**A PROPOSED OFFERING OF A GYM MANAGEMENT**

**SYSTEM FOR ANYTIME FITNESS GYM**

A Testing Documentation 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

By:

Revilla, James Andrei N.

**INTRODUCTION**

This section introduces the **testing phase of the Gym Management System (GMS)** and explains its purpose, objectives, and scope. It lays the foundation for understanding why testing is a critical part of the software development lifecycle and how it ensures that the system is both reliable and ready for deployment in a real-world environment.

The purpose of the testing process is to validate that all components of the Gym Management System function as intended, meet user requirements, and provide a secure and stable platform for managing gym operations. Testing is not only about identifying bugs but also about verifying that the system delivers a seamless user experience for administrators, staff, and members. By systematically testing the system, developers and testers can confirm that it meets business needs, complies with requirements, and performs well under expected usage conditions.

The objectives of the testing phase include several key points. First, it aims to ensure the **functional correctness** of all core modules, such as staff authentication, member registration, attendance logging, payment processing, and equipment management. Second, it seeks to assess the **performance and reliability** of the system under typical workloads, ensuring that the software responds quickly and without failure. Third, the process evaluates the **usability and accessibility** of the application, ensuring that staff can navigate the system efficiently with minimal training. Finally, it focuses on **security validation**, confirming that sensitive information such as staff credentials, member details, and payment records are handled securely.

The scope of testing for the Gym Management System defines what will and will not be evaluated during this phase. Within scope are all primary features and workflows that affect daily operations, including login and authentication processes, member and staff account management, payment transactions, attendance recording, and report generation. Out of scope for this testing phase are integrations with external third-party systems, such as online payment gateways, SMS, or email services, since these are planned for future system enhancements. Performance testing is limited to local environments and does not cover cloud or distributed deployment scenarios.

By the conclusion of this testing phase, the goal is to provide confidence that the system is stable, functional, and secure enough for production use. The outcomes of the tests, including test cases, bug reports, and user acceptance feedback, will serve as a record of the system’s quality and readiness for release. This introduction underscores the importance of the testing phase as both a safeguard against failure and a guarantee of software reliability for Anytime Fitness Gym.

**TESTTING ENVIRONMENT**

This section describes the hardware, software, and test data used during the testing phase. It ensures that testing was performed under controlled and well-documented conditions, making the results reproducible and reliable.

**Hardware Specification**

| **Device** | **Processor** | **RAM** | **OS** | **Storage** |
| --- | --- | --- | --- | --- |
| Laptop 1 | AMD Ryzen 3 3500U | 8 GB | Windows 10 | 256 GB SSD |
| Laptop 2 | AMD Ryzen 7 5500U | 16GB | Windows 11 | 512 GB SSD |

**Table 1.** Hardware Specification

**Software Requirements**

* **Development Environment**: Visual Basic 2010 (VB.NET)
* **Database Server**: MySQL 8.0 via XAMPP 8.2
* **Testing Tools**: Manual test execution, MySQL Workbench for query validation
* **Frameworks**: .NET Framework 4.x

**Test Data**

* Sample staff accounts (Admin, Staff).
* Dummy member records with varying membership plans and add-ons.
* Sample equipment entries (Treadmill, Dumbbells, Stationary Bike).
* Test payments (downpayment, balance settlement, add-on purchase).

**TESTING METHODOLOGY**

This section outlines the approach used to ensure the system’s quality and reliability. It describes the testing techniques applied, including functional testing to verify features, structural testing to validate code logic, and user acceptance testing to confirm that the system meets end-user requirements.

**Testing Process**

The testing process follows a **hybrid approach**, meaning it combines several testing techniques to thoroughly evaluate both the internal logic and the external behavior of the system. This ensures that the software is not only technically sound but also practical and useful for real-world users.

1. **Black-Box Testing**

Black-box testing is used to verify the system’s functionality without examining the internal code structure. The focus is purely on the inputs and outputs—what the user sees and experiences.

**Purpose:** To confirm that the system behaves as expected when users interact with it.

Examples of Application:

* Validating the login process: ensuring the system accepts correct credentials, rejects incorrect ones, and displays proper error messages.
* Testing payment entries: checking that payment amounts entered by users are stored correctly and reflected in the system reports
* Confirming that the system navigates smoothly from one feature to another without unexpected errors.

1. **White-Box Testing**

White-box testing, also known as structural testing, focuses on examining the internal logic, conditions, and flow of the code. This method requires knowledge of the actual implementation.

**Purpose:** To ensure the system’s logic and data handling are reliable, efficient, and error-free.

Examples of Application:

* Checking conditional statements that handle exceptions, such as applying discounts only if criteria are met.
* Verifying that data flows correctly through the system like the member information is correctly updated in the database after editing a profile.

1. **User Acceptance Testing (UAT)**

User Acceptance Testing involves testing the system in a real-world environment by actual end-users, in this case, gym staff.

**Purpose:** To validate that the system meets operational requirements and is user-friendly.

How It’s Done:

* Staff members perform their everyday tasks using the system, such as registering new gym members, processing payments, booking classes, and checking reports.
* Feedback is collected on whether the system is intuitive, easy to learn, and efficient for daily operations.

**Testing Tools and Frameworks**

The testing process was carried out **purely through manual execution**, without the use of any automated testing tools or frameworks. Testers interacted directly with the system, running it multiple times under different scenarios to validate its functionality and stability.

This involved repeatedly performing typical user actions such as logging in, creating records, updating information, and processing transactions. Each feature was tested by providing both valid and invalid inputs to ensure the system responded correctly in all cases. For example, valid credentials were entered to confirm successful logins, while invalid credentials were used to test error handling. Similarly, various payment transactions were recorded to verify that the system properly saved and displayed them in reports.

By running the system multiple times, testers were able to observe how it behaved under different conditions, identify any inconsistencies, and record issues when they occurred. Since no automation tools were used, the focus was on **realistic user interactions** and **hands-on validation** of the system’s performance, accuracy, and reliability.

**Testing Criteria**

To ensure consistency and clarity in evaluating results, the following criteria were applied:

1. **Pass/Fail Status**

* Each test case had a clear expected result.
* If the actual result matched the expected result, the test was marked Pass.
* If the result did not match, it was marked Fail.

1. **Bug Logging and Severity Levels**
   * **Critical:** Prevents core functions of the system from working (e.g., inability to log in).
   * **High:** Affects important operations but does not completely stop system usage (e.g., payments recorded but reports not updating).
   * **Medium:** Functional errors with a moderate impact, usually involving secondary features.
   * **Low:** Minor issues such as typos, misaligned buttons, or small cosmetic problems.
2. **Readiness for Deployment**

* The system was declared ready for deployment only when all **Critical** and **High severity bugs** were resolved.
* Medium and Low severity issues could remain temporarily but were logged for future fixes and updates.

**TEST CASES**

This section provides detailed test cases designed to validate core functionalities of the system. Each test case includes step-by-step actions, expected outcomes, and actual results, making it easier to track system performance and identify issues.

| **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 dashboard | User is redirected to dashboard | Pass | N/A |
| TC002 | Login with invalid password | 1. Enter username  2. Enter incorrect password  3. Click login | Error message appears | No error message appeared | Fail | Bug identified |
| TC003 | Add new member | 1. Open Members form 2. Enter details 3. Save | Member record is added | Member record is added | Pass | N/A |
| TC004 | Record attendance | 1. Enter MemberID  2. Click Check-In  3. Verify Check-Out | Attendance logged with timestamps | Attendance logged successfully | Pass | N/A |
| TC005 | Payment transaction | 1. Open Payments form 2. Enter amount  3. Save | Payment recorded in database | Payment recorded correctly | Pass | N/A |

**Table 2.** Test Case Execution Log

**BUG TRACKING & ISSUE LOG**

This section documents all bugs or issues encountered during the testing phase. Each bug is carefully recorded with a unique identifier, description, severity, reporter, current status, and resolution. This structured approach ensures efficient tracking and prioritization of issues for timely resolution.

**Bug Severity Categories**

* **Critical:** Issues that prevent core system functionalities from working and must be fixed before deployment.
* **High:** Major issues affecting performance, stability, or usability.
* **Medium:** Moderate issues that do not halt operations but should be addressed.
* **Low:** Minor issues or cosmetic problems that do not impact core functionality.

**Status Tracking**

* **Open:** Issue reported but not yet addressed.
* **In Progress:** Developers are actively resolving the issue.
* **Resolved:** Bug has been fixed and verified by testers.

| **Bug ID** | **Description** | **Severity** | **Reported By** | **Status** | **Resolution** |
| --- | --- | --- | --- | --- | --- |
| B001 | Login page crashes on incorrect password | High | Tester 1 | Open | Pending fix |
| B002 | Payment receipt not printing | Medium | Tester 2 | Resolved | Updated print function |
| B003 | Attendance allows duplicate check-ins | Critical | Tester 1 | In Progress | Validation under review |

**Bug Trackin Log**

**Table 3.** Bug tracking Log

**USER ACCEPTANCE TESTING**

User Acceptance Testing (UAT) was performed to validate that the Gym Management System functions correctly under practical usage scenarios and meets user requirements. The testing focused on simulating typical gym operations to evaluate system usability, functionality, and performance.

**Test Scenarios for End-Users**

1. Registering new members and assigning appropriate membership plans.
2. Recording attendance for multiple members simultaneously.
3. Processing payments and generating receipts accurately.
4. Updating and tracking gym equipment records.

**Feedback from Actual Users**

* Staff found the interface intuitive and easy to navigate.
* Reports were accurate but exporting to Excel or PDF was slower than expected.
* Users requested the addition of a search filter by contact number for faster member lookup.

**Necessary Improvements or Fixes**

* Implement a member search feature using contact numbers.
* Optimize report export functions for faster processing.
* Add tooltips or guidance for dashboard icons to enhance usability.

**CONCLUSION AND RECOMMENDATIONS**

This section summarizes the results of the testing phase, highlights key observations, and provides actionable recommendations to improve the Gym Management System. It aims to ensure that the system is stable, functional, and ready for deployment addressing usability and performance consideration.

**Overall Test Results**

* Core functionalities such as login, member management, payment processing, and attendance tracking are performing correctly.
* Critical issues, including duplicate attendance entries, require attention before deployment.
* Feedback from testing scenarios highlighted minor usability improvements that can enhance efficiency and satisfaction.

**Key Observation and Insights**

* The system demonstrates stability and effectiveness in simulated operational scenarios.
* Simulated end-user feedback provides valuable insights for further optimization.
* Addressing both functional bugs and usability enhancements is essential for a reliable deployment.

**Recommendation for Further improvements**

1. **Resolve all Critical and High severity bugs** before system release to ensure stability and reliability.
2. **Implement additional validations** for attendance logging and payment processing to prevent errors or duplicate entries.
3. **Optimize report generation and exporting features** for faster performance and smoother data handling.
4. **Incorporate usability enhancements**, such as member search by contact number and dashboard tooltips, to improve user experience.
5. **Conduct a follow-up UAT** to verify that all improvements and bug fixes function as intended.
6. **Plan for future testing**, including API integration and performance evaluation, to ensure scalability and compatibility with external systems.
7. **Integrate online payment functionality** to allow members to pay membership fees securely via credit/debit cards, e-wallets, or online banking.
8. **Develop member account management features**, enabling members to create and manage personal profiles, view payment history, track attendance, and update personal information.
9. **Implement automated notifications** for members, such as reminders for membership renewal, payment confirmation, and class schedules.
10. **Enhance security measures** for member accounts and online payments, including encryption, secure authentication, and password management protocols.
11. **Enable reporting and analytics for online transactions and member activity**, providing insights into membership trends and system usage.