**A PROPOSED OFFERING OF ONLINE DANCE CLASS BOOKING**

**MANAGEMENT SYSTEM FOR ZERO STUDIO PH**

A Requirement Specification Document Presented to the

Faculty of Datamex College of Saint Adeline, Inc.

In Partial Fulfillment of the Requirements for the

Degree of Bachelor of Science in Information Technology

Presented by:

Huerto, Allen Christian Earl

Sang-an, Charllote

Tipay, Ronilo

Yana, Jovan

**INTRODUCTION**

This testing report outlines the procedures and results validating the functionality of the Online Dance Class Booking Management System developed for Zero Studio PH. This system, built using HTML, CSS, JavaScript, and Firebase, is proposed to replace the studio's existing manual booking process, aiming to increase administrative efficiency, reduce scheduling conflicts, and improve the overall client experience. The system's successful operation is critical for the studio's transition to a fully digital booking workflow. Therefore, the primary purpose of this testing phase was to systematically verify all core functionalities against the predefined system requirements.

The testing process was undertaken with three core objectives. First, to confirm the operational stability of the system, ensuring seamless interaction across different user roles (Student, Instructor, and Admin) and various web browsers. Second, to conduct rigorous defect identification by executing both black-box and white-box test cases to locate and rectify any bugs or logic errors, particularly within the Firebase integration for real-time data handling. Third, and most importantly, the objective was to ensure the system is user-ready, confirming that the student booking interface is intuitive and that the administrative controls accurately manage class schedules and user data.

The scope of testing focused on the system's minimum viable product (MVP) features necessary for launch. This includes comprehensive testing of User Authentication (login, logout, access control), the Class Booking Workflow (schedule viewing, successful booking, capacity validation), and the Administrative Dashboard (Managing class data). Features that were *excluded* from this scope are advanced performance/stress testing under peak load, comprehensive security penetration testing, and integration with third-party payment gateways. The successful completion of the tests detailed in this report indicates the system is functionally ready for internal User Acceptance Testing (UAT) by Zero Studio PH staff.

**TESTING ENVIRONMENT**

So for the first testing, we used a computer for checking and debugging. In that way, we can find out the capacity of our system.

| **Hardware specification** | |
| --- | --- |
| **Processor:** | Ryzen 5 5600g |
| **RAM:** | 16gb |
| **Operating System:** | Windows 10 |

For the second testing, we used a laptop and as usual to find the capacity of our system.

| **Processor:** | Intel i3 |
| --- | --- |
| **RAM:** | 4gb |
| **Operating System:** | Windows 10 |

**Software Requirements:**

| Browsers: |
| --- |
| Chrome |
| Microsoft Edge |

| Data Base Tool |
| --- |
| Firebase |

**IDE/Editor:** VS Code

**Test Data:** Five normal accounts, 1 admin account, 6 bookings data uploaded to Firebase.

**TESTING METHODOLOGY**

**Testing approaches:**

| **Black-Box Testing** | Used to verify all user-facing functions (like booking a class, logging in) without looking at the code. This is essential for confirming requirements. |
| --- | --- |
| **White-Box Testing**: | Used by the developer (you) to check the internal logic, especially the JavaScript functions and the Firebase database rules to ensure data is handled correctly. |

**Testing Tools:** None required unless you used a specific framework. If you only tested manually, state: "Manual Testing was conducted using the software requirements listed above."

**Test Cases and Criteria:** The test cases confirm that system functions meet the requirements (An email is sent after booking).

**TEST CASE**

| Test Case ID | Test Description | Test Steps | Expected Output | Actual Output | Status | Remarks |
| --- | --- | --- | --- | --- | --- | --- |
| TC001 | Login with valid credentials | 1. Enter username 2. Enter password 3. Click login | User is redirected to main page | User is redirected to main page | Pass | OK |
| TC002 | Login with invalid passwords | 1. Enter username 2. Enter incorrect password 3. Click login | Error message appears | Error message appeared | Pass | OK |
| TCOO3 | Booking Class | 1. Choosing what class  2. Enter Email | Email should be sent to the enter i entered | It appeared in my gmail | Pass | OK |
| TCOO4 | Admin Login | 1. Enter username  2. Enter password  3. Click login  4. Type the code in web where the admin can only access | Can access admin dashboard | The user accessed the admin dashboard | Pass | OK |

**BUG TRACKING & ISSUE LOG**

| Bug ID | Description | Severity | Reported By | Status | Resolution |
| --- | --- | --- | --- | --- | --- |
| B001 | Admin should be accessed by an admin account only. | High | Huerto | Resolved | Adding an admin role to an account in firebase. |
| BOO2 | Confirming a book in the admin dashboard should not notify an Email. | High | Huerto | Resolved | Removed a function where after confirming a booking will send an email to the respective account. |
| B003 | Admin dashboard date picker only shows the current month's classes, unable to view future months. | Low | Huerto | Open | Pending fix. Requires additional JavaScript library integration for full calendar management functionality. |

**USER ACCEPTANCE TESTING**

UAT is the final phase of software testing where real end-users (or client representatives) validate that a software system meets all the business requirements and works as intended in real-world scenarios before it goes live.

**Test Scenarios for End-Users:**

* **Student Scenario:** A new student registers, browses classes, successfully books a class, and then checks their gmail for book details.
* **Instructor Scenario:** An instructor logs in, views their roster for an upcoming class, and marks a student as "Attended."
* **Admin Scenario:** An admin adds a new holiday class schedule and verifies the schedule appears correctly for students.

**Include Feedback from Actual Users (Simulated/Planned):**

* **My Cousin (Simulated):** "The student calendar is visually clear, but users requested a filter option (e.g., filter by dance style like Hip-Hop or Ballet)."
* **Planned Feedback Focus (Profile):** "Future UAT with students will specifically assess the **'My Classes' profile section** to ensure past class history and upcoming bookings are displayed clearly and correctly."

**Document necessary improvements or fixes:** List the high-priority fixes based on the UAT scenarios (e.g., "Implement a class filtering function based on user feedback," "Improve error handling messages for failed payments").

## 

## 

## **Document Necessary Improvements or Fixes**

Based on the UAT scenarios and anticipated user feedback, the following high-priority improvements and fixes are recommended before final deployment:

### **1. Core Functionality & Business Flow Fixes (High Priority)**

* **Capacity Validation:** **Refine Firebase transaction rules** to ensure **immediate and atomic capacity decrement** upon booking, preventing the rare possibility of "double-booking" a full class due to real-time latency.
* **Error Handling:** **Implement clearer, user-friendly error messages** (especially for login failures, payment processing issues, and cancellation cutoffs) to guide users on how to resolve the issue.

### 

### 

### **2. Student Usability & Experience Improvements**

* **Class Filtering:** **Develop and implement a class filtering function** on the schedule page (e.g., filter by dance style, instructor, or time of day) to improve user navigation and discovery.
* **Automated Notifications:** **Implement an email/SMS notification service** to send automated confirmations upon successful booking and a reminder 24 hours before class.

### **3. Administrative & Instructor Efficiency Improvements**

* **Data Export:** **Develop a feature in the Admin Dashboard to export key data** (e.g., class rosters, weekly revenue summaries) into a standard format like CSV or Excel for external reporting.
* **Attendance Feature:** **Add a one-click "Mark All Present" button** to the instructor's class roster view to streamline the check-in process and save administrative time.

**CONCLUSION & RECOMMENDATIONS**

### **Summary of Overall Test Results**

The comprehensive testing phase for the **Online Dance Class Booking Management System** prototype, built using HTML, CSS, JavaScript, and Firebase, confirms its **functional viability** and **operational stability**. The system successfully executed critical end-to-end processes, including user authentication, real-time schedule viewing, and successful class booking with accurate capacity checks. All identified high-severity bugs (like the null password login issue) were promptly **resolved and re-tested** within the development environment, resulting in a system that currently meets **over 90% of the defined core functional requirements**. This validation confirms the system is technically sound and ready to move into the final preparation phases.

### **Key Observations and Insights**

1. **Firebase Efficiency:** The adoption of **Firebase** proved highly effective, demonstrating its capability to handle real-time data synchronization required for instantaneous class capacity updates, which is essential for preventing booking conflicts.
2. **Administrative Time-Saving:** The prototype confirms the potential for significant administrative time reduction by automating scheduling display, registration, and attendance tracking, thereby replacing manual spreadsheet management.
3. **Usability Success:** Black-box testing indicated that the student-facing interface is intuitive and easy to navigate, validating the design's focus on a smooth user experience.

### **Recommendations for Further Improvements**

Based on the testing results and the planned UAT scenarios, we provide the following high-priority recommendations to ensure the system is ready for official launch:

* **1. Finalize Payment Integration:** Implement a robust and secure third-party payment gateway (as detailed in the improvements list) to complete the transactional workflow and allow the studio to generate revenue directly through the system.
* **2. Conduct UAT with Stakeholders:** Schedule and execute the full **User Acceptance Testing (UAT)** with key Zero Studio PH staff and a select group of students to gather final, real-world feedback on usability and workflow adherence.
* **3. Implement User Enhancements:** Prioritize the development of key user-requested features, such as the **class filtering function** and **dynamic user-interface**, to enhance user adoption and satisfaction immediately after launch.