**TICKETBOOTH:**
**TEST CASE 001**

| Test Title | Valid Log-In | Test Priority | High |
|---|---|---|---|
| Design by | Sara Aldhaheri | Design date | 10/12/19 |
| Test Executed by | Russell Coke | Execution date | 11/12/19 |

**Description**

This test case tests Users ability to use the log-in functionality of the system when an email and password is entered. This test is not applicable for System Administrators.

**Objective**

1. To effectively login to the system.

**Pre-conditions**

1. The system is running.
2. Account is existent and active.

| Step | Test Steps | Test Data | Expected Results | Actual Results | Status (P/F) | Notes |
|---|---|---|---|---|---|---|
| 1 | Navigate to home / login screen | | Home screen appears with fields for username and password | Page appears as expected | P | |
| 2 | Provide email | Valid email | Only characters can be entered | Any ASCII character can be entered | P | |
| 2a | Provide email | Invalid email | Error message: “Unregistered Account” | Error message appears | P | |
| 3 | Provide password | Valid Password | Only characters can be entered and are hidden | As expected | P | |
| 3a | Provide password | Invalid Password | Error message: “Incorrect Password” | Error message appears | P | |
| 4 | Click on the Login button | Valid credentials | User is logged in | As expected | P | |

| Post-conditions |
| --- |
| 1. The system checks the database and validates the user.<br>2. The user is successfully logged in and redirected to the home screen of the program, with the appropriate features for their system privilege level unlocked. |

**TICKETBOOTH:**
**TEST CASE 002**

| Test Title | Purchase Tickets | Test Priority | High |
|---|---|---|---|
| Design by | Sara Aldhaheri | Design date | 10/12/19 |
| Test Executed by | Russell Coke | Execution date | 11/12/19 |

| Description |
|---|
| This use case is to enable Customers to buy a maximum of 5 tickets to an Event. |

| Objective |
|---|
| 1. Allow Customers to make purchases<br>2. Update System of purchases made |

| Pre-conditions |
|---|
| 1. Customer is logged on<br>2. System is running<br>3. An event has been selected<br>4. Event is not fully booked |

| Step | Test Steps | Test Data | Expected Results | Actual Results | Status (P/F) | Notes |
|---|---|---|---|---|---|---|
| 1 | Enter number of tickets | Value | Only up to 5 tickets can be chosen | As expected | P | |
| 2 | Confirm purchase | Confirm button | Event seat capacity is updated, purchase is confirmed, redirect to Events page | As expected | P | A checkout page should be added |

| Post-conditions |
|---|
| 1. Update Seat Capacity<br>2. Send Confirmation Email with Receipt<br>3. Update Customer Transaction History |

**TICKETBOOTH:**
**TEST CASE 003**

| Test Title | Search Catalog for Event | Test Priority | Medium |
|---|---|---|---|
| Design by | Sara Aldhaheri | Design date | 10/12/19 |
| Test Executed by | Russell Coke | Execution date | 11/12/19 |

| Description |
|---|
| This test focuses on allowing Users to search database for UPCOMING Events. Users will be able to filter their searches by Event type / name / date / location. |

| Objective |
|---|
| 1. To effectively search system database for specific Event |

| Pre-conditions |
|---|
| 1. User is logged in<br>2. User is on Home Page<br>3. Search Bar displays text<br>4. Location, Time, Date displays text |

| Step | Test Steps | Test Data | Expected Results | Actual Results | Status (P/F) | Notes |
|---|---|---|---|---|---|---|
| 1 | Type text in Search Bar | Existing Event | Matching results displayed | Matching results displayed | P | |
| 1a | Type text in Search Bar | Non existent Event | No match found | No results displayed | P | |
| 2 | Filter by date / type / location | Filter | Events on matching date / type / location displayed | Multiple filter fields works as expected | P | |

| Post-conditions |
|---|
| 1. Events that match the search will be displayed |

**TICKETBOOTH:**
**TEST CASE 004**

| Test Title | Send Refund Request | Test Priority | Medium |
|---|---|---|---|
| Design by | Sara Aldhaheri | Design date | 10/12/19 |
| Test Executed by | Russell Coke | Execution date | 11/12/19 |

| Description |
|---|
| This test is designed to ensure the Refund Request option available to Customers and Wholesalers when an event has been edited works properly. |

| Objective |
|---|
| 1. Send refund to Customers who have sent a Request<br>2. Update transaction history |

| Pre-conditions |
|---|
| 1. User is logged in<br>2. User is on Transaction History<br>3. Event Organizer edited Event that Customer / Wholesaler bought tickets for |

| Step | Test Steps | Test Data | Expected Results | Actual Results | Status (P/F) | Notes |
|---|---|---|---|---|---|---|
| 1 | “Request Refund” button appear for Event which edited by Event Organizer | Button | Button changes to “Refund Pending” | Refund Pending shows | P | |
| | | | Request sent to System Administrator | Request is sent | P | |
| | | | | | | |

| Post-conditions |
|---|
| 1. Refund Confirmation email sent to Customer or Wholesaler<br>2. Update Seat Capacity |

**TICKETBOOTH:**
**TEST CASE 005**

| Test Title | Approve Refund Request | Test Priority | Medium |
|---|---|---|---|
| Design by | Sara Aldhaheri | Design date | 10/12/19 |
| Test Executed by | Russell Coke | Execution date | 11/12/19 |

| Description |
|---|
| This test is designed to ensure the Approval for the Refund Request sent by a Customer can be executed by the System Administrator. |

| Objective |
|---|
| 1. Send refund to Customers who have sent a Request<br>2. Update transaction history<br>3. Update Seat Capacity |

| Pre-conditions |
|---|
| 1. System Administrator is logged in<br>2. System Administrator is on Refund Requests page<br>3. Event Organizer edited Event that Customer / Wholesaler bought tickets for |

| Step | Test Steps | Test Data | Expected Results | Actual Results | Status (P/F) | Notes |
|---|---|---|---|---|---|---|
| 1 | “Approve Refund” button appear for Event which edited by Event Organizer | Button | Request is removed from screen | Request disappears from list | P | |
| | | | Approval Confirmation sent to customer | Email is sent | P | |
| | | | Seat Capacity increases | Number of tickets increases by amount the customer returned | P | |

| Post-conditions |
|---|
| 1. Refund Confirmation email sent to Customer or Wholesaler<br>2. Update Seat Capacity |

**TICKETBOOTH:**
**TEST CASE 006**

| Test Title | Register Account | Test Priority | High |
|---|---|---|---|
| Design by | Sara Aldhaheri | Design date | 10/12/19 |
| Test Executed by | Russell Coke | Execution date | 11/12/19 |
| Description | | | |
| This test is responsible for creating a new account in the database. Accounts can be Customer / Wholesaler / Event Organizer. | | | |
| Objective | | | |
| 1. Create a new account to AccountCollection. | | | |
| Pre-conditions | | | |
| 1. System is running<br>2. Login page displayed | | | |

| Step | Test Steps | Test Data | Expected Results | Actual Results | Status (P/F) | Notes |
|---|---|---|---|---|---|---|
| 1 | Click sign up | Sign Up | Registration page opens | Page opens | P | |
| 2 | Fill required fields on the login page | | Text appears in fields | Text Appears | P | |
| 3a | Sign Up button clicked | All fields are filled | Account created. Customer logged in automatically | Customer is added to DB and logged in | P | |
| 3b | Sign Up button clicked | Missing field | Refreshes to indicate missing data | Error showing missing field | P | |
| 4 | System sends Confirmation email | | Email sent | No email sent | F | Investigating failure point currently, will be addressed in next testing round. |

| Post-conditions |
| --- |
| 1. Confirmation email sent to new User |

**TICKETBOOTH:**
**TEST CASE 007**

| Test Title | Edit Existing Event | Test Priority | Medium |
|---|---|---|---|
| Design by | Sara Aldhaheri | Design date | 10/12/19 |
| Test Executed by | Russell Coke | Execution date | 11/12/19 |

**Description**

This test focuses on allowing Event Organizers to edit information on their own UPCOMING Events. Event Organizers will be able to edit their details of their Event like type / name / date / location.

**Objective**

1. To effectively edit database on details for a specific Event

**Pre-conditions**

1. Event Organizer is logged in
2. Only Event Author (AKA Event Organizer) can edit their own Event.

| Step | Test Steps | Test Data | Expected Results | Actual Results | Status (P/F) | Notes |
|---|---|---|---|---|---|---|
| 1 | Click on Event to be edited | Event link | Event page open with field boxes filled with previous information | Page opens | P | |
| 2 | Type changes in field | | Changes appear in the fields | Field is initially populated with current data but is editable to accept new changes | P | |
| 3 | Click Edit Event | “Edit Event” button | New description appear in the Event page | Changes are reflected in the Events page | P | |
| 4 | System sends Event Changes email to Customers and Wholesaler | | Email sent | Event ticket holders are notified | P | |

| Post-conditions |
| --- |
| 1. Request Refund button opens up in Customer’s Transaction History<br>2. Customers / Wholesalers are notified via email of the Event Changes |

**TICKETBOOTH:**
**TEST CASE 008**

| Test Title | Cancel Event | Test Priority | High |
|---|---|---|---|
| Design by | Jahnae Miller | Design date | 10/12/19 |
| Test Executed by | Russell Coke | Execution date | 11/12/19 |

| Description |
|---|
| This test focuses on allowing Event Organizers to cancel their own Event |

| Objective |
|---|
| 1. To effectively cancel an Event from the system database |

| Pre-conditions |
|---|
| 1. Event active<br>2. Event Author (AKA Event Organizer) accessing Event |

| Step | Test Steps | Test Data | Expected Results | Actual Results | Status (P/F) | Notes |
|---|---|---|---|---|---|---|
| 1 | Navigate Active Events page | | List of active events appear | Events appear | P | |
| 1a | Click on Event to be cancelled | Event link | Event page opens | Page Opens | P | |
| 2 | Click Cancel Event | “Cancel” button | Event is deleted from the records, redirect back to Events page | Event removed, Events page reflects this deletion | P | |
| 3 | System sends Event Cancellation email to Customers and Wholesaler | | Email sent | Ticket holders are notified and refunded | P | |

| Post-conditions |
|---|
| 1. Event no longer appears on search page<br>2. Refunds automatically issued<br>3. Customers / Wholesalers are notified via email of the Event Cancellation |

4. Transaction History updated

**TICKETBOOTH:**
**TEST CASE 009**

| Test Title | Create Event | Test Priority | High |
|---|---|---|---|
| Design by | Sara Aldhaheri | Design date | 10/12/19 |
| Test Executed by | Jahnae Miller | Execution date | 11/12/19 |

**Description**

This test focuses on allowing Users to search database for UPCOMING Events. Users will be able to filter their searches by Event type / name / date / location.

**Objective**

1. To effectively create a new Event in the system database
2. To promote new events in the Events Page for Customers to View
3. To create a new Event page with descriptions and status

**Pre-conditions**

1. Event Organizer is logged in
2. Event Organizer is on Home Page

| Step | Test Steps | Test Data | Expected Results | Actual Results | Status (P/F) | Notes |
|---|---|---|---|---|---|---|
| 1 | Click Create Event button | Create Event button | Create New Event page is displayed | Page is displayed | P | |
| 2 | Type text in required field | | Information typed appear on screen | Information appears | P | |
| 3 | Upload Supporting Documents | | Documents upload | Document dialog box appears, selected document is uploaded | P | |
| 4 | Create New Event button | “Create New Event” button | Event is added to DB, Events page reflects this addition | Event is created and shows up on Events page | P | |

| Post-conditions |
| --- |
| 1. New Event found in database and Event Pages<br>2. Event Organizer associated to the Events they created. Only said Event Organizer have editing rights to their Event. Every other other will have viewing rights only. |

**TICKETBOOTH:**
**TEST PLAN 001**

*Introduction*

Ticketbooth is a web-based application that connects Event Organizers and Concert Goers. The new ticketing system will allow Event Organizers to perform all necessary process in creating / modifying / cancelling / boosting their events online. Customers can browse through all the upcoming Events and purchase tickets for them from the comforts of their home. The following is a test plan made for acceptance testing of the TicketBooth. These tests will be performed to ensure that the features of the systems conform exactly to the requirements given by the project specification. This tests

*Objectives*

- To ensure that the system is properly protected behind the log-in mechanism.
- To ensure that the log-in system works as intended.
- To effectively allow access to certain features according to account type.
- To ensure that the system provides an organized means to keep track of Events / Seat Capacity.
- To ensure that the system provides a user-friendly interface for Customers / Wholesalers / Event Organizers / System Administrators to better navigate the system for their information.
- To ensure that the system effectively create new Event(s).
- To effectively allow Event Organizers to cancel their Event(s)..
- To ensure that Event Organizers can edit their Event(s).
- To ensure that a Customer can purchase their tickets is only limited to five (5) tickets maximum.
- To ensure refunds can be successfully transferred upon request after amendments to an Event or cancellation of an Event.

*Test Items*

| Test Item Name | Version Number |
|---|---|
| The Log-In Subsystem | 0.1 |
| Customer Subsystem | 0.1 |
| Event Organizer Subsystem | 0.1 |
| System Administrator Subsystem | 0.1 |

*Features to be Tested*

Keys use cases of medium-to-high priority will be tested. These include the following units:

| Module Name | Actors | Description |
|---|---|---|
| Log-In | Customers, Wholesalers, Event Organizers, System Administrators | Any user with a valid username and password should be able to log in. Specific functionalities of the system should be unlocked after a user is |

| | | |
|---|---|---|
| | | logged in. Logging in should allow different users different privileges. |
| Purchase Tickets | Customers, Wholesalers | Customers and Wholesalers have access to this feature allowing them to purchase tickets to selected Events. Customers are able to purchase up to 5 tickets while Wholesalers can only purchases 6 or more tickets. Upon the completion of this use case, the number of available tickets is updated. |
| Search Catalog | Customers, Wholesalers, Event Organizers, System Administrators | This function allows Users to navigate through the Event page with ease using different filters. |
| Request Refund | Customers, Wholesalers | When an Event Organizer makes any changes to their Event, Customers and Wholesalers who have purchased tickets may be provided with the option to request a refund. When an Event is cancelled, Customers / Wholesalers are automatically refunded. |
| Approve Refund Request | System Administrators | This feature allows the System Administrator to approve a refund request by Customers / Wholesalers. |
| Register Account | Customers, Wholesalers, Event Organizers, System Administrators | Users are able to create a new account with TicketBooth through this feature. They may create an account as Customers / Wholesalers / Event Organizer. |
| Create Event | Event Organizers | Event Organizers can add a new Event using this feature. |
| Edit Existing Event | Event Organizers | Event Organizers can edit their Event using this feature. |
| Cancel Event | Event Organizers | Event Organizers can cancel their Event using this feature. |

*Features Not to be Tested*

Features that will not be tested include:

- Accessibility of each Account type
- Successful of transactions requiring connectivity with Banks
- Automatic notifications by System
- Conditional number of tickets available to Customers for purchase
- Promotional features such as the Boost Event use case
- Bidding process to increase visibility of Events

These features will not currently be tested as they will be implemented at a later time. This round of testing is also to ensure that the features meet the basic system requirements.

*Test Approach*

Acceptance tests will be conducted according to the test case specifications. Each tester will perform the test steps as outlined in the test case specifications. Once the test is executed, based on the results, the tester will mark the test as Passed or Failed. Everytime a test case is marked as failed, the generated error log must be sent to the test manager for review, so that they can liaise with the development team to prepare for possible fixes. The tests will be conducted before the release of the product (Alpha Testing). Further testing (Beta) will be launched in the near future. End-users will not be able to try out the product until the team deems it functional for public use.

*Resources*

TicketBooth consist of a team of developers who share a common goal of launching named app to the public. The company has not reach a size that requires Human Resources. The equipment in use are personal computers and rented servers.

*Item Pass / Fail Criteria*

1. 95% of test cases completed successfully, with a 5% margin allowed accounting for minor defects with system and outliers that should not impact user satisfaction.
2. Features will be deemed successful if they respond in a timely manner.
3. All test steps must pass for the test case to pass.

*Suspension Criteria and Resumption Requirements*

Testing must be suspended when:

1. The software throws several fatal errors that need to be addressed immediately.
2. There are significant changes in the project's requirements.
3. There are serious hardware problems.

Testing will only resume when the problems that caused the suspension are properly resolved.

*Test Deliverables*

The following documentation will be produced during these tests:

1. Test Plan
2. Test Cases
3. Test Design Specifications
4. Error logs
5. Execution logs
6. Test Summary Report

*Environmental Needs*

The test site must be populated with test data that includes Events with varying details such as Location / Date / Timing / Capacity. There must also be user accounts with various levels of access rights, ie. Customer, System Administrator, Event Organizer set up. The interface must also allow different features for different user.

*Responsibilities*

The Test Designer is in charge of finalizing comprehensive sets of steps for the test cases. Test Designers must complete the test cases by the provided date.

The Tester includes customer (TicketBooth) acting as Event Organizer / System Administrator / Customer. Tester also includes Project Manager (Russell Coke). This test does not include testing for Wholesale features. The Tester is responsible for facilitating the test steps. Each Tester should understand the expectations of the use cases as well as the importance of completing the test by the deadline assigned.

The Test Manager is in charge of reviewing the error logs and the test case results. They will be a member from the development team capable of liasing with the users and the developers.

*Schedule*
This round of testing should take only about 1 day to complete. The level of progress requires minimal timing between testing and milestone report.

*Risks and Contingencies*
If the first round of testing is not completed in a timely manner, it could delay bug fixes and final testing. If this happens, deadlines would be pushed back affecting the launch date of the software. If the testers don't have a basic exposure to the interface, testing could be delayed or not conducted properly.