, Page 5 |
| 10.3.1 | Tickets Purchase Number Constraint | Functional | A non-privileged user must not be able to purchase more than 4 tickets in one transaction. |  | Project Description, Page 5 |
| 10.3.2 | Privileged Tickets Purchase Volume Constraint | Non-Functional | Privileged accounts must not have ticket purchase limits. |  | Project Description, Page 5 |
| 10.4 | Seller Username Input | Functional | After entering the event title the user must be prompted for the seller’s username. |  | Project Description, Page 5 |
| 10.4.1 | Seller Must Exist Constraint | Functional | The seller’s username must exist in the user file. |  | Project Description, Page 5 |
| 10.4.2 | Seller Must Not be Current User Constraint | Functional | The seller’s username must not be the name of the current active user session. |  | Appendix A Table 1 Item 16 |
| 10.4.3 | Seller Must Be Selling Ticket Constraint | Functional | The seller must be selling the ticket specified by a buyer in order for a transaction to take place. |  | Project Description, Page 5 |
| 10.5 | Display Ticket Information | Functional | After the event title, number of tickets and seller’s username are specified the cost per ticket and total cost of tickets must be displayed. |  | Project Description, Page 5 |
| 10.6 | Ticket Purchase Confirmation Prompt | Functional | After displaying ticket price information the user must be prompted to confirm the transaction with “yes” or “no”. |  | Project Description, Page 5 |
| 10.6.1 | Ticket Purchase Confirmation Input | Functional | Users must confirm the ticket purchase transaction with “yes” or “no”. | An error should occur if something other than “yes” or “no” is used to confirm the transaction. | Project Description, Page 5 |
| 10.7 | Event Tickets Available Constraint | Functional | User must not purchase more tickets from the specified seller than are available. |  | Project Description, Page 5  Appendix A Table 1 Item 18 |
| 10.8 | Subtract Tickets From Seller Inventory | Functional | The number of tickets sold must be subtracted from the seller’s inventory. |  | Project Description, Page 5 |
| 10.9 | Save Buy Transaction to the Daily Transaction File | Functional | After the buy transaction has been executed by the user. The buy transaction must save this information to the daily transaction file. |  | Appendix A Table 1 Item 7 |
| 11 | Refund Transaction | Functional | Entering “refund” into the input must begin the process of issuing credit to a buyer’s account from a seller’s account. |  | Project Description, Page 6 |
| 11.1 | Refund Transaction Privileges | Functional | The refund transaction is a privileged transaction and must only be accessible by accounts which are of the admin type. |  | Project Description, Page 6 |
| 11.2 | Refund Buyer Username Input | Functional | After executing the refund transaction the user must be prompted for the buyer’s username. |  | Project Description, Page 6 |
| 11.2.1 | Refund Buyer Must Exist Constraint | Functional | The buyer must be a current user that exists. |  | Project Description, Page 6 |
| 11.3 | Refund Seller Username Input | Functional | After entering the buyer’s username, the user must be prompted for the seller’s username. |  | Project Description, Page 6 |
| 11.3.1 | Refund Seller Must Exist Constraint | Functional | The seller must be a current user that exists. |  | Project Description, Page 6 |
| 11.4 | Refund Credit Transfer Input | Functional | After entering the seller’s username the user must be prompted for the amount of credit to transfer. |  | Project Description, Page 6 |
| 11.4.1 | Valid Credit Format Constraint | Functional | The dollar amount for the credit must have the correct format - it must contain at least one digit followed by a decimal and must contain only two digits after the decimal. |  | Appendix A Table 1 Item 14 |
| 11.5 | Refund Credit Transfer Operation | Functional | The amount of credit specified by the user must be transferred from the seller’s credit balance to the buyer’s credit balance. |  | Project Description, Page 6 |
| 11.5.1 | Only Credit Refunded | Functional | The amount of tickets sold is not replaced, only the amount of credit is refunded. |  | Appendix A Table 1 Item 7 |
| 11.6 | User Maximum Credit | Functional | The maximum credit that a user can have at any time must be 999,999.99 or less. |  | Appendix A Table 1 Item 5 |
| 11.7 | Save Refund Transaction to the Daily Transaction File | Functional | After the refund transaction has been executed by the user the refund transaction must save this information to the daily transaction file. |  | Appendix A Table 1 Item 7 |
| 12 | Add Credit Transaction | Functional | Entering “addcredit” into the input must begin the process of issuing credit to a user account. |  | Project Description Page 6 |
| 12.1 | Add Credit Transaction Privileges | Functional | When logged in as a privileged user, the addcredit transaction must prompt the user for the amount of credit to add and the username of the account. |  | Project Description, Page 6 |
| 12.1.1 | Privileged Add Credit Input Amount | Functional | After executing the addcredit transaction as a privileged user, the user must be prompted for the amount of credit to add. |  | Project Description, Page 6 |
| 12.1.2 | Valid Credit Format Constraint | Functional | The dollar amount for the credit must have the correct format - it must contain at least one digit followed by a decimal and must contain only two digits after the decimal. |  | Appendix A Table 1 Item 14 |
| 12.1.3 | Add Credit Username Input | Functional | After the user has entered the amount of credit to add, the user must be prompted for the username of the account to which credit is being added. |  | Project Description, Page 6 |
| 12.1.4 | Add Credit Username Must Exist Constraint | Functional | The username specified for the addcredit transaction must be an existing username in the system. |  | Project Description, Page 6 |
| 12.2 | Non-Privileged Add Credit Input Amount | Functional | When logged in as a non-privileged user, the addcredit transaction must prompt the user for the amount of credit. |  | Project Description, Page 6 |
| 12.3 | Add Credit Per-Session Maximum | Functional | The amount of credit that can be added to an account in a single login session must not exceed $1000.00. |  | Project Description, Page 6 |
| 12.4 | User Maximum Credit | Functional | The maximum credit that a user can have at any time must be 999,999.99 or less. | This is a backend requirement, not a frontend. | Project Description, Page 4  Appendix A Table 1 Item 5 |
| 12.5 | Save Add Credit Transaction to the Daily Transaction File | Functional | After the addcredit transaction has been executed by the user the addcredit transaction must be saved to the daily transaction file. |  | Appendix A Table 1 Item 11 |
| 13 | Current User Accounts File Format | Functional | The current user accounts file must only be accepted if it conforms to the correct file format.  The program must report an error if the file is corrupted and must not crash. |  | Project Description, Page 9  Appendix A Table 1 Item 21 |
| 13.1 | User Accounts File Length Constraint | Non-Functional | Every line in the current user accounts file must be exactly 28 characters in length (plus newline). |  | Project Description, Page 9 |
| 13.2 | Username Format | Non-Functional | Every current user account file entry must start with the username left justified. |  | Project Description, Page 9 |
| 13.2.1 | Username Length Constraint | Non-Functional | The username must not be more than 15 characters in length. |  | Project Description, Page 9  Appendix A Table 1 Item 24 |
| 13.2.2 | Username Character Constraint | Non-Functional | The username must only contain alphanumeric characters or underscores. |  | Appendix A Table 1 Item 23 |
| 13.3 | User Type Format | Non-Functional | The user type must be exactly two characters and must be one of the following identifiers:  AA (admin)  FS (full-standard)  BS (buy-standard)  SS (sell-standard) |  | Project Description, Page 9 |
| 13.4 | User Credit Format | Non-Functional | Every current user account file entry must end with the available credit as the last entry (plus the newline). |  | Project Description, Page 9 |
| 13.4.1 | User Credit Length Constraint | Non-Functional | The user credit must not be more than 9 characters. |  | Project Description, Page 9 |
| 13.4.2 | User Credit Monetary Value Constraint | Non-Functional | If the credit numeric field represents a monetary value and the value is in dollars “.00” must be appended to the end of the value. |  | Project Description, Page 9 |
| 13.5 | Alphabetic Fields Left Justified Filled Constraint | Non-Functional | Alphabetic fields must be left justified and must be filled with spacing characters “\_” up to the maximum length of the event name entry. |  | Project Description, Page 9 |
| 13.6 | Numeric Fields Right Justified Filled Constraint | Non-Functional | Numeric fields must be right justified and must be filled with zeros (e.g. 0000) for unused numeric fields. |  | Project Description, Page 9 |
| 13.7 | User Accounts File Separator Format Constraint | Non-Functional | A separator between the username (UUUUUUUU), account type (TT) and available credit (CCCCCCC) must be specified and must be the “\_” character. |  | Project Description, Page 9 |
| 13.8 | Unused Alphabetic Fields Constraint | Non-Functional | Unused alphabetic fields must be filled with the spacing character “\_”. |  | Project Description, Page 9 |
| 13.9 | Unused Numeric Fields Constraint | Non-Functional | Unused numeric fields must be filled with zeros “0”. |  | Project Description, Page 9 |
| 13.10 | End of File Format | Non-Functional | The current user accounts file must end with a special user named “END” with an empty user type and empty credit field. |  | Project Description, Page 9 |
| 14 | Available Tickets File Format | Functional | The available tickets file must only be accepted if it conforms to the correct file format.  If the file is corrupted an error must be reported. The program must never crash even if the file is corrupted. |  | Appendix A Table 1 Item 21 |
| 14.1 | Available Tickets File Length Constraint | Non-Functional | Every line in the available tickets file must be exactly 45 characters in length (plus newline). |  | Project Description, Page 10 |
| 14.2 | Event Name Format | Non-Functional | Every available ticket file entry must start with the event name left justified. |  | Project Description, Page 10 |
| 14.2.1 | Event Name Length Constraint | Non-Functional | The event must not be more than 19 characters in length. |  | Project Description, Page 10  Appendix A Table 1 Item 15 |
| 14.2.2 | Event Name Character Constraint | Non-Functional | The event name must only contain alphanumeric characters and punctuation. | The event name must only contain alphanumeric characters and punctuation. | Appendix A Table 1 Item 22 |
| 14.3 | Seller Username Format | Non-Functional | The event name must be directly followed by the seller’s name separated by a spacing character “\_”. |  | Project Description, Page 10 |
| 14.3.1 | Seller Username Length Constraint | Non-Functional | The seller’s username must not be more than 15 characters in length. |  | Project Description, Page 9  Appendix A Table 1 Item 24 |
| 14.3.2 | Seller Username Character Constraint | Non-Functional | The username must only contain underscores, letters and numbers. |  | Appendix A Table 1 Item 22 |
| 14.4 | Number of Tickets Format | Non-Functional | The seller’s username must be directly followed by the number of tickets for sale separated by a spacing character “\_”. |  | Project Description, Page 10 |
| 14.4.1 | Number of Tickets Length Constraint | Non-Functional | The number of tickets for sale filed must not be more than 3 characters in length. |  | Project Description, Page 10 |
| 14.4.2 | Number of Tickets Value Constraint | Non-Functional | The number of tickets for sale must be a decimal value. |  | Project Description, Page 10 |
| 14.5 | Ticket Price Format | Non-Functional | Every available tickets file entry must end with the price per ticket as the last entry (plus the newline). |  | Project Description, Page 10 |
| 14.5.1 | Ticket Price Length Constraint | Non-Functional | The user credit must not be more than 6 characters. |  | Project Description, Page 10 |
| 14.5.2 | Ticket Price Monetary Value Constraint | Non-Functional | If the ticket price numeric field represents a monetary value and if the value is in dollars “.00” must be appended to the end of the value. |  | Project Description, Page 10 |
| 14.6 | Unused Alphabetic Fields Constraint | Non-Functional | Unused alphabetic fields must be filled with the spacing character “\_”. |  | Project Description, Page 10 |
| 14.7 | Unused Numeric Fields Constraint | Non-Functional | Unused numeric fields must be filled with zeros “0”. |  | Project Description, Page 10 |
| 14.8 | Alphabetic Fields Left Justified Filled Constraint | Non-Functional | Alphabetic fields must be left justified and must be filled with spacing characters “\_” up to the maximum length of the event name entry. |  | Project Description, Page 10 |
| 14.9 | Numeric Fields Right Justified Filled Constraint | Non-Functional | Numeric fields must be right justified and must be filled with zeros (e.g. 0000) for unused numeric fields. |  | Project Description, Page 10 |
| 14.10 | End of File Format | Non-Functional | The current user accounts file must end with a special user named “END” with an empty user type and empty credit field. |  | Project Description, Page 10 |
| 15 | Daily Transaction File Format  Create, Delete, Addcredit, End of Session Transactions | Functional | The daily transaction file must only be accepted if it conforms to the correct file format.  If the file is corrupted an error must be reported. The program must never crash even if the file is completely corrupted. |  | Appendix A Table 1 Item 21 |
| 15.1 | Transaction Code Format | Non-Functional | Every daily transaction file entry must start with the transaction code.  The transaction code must be exactly two characters and must be one of the following identifiers:  01-create  02-delete  06-addcredit  00-end of session |  | Project Description, Page 7 |
| 15.2 | Username Format | Non-Functional | The transaction code must be directly followed by the user’s name separated by a spacing character “\_”. |  | Project Description, Page 7 |
| 15.2.1 | Username Length Constraint | Non-Functional | The username must not be more than 15 characters in length. |  | Project Description, Page 7  Appendix A Table 1 Item 24 |
| 15.2.2 | Username Character Constraint | Non-Functional | The username must only contain underscores, letters and numbers. |  | Appendix A Table 1 Item 22 |
| 15.3 | User Type Format | Non-Functional | The user type must be exactly two characters and must be one of the following identifiers:  AA (admin)  FS (full-standard)  BS (buy-standard)  SS (sell-standard) |  | Project Description, Page 7 |
| 15.4 | User Credit Format | Non-Functional | Every current user account file entry must end with the available credit as the last entry (plus the newline). |  | Project Description, Page 7 |
| 15.4.1 | User Credit Length Constraint | Non-Functional | The user credit must not be more than 9 characters. |  | Project Description, Page 7 |
| 15.4.2 | User Credit Monetary Value Constraint | Non-Functional | If the credit numeric field represents a monetary value and the value is in dollars “.00” must be appended to the end of the value. |  | Project Description, Page 7 |
| 16 | Daily Transaction File Format refund Transactions | Functional | The daily transaction file must only be accepted if it conforms to the correct file format.  If the file is corrupted an error must be reported. The program must never crash even if the file is completely corrupted. |  | Appendix A Table 1 Item 21 |
| 16.1 | Transaction Code Format | Non-Functional | Every daily transaction file entry must start with the transaction code.  The transaction code must be exactly two characters and must be one of the following identifiers:  05-refund |  | Project Description, Page 7 |
| 16.2 | Buyer Username Format | Non-Functional | The transaction code must be directly followed by the Buyer’s username separated by a spacing character “\_”. |  | Project Description, Page 7 |
| 16.2.1 | Username Length Constraint | Non-Functional | The username must not be more than 15 characters in length. |  | Project Description, Page 7  Appendix A Table 1 Item 24 |
| 16.2.2 | Username Character Constraint | Non-Functional | The username must only contain underscores, letters and numbers. |  | Appendix A Table 1 Item 22 |
| 16.3 | Seller Username Format | Non-Functional | The event name must be directly followed by the seller’s name separated by a spacing character “\_”. |  | Project Description, Page 7 |
| 16.3.1 | Username Length Constraint | Non-Functional | The seller’s username must not be more than 15 characters in length. |  | Project Description, Page 9  Appendix A Table 1 Item 24 |
| 16.3.2 | Username Character Constraint | Non-Functional | The username must only contain underscores, letters and numbers. |  | Appendix A Table 1 Item 22 |
| 16.4 | User Credit Format | Non-Functional | Every current user account file entry must end with the available credit as the last entry (plus the newline). |  | Project Description, Page 7 |
| 16.4.1 | User Credit Length Constraint | Non-Functional | The user credit must not be more than 9 characters. |  | Project Description, Page 7 |
| 16.4.2 | User Credit Monetary Value Constraint | Non-Functional | If the credit numeric field represents a monetary value and the value is in dollars “.00” must be appended to the end of the value. |  | Project Description, Page 7 |
| 17 | Daily Transaction File Format Sell, Buy Transactions | Functional | The daily transaction file must only be accepted if it conforms to the correct file format.  If the file is corrupted an error must be reported. The program should never crash even if the file is completely corrupted. |  | Appendix A Table 1 Item 21 |
| 17.1 | Transaction Code Format | Non-Functional | Every daily transaction file entry must start with the transaction code.  The transaction code must be exactly two characters and must be one of the following identifiers:  03-sell  04-buy |  | Project Description, Page 8 |
| 17.2 | Event Name Format | Non-Functional | The transaction code must be directly followed by the event name separated by a spacing character “\_”. |  | Project Description, Page 8 |
| 17.2.1 | Event Name Length Constraint | Non-Functional | The event must not be more than 19 characters in length. |  | Project Description, Page 8  Appendix A Table 1 Item 15 |
| 17.2.2 | Event Name Character Constraint | Non-Functional | The event name must only contain alphanumeric characters and punctuation. | The event name must only contain alphanumeric characters and punctuation. |  |
| 17.3 | Seller Username Format | Non-Functional | The event name must be directly followed by the seller’s name separated by a spacing character “\_”. |  | Project Description, Page 8 |
| 17.3.1 | Seller Username Length Constraint | Non-Functional | The seller’s username must not be more than 15 characters in length. |  | Project Description, Page 9  Appendix A Table 1 Item 24 |
| 17.3.2 | Seller Username Character Constraint | Non-Functional | The username must only contain underscores, letters and numbers. |  | Appendix A Table 1 Item 22 |
| 17.4 | Number of Tickets Format | Non-Functional | The seller’s username must be directly followed by the number of tickets for sale separated by a spacing character “\_”. |  | Project Description, Page 8 |
| 17.4.1 | Number of Tickets Length Constraint | Non-Functional | The number of tickets for sale must not be more than 3 characters in length. |  | Project Description, Page 8 |
| 17.4.2 | Number of Tickets Value Constraint | Non-Functional | The number of tickets for sale must be a decimal value. |  | Project Description, Page 8 |
| 17.5 | Ticket Price Format | Non-Functional | Every available tickets file entry must end with the price per ticket as the last entry (plus the newline). |  | Project Description, Page 8 |
| 17.5.1 | Ticket Price Length Constraint | Non-Functional | The ticket price must not be more than 6 characters. |  | Project Description, Page 8 |
| 17.5.2 | Ticket Price Monetary Value Constraint | Non-Functional | If the ticket price numeric field represents a monetary value and if the value is in dollars “.00” must be appended to the end of the value. |  | Project Description, Page 8 |
| 18 | Daily Transaction File Constraints | Non-Functional | The daily transaction file must have alphabetic values left justified and numeric values right justified. |  | Project Description, Page 8-10 |
| 18.1 | Alphabetic Fields Left Justified Filled Constraint | Non-Functional | Alphabetic fields must be left justified and must be filled with spacing characters “\_” up to the maximum length of the event name entry. |  | Project Description, Page 10 |
| 18.2 | Numeric Fields Right Justified Filled Constraint | Non-Functional | Numeric fields must be right justified and must be filled with zeros (e.g. 0000) for unused numeric fields. |  | Project Description, Page 8-10 |
| 18.3 | User Accounts File Separator Format Constraint | Non-Functional | A separator between the each section must be specified and must be the “\_” character. |  | Project Description, Page 8-10 |
| 18.4 | Unused Alphabetic Fields Constraint | Non-Functional | Unused alphabetic fields must be filled with the spacing character “\_”. |  | Project Description, Page 8-10 |
| 18.5 | Unused Numeric Fields Constraint | Non-Functional | Unused numeric fields must be filled with zeros “0”. |  | Project Description, Page 8-10 |
| 18.6 | End of File Format | Non-Functional | The current user accounts file must end with a special user named “END” with an empty user type and empty credit field. |  | Project Description, Page 8-10 |
| 19 | Files Specified as Program Command Line Arguments | Functional | The daily transaction, current user accounts, and available tickets files must be provided as command line arguments at program start. |  | Appendix A Table 1 Item 11 |
| 20 | File Name and Extension Format | Non-Functional | The order does not matter; the filename and extension must be specified by us in our implementation.  Each of the filenames and extensions must be explained in the report. |  | Appendix A Table 1 Item 11 |
| 20 | Read Available Tickets File | Functional | The front-end application must read the Available Tickets file during startup execution.  If the file is corrupted an error must be reported. The program must never crash even if the file does not exist. |  | Appendix A Table 1 Item 21 |
| 21 | Read Current User Accounts File | Functional | The front-end application must read the Current User Accounts file during startup execution.  If the file is corrupted an error must be reported. The program must never crash even if the file does not exist. |  | Appendix A Table 1 Item 21 |
| 22 | Front End Application Termination | Functional | The front-end application must terminate when the quit command has been executed. |  | Appendix A Table 1 Item 25 |
| 22.1 | User Session Ended Constraint | Functional | The front end session must have ended—no user must be logged into the system. |  | Appendix A Table 1 Item 25 |

## Back End Requirements

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Item #** | **Requirement Name** | **Requirement Category** | **Requirement Description** | **Assumptions** | **Citation** |
| 1 | Programming Language | Non-Functional | The back-end must be written in Java. |  | Project Description, pg. 2 |
| 2 | Applying Transactions | Functional | The back-end must read files produced by the front-end and generate output files that are used by the front-end for the following day’s execution. |  | Project Description, pg. 11 |
| 2.1 | Merged Daily Transaction File | Functional | The back-end must read in the merged daily transaction file. |  | Project Description, pg. 11 |
| 2.1.1 | Merged DTF Concatenation | Functional | The merged daily transaction file must concatenate any number of daily transaction files output from the front-end. |  | Project Description, pg. 12 |
| 2.1.1.1 | Merged DTF Concatenation Order | Functional | The merged daily transaction file must concatenate the previous DTF files in chronological order. | Assumed chronological order. | Project Description, pg. 12  Appendix B table 1 tuple 1 |
| 2.1.1.2 | Merged DTF File Renaming | Functional | After merging, daily transaction files must be renamed in order to mark them as archived (using their datestamp in the file name). |  | Appendix B table 1 tuple 8 |
| 2.1.2 | Merged DTF Formatting Constraint | Functional | The merged daily transaction file must end with an empty entry, consisting of a line with a 00 transaction code. |  | Project Description, pg. 12 |
| 2.2 | Updating Current User Accounts File | Functional | The back-end must read in the current user accounts file, and update the entries based on the transactions which are processed from the daily transaction file input. |  | Project Description, pg. 11 |
| 2.2.1 | Current User Accounts File Entry Constraint | Functional | All entries in the updated current user accounts file must have unique user name titles. |  | Project Description, pg. 11 |
| 2.2.2 | Updated Current User Accounts File Output | Functional | The updated current user accounts file output must overwrite the previous current user accounts file. | Assumed that the newly created CUA file must overwrite the old one. | Appendix B table 1 tuple 2 |
| 2.2.3 | Updated CUA State Constraint (Credit) | Functional | The updated current user accounts file must not contain credit values outside acceptable parameters. |  | Project Description, pg. 11 |
| 2.2.3.1 | Credit Overflow Constraint | Functional | The maximum credit value of any user must not exceed 999999.99 | Requirement 11.6 from the front-end is carried over. | Appendix B table 1 tuple 3 |
| 2.2.3.2 | Credit Underflow Constraint | Functional | The credit value of any user must always be greater than or equal to zero. |  | Appendix B table 1 tuple 4 |
| 2.2.4 | Updated CUA State Constraint (User Deletion) | Functional | The back-end must not permit the deletion of users who do not exist within the CUA file. |  | Appendix B table 1 tuple 5 |
| 2.3 | Updating Available Tickets File | Functional | The back-end must read in the available tickets file, and update the entries based on the transactions which are processed from the daily transaction file input. |  | Project Description, pg. 11 |
| 2.3.1 | Available Tickets File Entry Underflow Constraint | Functional | All entries in the updated available tickets file must not have negative values for the number of tickets left. |  | Project Description, pg. 11 |
| 2.3.2 | Available Tickets File Entry Overflow Constraint | Functional | All entries in the updated available tickets file must not exceed 100 values for the number of tickets for sale. |  | Project Description, pg. 5  Appendix A table 1 tuple 6 |
| 2.3.3 | Updated Available Tickets File Output | Functional | The updated available tickets file output must overwrite the previous available tickets file. | Assumed that the newly created ATF must overwrite the old one. | Appendix B table 1 tuple 2 |
| 3 | Failed Constraint Log Production | Functional | The back-end must produce a log of transactions that cause business constraints to fail. |  | Project Description, pg. 12 |
| 3.1 | Failed Constraint Log Console Output | Functional | The failed constraint log must output to the console. |  | Project Description, pg. 12 |
| 3.1.1 | Failed Constraint Log Output Format | Functional | The failed constraint log must record errors in the form:  ERROR: <msg> |  | Project Description, pg. 12 |
| 3.1.2 | Failed Constraint Log Output Message Format | Functional | The failed constraint log error messages (denoted by <msg>) must first contain business constraint error information, followed by diagnostic information. |  | Project Description, pg. 12 |
| 3.2 | Fatal Error Console Output | Functional | In circumstances where the program must terminate due to a failed constraint, a fatal error must be output to the console. |  | Project Description, pg. 11, 12 |
| 3.2.1 | Fatal Error Constraint Log Output Format |  | The fatal error log must record errors in the form:  ERROR: <msg> |  | Project Description, pg. 12 |
| 3.2.2 | Fatal Error Log Output Message Format | Functional | The fatal error log error messages (denoted by <msg>) must first contain business constraint error information, followed by diagnostic information, and the file that caused the error. |  | Project Description, pg. 12 |
| 3.3 | Failed Constraint Log Output Messages (Business Constraints) | Functional | All failed constraint log entries must contain the type and description of the failed business constraint error. |  | Project Description, pg. 12 |
| 3.4 | Failed Constraint Log Output Messages (Diagnostic Information) | Functional | All failed constraint log entries must contain a record of the transaction which caused the error. |  | Project Description, pg. 12 |
| 3.5 | Fatal Error Log Output Messages (File Information) | Functional | All fatal error log entries must contain a record of the file that caused the error. |  | Project Description, pg. 12 |
| 3.6 | Failed Constraints Evaluation | Functional | All transactions which fail constraints must not be applied to output files generated by the back-end. |  | Appendix B table 1 tuple 6 |
| 4 | Read Old Available Tickets File | Functional | The back-end application must validate the old available tickets file during startup execution. |  | Project Description, pg. 11 |
| 4.1 | ATF Bad Input Termination | Functional | The back-end application must immediately terminate if there is bad input in the file. |  | Project Description, pg. 11 |
| 4.2 | ATF Bad Input Error Display | Functional | If there is bad input in the ATF, a fatal error must be displayed in the terminal. |  | Project Description, pg. 11 |
| 5 | Read Old Current User Accounts File | Functional | The back-end application must validate the old current user accounts file during startup execution. |  | Project Description, pg. 11 |
| 5.1 | CUA Bad Input Termination | Functional | The back-end application must immediately terminate if there is bad input in the file. |  | Project Description, pg. 11 |
| 5.2 | CUA Bad Input Error Display | Functional | If there is bad input in the CUA file, a fatal error must be displayed in the terminal. |  | Project Description, pg. 11 |