T.T.S Verification Test Plan

Presented by: Robert Albon

Unit Test Plans

Total Tickets Bought: __**Test Plan 1**__

1) **userPayment guestUser1**

```
testAmount = 23;
guestUser1.totalTickets = testAmount;
overMaximum = verifyCount(guestUser1.totalTickets);
if(overMaximum == 1) PASS
else FAIL
```

2) **userPayment guestUser2**

```
testAmount = 2;
guestUser2.totalTickets = testAmount;
overMaximum = verifyCount(guestUser2.totalTickets);
if(overMaximum == 1) PASS
else FAIL
```

- **Description:** This test uses the verifyCount function along with the total tickets the user is attempting to purchase to determine whether the system prevents them from purchasing more than the maximum allotment of tickets per user. The test sets an amount and uses the totalTickets variable to plugin to the verifyCount function and determine if the user will get an error or not.

Seat Availability: __**Test Plan 2**__

1) **movieScreening badMovie**

```
seatMap = badMovie.ticketSeat = [0, 0, 0, 0, 1, 1];
seatAvailable = badMovie.verifySeat(3)
assert(seatMap == seatAvailable)
```

2) **movieScreening goodMovie**

```
seatMap = goodMovie.ticketSeat = [1, 1, 1, 0, 1, 1];
seatAvailable = goodMovie.verifySeat(4)
assert(seatMap == seatAvailable)
```

- **Description:** This test is used as a verification that the verifySeat function can be used to determine whether a seat is available based on given seats taken / reserved. The test sets a movie, grabs the seats taken from availableSeats and then uses verifySeat function and assert to verify seating.

Functional Test Plans

Tickets Available For Movie: **<u>Test Plan 1</u>**

1) **movieScreening aMovie1**

```
currentUser guestUser;
currentTickets = 60;
if(currentTickets >= guestUser.totalTickets)
  currentTickets = currentTickets - guestUser.totalTickets;
else if(currentTickets < guestUser.totalTickets)
currentTickets = currentTickets;
aMovie1.ticketCounter(currentTickets);
return(aMovie1.currentTickets.ticketCounter());
```

2) **movieScreening aMovie2**

```
currentUser guestUser;
currentTickets = 0;
if(currentTickets >= guestUser.totalTickets)
  currentTickets = currentTickets - guestUser.totalTickets;
else if(currentTickets < guestUser.totalTickets)
currentTickets = currentTickets;
aMovie2.ticketCounter(currentTickets);
return(aMovie2.currentTickets.ticketCounter());
```

- **Description:** This test attempts to use the tickets available for a movie and the tickets being purchased to determine the amount of tickets newly available. It takes a guestUsers totalTickets and sees if the amount is valid in the scope of tickets available then removes the tickets being bought from the total tickets counter to set the new amount available.

User Ticket Discount: **Test Plan 2**

1) **movieTicket aMovie (Military Discount)**

```
currentUser guestUser;
guestUser.userDiscount = “Military”
aMovie.ticketPrice = 20.50;
        if(aMovie.verifyDiscount(guestUser) == 1)
        aMovie.ticketPrice = applyDiscount(aMovie,guestUser)
else if(aMovie.verifyDiscount(guestUser) == 0)
        aMovie.ticketPrice = aMovie.ticketPrice;
return(aMovie.ticketPrice)
```

2) **movieTicket aMovie (Student Discount)**

```
        currentUser guestUser;
guestUser.userDiscount = “Student”
aMovie.ticketPrice = 20.50;
        if(aMovie.verifyDiscount(guestUser) == 1)
        aMovie.ticketPrice = applyDiscount(aMovie,guestUser)
else if(aMovie.verifyDiscount(guestUser) == 0)
        aMovie.ticketPrice = aMovie.ticketPrice;
return(aMovie.ticketPrice)
```

3) **movieTicket aMovie (No discount)**

```
        currentUser guestUser;
guestUser.userDiscount = “None”
aMovie.ticketPrice = 20.50;
        if(aMovie.verifyDiscount(guestUser) == 1)
        aMovie.ticketPrice = applyDiscount(aMovie,guestUser)
else if(aMovie.verifyDiscount(guestUser) == 0)
        aMovie.ticketPrice = aMovie.ticketPrice;
return(aMovie.ticketPrice)
```

- **Description:** This test takes the current user and verifies whether or not they satisfy the requirements to apply a discount to their ticket. Using the functions of movieTicket the system determines which discount the user is eligible for then applies said discount to the ticket being purchased which then returns the new ticket price. Tests the ability to change price / apply discount.

System Test Plans

User With Account (Premium Theater): **Test Plan 1**

- Login with credentials (Username: guest Password: testpassword)
- Verify subscription
- Select available movie from movies being played
- Select runtime for movie to attend
- Select ticket type
- Select ticket seat
- Select ticket discount (Military, Student, None)
- Verify ticket
- Purchase ticket amount (0 to 20)
- Confirm purchase
- Logout

- **Description:** This test runs through the functionality of the program with a user who has their own account through the system and verifies that they are able to move through each of the classes and their functionalities with no issues.

User Without Account (Non-Premium Theater): **Test Plan 2**

- Proceed as guest
- Select available movie from movies being played
- Select runtime for movie to attend
- Select ticket type
- Cancel ticket
- Create user account (Username: newguest Password: newtestpassword)
- Select available movie from movies being shown
- Select runtime for movie to attend
- Select ticket type
- Select discount type (Military, Student, None)
- Verify ticket
- Purchase ticket amount (0 to 20)
- Confirm purchase
- Logout

- **Description:** This test runs through the overall functionality of a user who at the instance isn’t a registered member of the system. The user runs through half the process once over before canceling and using the system to create an account.