

# Event Management System – Test Case Report

Test Case & Execution Report

## Event Management System - Test Case Report

### Document Information

Project Name	Event Management System Testing
Document Type	Test Case Execution Report
Version	1.0
Author	Bassam Ashraf
Date	September 28, 2025
Review Status	Draft

### Executive Summary

#### Test Execution Overview

Total Test Cases Executed	21
Passed	18
Failed	3 (Due to backend bugs)
Pass Rate	85.71%
Testing Duration	2 weeks
Testing Tools	Selenium WebDriver, Postman, JMeter

#### Key Testing Areas

- Frontend User Interface Testing (11 scenarios)
- Backend API Testing (10 endpoints)
- Performance Testing (Load & Stress)

#### GitHub Repository:

- Link of Repo:  
[https://github.com/Bassam0Ashraf/Event\\_Management\\_System\\_Testing.git](https://github.com/Bassam0Ashraf/Event_Management_System_Testing.git)

## Section 1: Frontend Test Cases (Selenium WebDriver)

### Frontend Test Cases Summary

Test ID	Title	Category	Priority	Method	Status
TC-001	Valid Admin Login	Authentication	High	Automated (Selenium)	PASS
TC-002	Invalid Login Credentials	Authentication	High	Automated (Selenium)	PASS
TC-003	Empty Email Field Validation	Input Validation	Medium	Automated (Selenium)	PASS
TC-004	Empty Password Field Validation	Input Validation	Medium	Automated (Selenium)	PASS
TC-005	Both Fields Empty Validation	Input Validation	Medium	Automated (Selenium)	PASS
TC-006	Home to Dashboard Navigation	Navigation	Medium	Automated (Selenium)	PASS
TC-007	Login to Logout Flow	Authentication Flow	High	Automated (Selenium)	PASS
TC-008	Login page to Register Navigation	Navigation	Low	Automated (Selenium)	PASS
TC-009	Event Creation	Event Management	High	Automated (Selenium)	PASS
TC-010	Event Deletion	Event Management	High	Automated (Selenium)	PASS
TC-011	End-to-End RSVP Workflow	RSVP Management	High	Automated (Selenium)	PASS

#### *TC-001: Valid Admin Login*

**Category:** Authentication

**Priority:** High

**Method:** Automated (Selenium)

**Prerequisites:** Application accessible, admin account exists

**Test Steps:**

1. Navigate to application URL (http://localhost:3000)
2. Enter admin email: admin@gmail.com
3. Enter admin password: admin123
4. Click login button
5. Verify successful login

**Test Data:**

- Email: admin@gmail.com
- Password: admin123

**Expected Result:** User successfully logged in, redirected to home page with logout button visible

**Actual Result:** Login successful, logout button displayed

**Status:** PASS

***TC-002: Invalid Login Credentials***

**Category:** Authentication

**Priority:** High

**Method:** Automated (Selenium)

**Test Steps:**

1. Navigate to login page
2. Enter invalid email: Wrongusername
3. Enter invalid password: Wrongpassword
4. Click login button
5. Verify error message displayed

**Test Data:**

- Email: Wrongusername
- Password: Wrongpassword

**Expected Result:** Error alert with message "Invalid login credentials"

**Actual Result:** Alert displayed with correct error message

**Status:** PASS

***TC-003: Empty Email Field Validation***

**Category:** Input Validation

**Priority:** Medium

**Method:** Automated (Selenium)

**Test Steps:**

1. Navigate to login page
2. Leave email field empty
3. Enter valid password: admin123
4. Click login button
5. Verify login fails

**Test Data:**

- Email: Empty
- Password: admin123

**Expected Result:** Login should fail, no redirect to home page

**Actual Result:** Login failed as expected

**Status:** PASS

***TC-004: Empty Password Field Validation***

**Category:** Input Validation

**Priority:** Medium

**Method:** Automated (Selenium)

**Test Steps:**

1. Navigate to login page
2. Enter valid email: admin@gmail.com
3. Leave password field empty
4. Click login button
5. Verify login fails

**Test Data:**

- Email: admin@gmail.com
- Password: Empty

**Expected Result:** Login should fail, no redirect to home page

**Actual Result:** Login failed as expected

**Status:** PASS

***TC-005: Both Fields Empty Validation***

**Category:** Input Validation

**Priority:** Medium

**Method:** Automated (Selenium)

**Test Steps:**

1. Navigate to login page
2. Leave both email and password fields empty
3. Click login button
4. Verify login fails

**Test Data:**

- Email: Empty
- Password: Empty

**Expected Result:** Login should fail, no redirect to home page

**Actual Result:** Login failed as expected

**Status:** PASS

***TC-006: Home to Dashboard Navigation***

**Category:** Navigation

**Priority:** Medium

**Method:** Automated (Selenium)

**Test Steps:**

1. Login as admin user
2. Verify home page is displayed
3. Click Dashboard button
4. Verify dashboard page loads with "Create New Event" header
5. Click Home button
6. Verify return to home page

**Test Data:**

- Email: admin@gmail.com
- Password: admin123

**Expected Result:** Smooth navigation between home and dashboard pages

**Actual Result:** Navigation worked correctly, pages loaded as expected

**Status:** PASS

***TC-007: Login to Logout Flow*****Category:** Authentication Flow**Priority:** High**Method:** Automated (Selenium)**Test Steps:**

1. Login as admin user
2. Verify successful login (home page visible)
3. Click logout button
4. Verify return to login page

**Test Data:**

- Email: admin@gmail.com
- Password: admin123

**Expected Result:** User logged out and redirected to login page**Actual Result:** Logout successful, login page displayed**Status:** PASS***TC-008: Login to Register Navigation*****Category:** Navigation**Priority:** Low**Method:** Automated (Selenium)**Test Steps:**

1. Navigate to login page
2. Click Register button
3. Verify register page loads
4. Click Login button
5. Verify return to login page

**Expected Result:** Navigation between login and register pages works correctly**Actual Result:** Page navigation successful**Status:** PASS

**TC-009: Event Creation****Category:** Event Management**Priority:** High**Method:** Automated (Selenium)**Test Steps:**

1. Login as admin user
2. Navigate to Dashboard page
3. Fill event creation form
4. Click "Create Event" button
5. Verify event appears on home page

**Test Data:**

- **Admin account:**
  - Email: admin@gmail.com
  - Password: admin123
- **Event creation form:**
  - Name: Test Event
  - Description: Test Description.
  - Date: 01-01-2025
  - Time: 10:00 AM
  - Location: Test Location

**Expected Result:** Event created successfully and visible in event list**Actual Result:** Event created and displayed with correct details**Status:** PASS**TC-010: Event Deletion****Category:** Event Management**Priority:** High**Method:** Automated (Selenium)**Test Steps:**

1. Login as admin user
2. Navigate to home page with existing events
3. Click Delete button on an event
4. Verify event is removed from list

**Test Data:**

- Email: admin@gmail.com
- Password: admin123

**Expected Result:** Event successfully deleted from the system

**Actual Result:** Event removed from display

**Status:** PASS

***TC-011: End-to-End RSVP Workflow***

**Category:** RSVP Management

**Priority:** High

**Method:** Automated (Selenium)

**Test Steps:**

1. Login as admin, create test event
2. Logout from admin account
3. Login as regular user (mohamed@gmail.com)
4. Navigate to event list
5. Click RSVP button for created event
6. Verify RSVP confirmation message
7. Verify RSVP status updates to "You have RSVPed to this event"

**Test Data:**

- **Admin account:**
  - Email: admin@gmail.com
  - Password: admin123
- **Regular user account:**
  - Email: bassam@gmail.com
  - Password: bassam123
- **Event creation form:**
  - Name: Test Final Project
  - Description: Testing Event Management System.
  - Date: 25-05-2025
  - Time: 10:00 AM
  - Location: Test Location online meeting



**Expected Result:** RSVP process completes successfully with confirmation

**Actual Result:** RSVP confirmed, status updated correctly

**Status:** PASS

## Section 2: Backend API Test Cases (Postman)

### API Test Cases Summary

Test ID	Title	Endpoint	Method	Status
TC-012	User Registration	POST /api/auth/register	Automated (Postman)	PASS
TC-013	User Login	POST /api/auth/login	Automated (Postman)	PASS
TC-014	Create Event (Admin Only)	POST /api/events	Automated (Postman)	PASS
TC-015	Get All Events	GET /api/events	Automated (Postman)	PASS
TC-016	RSVP to Event	POST /api/events/:eventId/rsvps	Automated (Postman)	PASS
TC-017	Get Event RSVPs (Admin Only)	GET /api/events/:eventId/rsvps	Automated (Postman)	PASS
TC-018	Delete Event (Admin Only)	DELETE /api/events/:eventId	Automated (Postman)	PASS
TC-019	Get User Profile	GET /api/users/profile	Automated (Postman)	FAIL
TC-020	User Registration (Already exist)	POST /api/auth/register	Automated (Postman)	FAIL
TC-021	Generate new JWT token	POST /api/auth/login	Automated (Postman)	FAIL

#### TC-012: User Registration

**Endpoint:** POST /api/auth/register

**Category:** API Authentication

**Priority:** High

**Method:** Automated (Postman)

#### Test Steps:

1. Send POST request to registration endpoint
2. Include valid user data in request body
3. Verify 201 response status
4. Validate response schema

**Test Data:**

```
{
  "username": "admin",
  "email": "admin@gmail.com",
  "password": "admin123"
}
```

**Expected Result:** 201 Created status with user object in response

**Actual Result:** User registered successfully with correct schema

**Status:** PASS

**TC-013: User Login**

**Endpoint:** POST /api/auth/login

**Category:** API Authentication

**Priority:** High

**Method:** Automated (Postman)

**Test Steps:**

1. Send POST request to login endpoint
2. Include valid credentials
3. Verify 200 response status
4. Extract and store JWT token

**Test Data:**

```
{
  "username": "mohamed",
  "email": "mohamed@gmail.com",
  "password": "Mohamed@Test_123"
}
```

**Expected Result:** 200 OK status with JWT token in response

**Actual Result:** Login successful, token generated and stored

**Status:** PASS

**TC-014: Create Event (Admin Only)**

**Endpoint:** POST /api/events

**Category:** API Event Management

**Priority:** High

**Method:** Automated (Postman)

**Test Steps:**

1. Send POST request with admin JWT token
2. Include event data in request body
3. Verify 201 response status
4. Validate event object schema

**Test Data:**

```
{  
  "name": "Tech Conference",  
  "description": "A conference to showcase the latest  
  technology trends.",  
  "date": "2024-12-20"  
}
```

**Expected Result:** 201 Created with event object

**Actual Result:** Event created successfully

**Status:** PASS

***TC-015: Get All Events*****Endpoint:** GET /api/events**Category:** API Event Management**Priority:** High**Method:** Automated (Postman)**Test Steps:**

1. Send GET request with valid JWT token
2. Verify 200 response status
3. Validate events array schema
4. Check RSVP status for each event

**Expected Result:** 200 OK with array of event objects**Actual Result:** Events retrieved with correct schema**Status:** PASS***TC-016: RSVP to Event*****Endpoint:** POST /api/events/:eventId/rsvps**Category:** API RSVP Management**Priority:** High**Method:** Automated (Postman)**Test Steps:**

1. Send POST request with user JWT token
2. Include event ID in URL parameter
3. Verify 201 response status
4. Validate RSVP object schema

**Expected Result:** 201 Created with RSVP confirmation**Actual Result:** RSVP created successfully**Status:** PASS

***TC-017: Get Event RSVPs (Admin Only)*****Endpoint:** GET /api/events/:eventId/rsvps**Category:** API RSVP Management**Priority:** High**Method:** Automated (Postman)**Test Steps:**

1. Send GET request with admin JWT token
2. Include event ID in URL parameter
3. Verify 200 response status
4. Validate RSVPs array schema

**Expected Result:** 200 OK with array of RSVP objects**Actual Result:** RSVPs retrieved successfully**Status:** PASS***TC-018: Delete Event (Admin Only)*****Endpoint:** DELETE /api/events/:eventId**Category:** API Event Management**Priority:** High**Method:** Automated (Postman)**Test Steps:**

1. Send DELETE request with admin JWT token
2. Include event ID in URL parameter
3. Verify 200 response status
4. Confirm deletion message

**Expected Result:** 200 OK with deletion confirmation**Actual Result:** Event deleted successfully**Status:** PASS

**TC-019: Get User Profile****Endpoint:** GET /api/users/profile**Category:** API User Management**Priority:** Medium**Method:** Automated (Postman)**Test Steps:**

1. Send GET request with valid JWT token
2. Verify 200 response status
3. Validate user profile schema

**Expected Result:** 200 OK with user profile object**Actual Result:** Profile not retrieved as expected**Status:** FAIL**TC-020: User Registration (Already exist)****Endpoint:** POST /api/auth/register**Category:** API Authentication**Priority:** High**Method:** Automated (Postman)**Test Steps:**

1. Send POST request to registration endpoint
2. Include valid user data in request body
3. Verify 201 response status
4. Validate response schema

**Test Data:**

```
{
  "username": "admin",
  "email": "admin@gmail.com",
  "password": "admin123"
}
```

**Expected Result:** 400 status code with message "Email already exists."

**Actual Result:** 500 status code "Something went wrong!"

**Status:** FAIL

**TC-021: Generate new JWT token**

**Endpoint:** POST /api/auth/login

**Category:** API Authentication

**Priority:** High

**Method:** Automated (Postman)

**Test Steps:**

1. Send POST request to login endpoint for first attempt to generate JWT token
2. Include valid credentials
3. Verify 200 response status
4. Extract and store JWT token
5. Send POST request to login endpoint for second attempt after 1hr for example to generate new JWT token
6. Extract and store new JWT token and compare it with first one

**Test Data:**

```
{
  "username": "mohamed",
  "email": "mohamed@gmail.com",
  "password": "Mohamed@Test_123"
}
```

**Expected Result:** 200 OK status with new JWT token in second response

**Actual Result:** Login failed, the JWT token is the same for both attempts generated

**Status:** FAIL

## Section 3: Performance Test Results

### Load Test Results

**Test Configuration:** 10 concurrent users, 2 minutes duration.

**Scenario:** Login → Get Events → RSVP workflow.

#### Results:

1. Total Request ~1,200
2. Average Response Time: <200ms
3. Error Rate: 0%
4. Throughput: ~10 requests/second

### Stress Test Results

**Test Configuration:** Progressive load 0→5→10→15 users

#### Results:

1. Peak Load Handled: 15 concurrent users
2. System Stability: Maintained
3. Response Degradation: Minimal
4. Breaking Point: Not reached

## Section 4: Test Summary and Conclusions

### Test Coverage Analysis

- **Functional Testing:** 100% of identified features tested.
- **API Testing:** All 8 endpoints validated.
- **UI Testing:** Complete user journey coverage.
- **Performance Testing:** Load and stress scenarios executed.

### Quality Assessment

- **Overall Pass Rate:** 85.71% (18/21 test cases passed).
- **Critical Functionality:** All core features working.
- **User Experience:** Smooth navigation and interactions.
- **Performance:** System handles expected load efficiently.

### Recommendations

1. Address the 3 identified backend bugs for 100% pass rate.
2. Implement automated regression testing.
3. Consider additional cross-browser testing.
4. Plan for higher load capacity testing as user base grows.