*[The Test Case ID should be unique. In addition, the name of each Test Case should reflect the intent of the test case, ideally expressed as a Boolean condition.]*

**<1> - <Scan Book>:**

Description: [Describe the logical condition that the Test Case evaluates. Include the expected result.]

Pre-conditions: [List conditions that must be true before this Test Case can start.]

Post-conditions: [List conditions that should be true when this Test Case ends.]

Data required: [Identify the type of data required for this Test Case.]

**<1> - <Swipe Card member exist not restricted>:**

Description: Evaluates the systems ability to scan a card and act according if the user is not restricted. result should be that a member is scanned and a loan started. The scanning screen, user details and users existing loans should all be displayed

Pre-conditions:

* Card Scanner must be present
* Member must be in the system

Post-conditions:

* Scanning panel is displayed
* complete and cancel buttorns are enabled
* card reader disabled
* scanner is enabled
* borrower details are displayed
* existing loans are displayed
* scann count initialized as number of existing loans
* existing fine message displayed if relevent

Data required:

* Member object
* member fineAmount
* member loans.
* member name
* memebr number
* member status

**<2> - <Swipe Card member exist restricted>:**

Description: Evaluates the systems ability to scan a card and act according if the user is restricted. //result should be that a member is scanned. The Borrowing restricet message, restricted panel, user details and users existing loans should all be dispayed. the relevant message for why the borrower is restricted should be displayed.

Pre-conditions:

* Card Scanner must be present
* Member must be in the system
* member is not null.

Post-conditions:

* Restricted panel is displayed
* cancel buttorns are enabled
* card reader disabled
* scanner is disabled
* borrower details are displayed
* existing loans are displayed
* overdue message displayed if relevant.
* existing fine message displayed if relevant
* borrowing restricted message displayed

Data required:

* Member object
* member fineAmount
* member loans.
* member name
* memebr number
* member status
* memberDAO object

**<3> <Swipe Card member does not exist>**

Description: Evaluates the systems ability to scan a card and act according if the user does not exist. //result should be that a member is scanned and a loan started. The scanning screen, user details and users existing loans should all be displayed

Pre-conditions:

* Card Scanner must be present.
* member is null.

Post-conditions:

* memeber not found message should be displayed.
* cardReader enabled
* scanner disabled
* cancel button enabled

Data required:

* Member object
* memberDAO object

**<4> - <Scan Book not found>:**

Description: Evaluates the systems ability to scan a book and respond if the book is not found. result should be that the book is scanned and the system return a message stating the book was not found.

Pre-conditions:

* Book scanner must be present.
* member must be in system
* Card must have been scanned.
* book not in system
* BorrowBookCTl class exists
* boorrowBookCTL class added as listener to scanner

Post-conditions:

* BorrowBookUI is displayed
* scanning panel of borrowBookUI displayed
* cancel button enabled
* cardReader is disabled
* scanner is enabled
* book not found error message displayed.

Data required:

* Member object
* Loan object
* Book object
* book state
* book id

**<5> - <Scan Book not available>:**

Description: Evaluates the systems ability to scan a book and respond if the book is not currently avaible. result should be that the book is scanned and the system return a message stating the book was not available.

Pre-conditions:

* Book scanner must be present.
* member must be in system
* Card must have been scanned.
* book state not available
* BorrowBookCTl class exists
* boorrowBookCTL class added as listener to scanner

Post-conditions:

* BorrowBookUI is displayed
* scanning panel of borrowBookUI displayed
* cancel button enabled
* cardReader is disabled
* scanner is enabled
* book not available error message displayed.

Data required:

* Member object
* Loan object
* Book object
* book state
* book id

**<6> - <Scan Book already scanned>:**

Description: Evaluates the systems ability to scan a book and respond if the book is already scanned as part of the loan. result should be that the book is scanned and the system return a message stating the book has already been scanned.

Pre-conditions:

* Book scanner must be present.
* member must be in system
* Card must have been scanned.
* book state available
* bookList contains book
* BorrowBookCTl class exists
* boorrowBookCTL class added as listener to scanner

Post-conditions:

* BorrowBookUI is displayed
* scanning panel of borrowBookUI displayed
* cancel button enabled
* cardReader is disabled
* current book details displayed
* pending loan list displayed
* scanner is enabled
* book already scanned error message displayed.

Data required:

* Member object
* Loan object
* Book object
* book state
* book id
* bookList
* pendingLoanList
* pendingLoan

**<7> - <Scan Book scan count < loan limit>:**

Description: Evaluates the systems ability to scan a book and respond if the book is scanned correctly and added to the loanlist. result should be that the book is scanned and the system adds the book to the pending loan list.

Pre-conditions:

* Book scanner must be present.
* member must be in system
* Card must have been scanned.
* book state available
* bookList contains book
* BorrowBookCTl class exists
* boorrowBookCTL class added as listener to scanner

Post-conditions:

* BorrowBookUI is displayed
* scanning panel of borrowBookUI displayed
* cancel button enabled
* cardReader is disabled
* current book details displayed
* pending loan list displayed
* scanner is enabled
* scanCount incremented by 1
* new pending loan created
* pending loan added to pending loan list

Data required:

* Member object
* Loan object
* Book object
* book state
* book id
* bookList
* pendingLoanList
* pendingLoan

**<8> - <Scan Book scan count == loan limit>:**

Description: Evaluates the systems ability to scan a book and respond if the book scanned equals the loan limit and returns a message saying that. result should be that the book is scanned and the system completes the loan.

Pre-conditions:

* Book scanner must be present.
* member must be in system
* Card must have been scanned.
* book state available
* bookList contains book
* BorrowBookCTl class exists
* boorrowBookCTL class added as listener to scanner

Post-conditions:

* BorrowBookUI is displayed
* confirmingloans panel of borrowBookUI is displayed
* cancel button enabled
* reject button enabled
* cardReader is disabled
* current book details displayed
* final pending loan list displayed
* scanner is disabled
* scanCount incremented by 1

Data required:

* Member object
* Loan object
* Book object
* book state
* book id
* bookList
* pendingLoanList
* pendingLoan

**<9> - <Complete scan>:**

Description: Evaluates the systems ability to complete scan, the result should be that the book scanning stage of the loan process is completed.

Pre-conditions:

* Book scanner must be present.
* member must be in system
* Card must have been scanned.
* cardReader present.
* PendingLoanList exists
* BorrowBookCTl class exists

Post-conditions:

* BorrowBookUI is displayed
* confirmingloans panel of borrowBookUI is displayed
* list of pending loand displayed
* cancel button enabled
* cardReader is disabled
* scanner is disabled

Data required:

* pendingLoanList

**<10> - <Confirm Loans>:**

Description: Evaluates the systems ability to confirm a loan, the result should be that the loan is confirmed and added the the members file.

Pre-conditions:

* Book scanner must be present.
* member must be in system
* Card must have been scanned.
* cardReader present.
* PendingLoanList exists
* BorrowBookCTl class exists

Post-conditions:

* Main Menu is displayed
* all pending loans are committed and recorded
* Loan slip of committed loans is printed
* cardReader is disabled
* scanner is disabled

Data required:

* pendingLoanList

**<11> - <Reject Loans>:**

Description: Evaluates the systems ability to reject a loan, the result should be that the loan is rejected.

Pre-conditions:

* Book scanner must be present.
* member must be in system
* Card must have been scanned.
* cardReader present.
* PendingLoanList exists
* BorrowBookCTl class exists

Post-conditions:

* Main Menu is displayed
* BorrowBookUI is displayed
* Scanning panel of BorrowBookUI is displayed
* borrower details displayed
* existing loan details displayed
* pending loan list is empty
* scan count equals number of existing loans
* cancel button enabled
* Loan slip of committed loans is printed
* cardReader is disabled
* scanner is enabled

Data required:

* Member object
* Loan object
* bookList
* pendingLoanList
* pendingLoan