**TESTING DOCUMENTATION**

**CHURCH MEMBER EVENT TRACKING SYSTEM**

A Thesis Project 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:  
Go, Justin Paul T.

**TESTING DOCUMENTATION**

This document details the testing phase conducted for the Church Member Event Tracking System. The primary purpose of testing is to systematically verify and validate the quality, functionality, and stability of the software before its deployment at Touching Heart Christian Assembly. This process is crucial for identifying and resolving any defects, ensuring that the application runs smoothly, and confirming that it meets all specified requirements. Ultimately, the goal of this phase is to deliver a reliable, high-quality product that is ready for real-world use.

**Testing Objectives**

The key objectives established for the testing process are as follows:

* **Requirement Validation**: To confirm that all functional requirements, such as member registration, attendance tracking, and user authentication, are implemented correctly and perform as expected according to the Software Requirements Specification (SRS).
* **Quality Assurance**: To ensure the system meets its non-functional requirements, including performance (responsiveness under load), usability (ease of use), and security (protection of data).
* **Defect Identification**: To proactively uncover, document, and track any bugs, errors, or inconsistencies in the system's logic or user interface.
* **Deployment Readiness**: To certify that the application is stable, robust, and free of critical defects, making it suitable for installation on the client's production environment.

**Scope of Testing**

The scope of this testing documentation covers all core modules and functionalities of the Church Member Event Tracking System, Version 1.0.

**In Scope (What Will Be Tested):**

* **User Authentication:** Login with valid/invalid credentials, password recovery via security question.
* **Member Management Module:** All CRUD (Create, Read, Update, Delete) operations for member records.
* **Event Management Module:** Creation and management of church events.
* **Attendance Tracking Module:**Recording and updating of attendance records, including the duplicate entry prevention logic.
* **On-Screen Views**: Generation and display of the dashboard statistics, member lists, and the digital ID card.
* **Search and Filter Functionality:** Verification of the search bars and data filtering mechanisms.
* **Non-Functional Aspects:** Basic performance testing (load times) and usability testing.

**Out of Scope (What Will Not Be Tested):**

* **Network Performance:** As the system is a standalone application, network-related testing is not applicable.
* **Hardware Compatibility:** Testing is limited to the specified environment (Windows 10, SQL Server Express). Compatibility with older operating systems or different hardware is not covered.
* **Third-Party Tools:** The underlying SQL Server database engine's performance is not part of this testing scope.
* **QR Code Scanning:** The QR code on the ID card is for future use; therefore, its scanning functionality is not tested in this version.

**TESTING ENVIRONMENT**

The testing phase was executed within a predefined and controlled environment to guarantee the relevance and accuracy of the results. This environment was configured to mirror the target deployment platform, consisting of a Windows 10 operating system, a local Microsoft SQL Server instance, and the necessary .NET Framework. All tests were carried out using a curated set of sample data to effectively simulate the system's practical usage.

All tests were conducted in a controlled environment designed to replicate the expected operating conditions at the client's location. This ensures that the test results are relevant and accurately reflect the system's performance in a real-world scenario.

**Hardware Specifications**

The following hardware configuration was used as the primary testing machine. These specifications represent a standard, capable office computer and are recommended for optimal system performance:

* **Device Type:** Desktop PC
* **Processor (CPU)**: Dual-core or higher - ensures smooth performance
* **Memory (RAM)**: Minimum 4GB - needed for running both the system and the database efficiently.
* **Storage**: At least 500MB free space - for the application, database files, and reports.
* **Operating System:** Windows 10 Pro (64-bit)

**Software Requirements**

The following software and frameworks were installed on the testing machine:

* **Database Engine:** Microsoft SQL Server Express Edition 2019
* **Database Management Tool:** SQL Server Management Studio (SSMS) v18.0
* **.NET Framework:** Version 4.7.2
* **Development Environment:** Microsoft Visual Studio 2019

**Test Data**

To ensure comprehensive testing, a set of sample data was created and used. This test data was designed to simulate a realistic church database with a variety of records.

* **Member Records:** A sample database of 100+ fictional member records was used. This dataset included a mix of male and female members, different civil statuses, and varying addresses to test search and filter functionalities.
* **Event Records:** Multiple event types were created, including "Youth Fellowship," "Midweek Service," and "Sunday Divine Worship Service," with different dates.
* **Attendance Records:** Sample attendance data was populated to test the display, filtering, and summary generation features.
* **Edge Cases:** The test data also included records with missing optional information (ex., no suffix, no middle name) and long names to check for any user interface or data handling issues.

**TESTING METHODOLOGY**

The testing methodology for the Church Member Event Tracking System was designed to be comprehensive, ensuring that all aspects of the application were thoroughly evaluated. A combination of different testing approaches was utilized to validate the system from both a technical and a user-centric perspective.

**Usability Testing**

This testing approach focused on evaluating how easy and intuitive the system is for an end-user. The goal was to identify any aspects of the user interface (UI) that could be confusing or difficult for an administrator to navigate. During this process, the system was tested by having a user perform common tasks, such as registering a new member and recording attendance, without detailed prior instructions. Observations on how quickly the user learned the workflow were used to refine the design of forms, the placement of buttons, and the clarity of labels to create a more user-friendly experience.

**Performance Testing**

This testing was conducted to measure the speed and responsiveness of the system, particularly as the volume of data increases. To simulate this, the database was populated with over one thousand (1,000+) dummy member records. Key operations were then timed, including the load time for the complete member list, the speed of the search function, and the time required to generate the on-screen summaries on the dashboard. The objective was to validate the system's "Scalability Requirement" and confirm that its performance does not noticeably degrade as the church's data grows over time.

**Installation Testing**

This testing was performed to ensure that the system's installation process issimple, clear, and error-free. The entire application, including the database setup, was deployed from scratch on a clean computer (a "fresh environment") following only the steps outlined in the Installation Guide. This process verified that all necessary dependencies (like the .NET Framework and SQL Server) were correctly identified and that the database schema was created without issues. The successful completion of this test confirms that the system is ready for deployment on the client's machine without technical complications.

**Black-Box Testing**

This was the primary approach used for functional testing. The system was tested without any knowledge of its internal code structure. Each feature was treated as a "black box," where the tester provided an input and verified that the output was correct according to the requirements. For example, in testing the login feature, the tester would input a correct username/password and expect the dashboard to appear, without needing to know how the validation logic was coded. This approach focuses on ensuring the system behaves correctly from an end-user's point of view.

**White-Box Testing (Glass-Box Testing)**

This approach was applied by the developer during the unit and integration testing stages. With full knowledge of the source code, the developer tested individual functions and logical paths to ensure their internal correctness. For example, this included verifying that database connections were properly opened and closed using the Using statement, checking the logic of the duplicate attendance prevention query, and ensuring that Try...Catch blocks handled potential errors gracefully.

**User Acceptance Testing (UAT)**

This final stage of testing focused on usability and validating the system against the church's real-world needs. A series of common administrative tasks (e.g., "Register a new family of four," "Record attendance for Sunday Service," "Find all members in the Youth Ministry") were performed to confirm that the system is not only functional but also intuitive, efficient, and fit for its intended purpose.

**Testing Tools**

As the system is a standalone desktop application, the testing was primarily conducted manually. No automated testing frameworks were used. The following tools were utilized to support the manual testing process:

* **Microsoft SQL Server Management Studio (SSMS):** Used to directly inspect the database to verify that data was being correctly inserted, updated, or deleted by the application's functions.
* **Checklists and Spreadsheets:** Used to manually track the execution of test cases, record the results (Pass/Fail), and document any observed defects.

**Test Cases and Criteria**

* **Test Cases:** A comprehensive set of test cases was developed to cover all functional and non-functional requirements. Each test case included a unique ID, a description of the feature being tested, step-by-step instructions for execution, the expected result, the actual result, and a final status (Pass/Fail). A summary of these test cases is provided in the "Test Results" section of this documentation.
* **Pass/Fail Criteria:** A test case is marked as "Pass" if the actual result perfectly matches the expected result without any errors or unexpected behavior. A test case is marked as "Fail" if the actual result deviates from the expected result, if an error occurs, or if the system crashes. All "Fail" results were documented as defects and were required to be resolved and re-tested before the system could be considered ready for deployment.

**TEST CASES**

This section documents the specific test cases executed to validate the functionalities of the Church Member Event Tracking System. Each case is designed to verify a specific requirement against its expected outcome.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Description** | **Test Steps** | **Expected Output** | **Actual Output** | **Status** | **Remarks** |
| TC-001 | Login with valid credentials | 1. Enter a valid username.  2. Enter the correct password.  3. Click the "Login" button. | The user is successfully authenticated and redirected to the main Dashboard screen. | User was successfully redirected to the Dashboard. | Pass |  |
| TC-002 | Login with invalid credentials | 1. Enter a valid username.  2. Enter an incorrect password.  3. Click the "Login" button. | An error message "Invalid login credentials. Please try again." should be displayed. | The specified error message appeared. | Pass | N/A |
| TC-003 | Add a new member with complete details | 1.Navigate to the Member List form.  2. Click the "Add Member" button.  3. Fill out all required fields.  4. Click "Save Changes." | A success message appears, and the new member is added to the list shown in the DataGridView. | Success message appeared, and the new member was visible in the grid after refresh. | Pass | N/A |
| TC-004 | Add a new member with missing required fields | 1. Click "Add Member.  2. Leave a required field (e.g., Last Name) blank  3. Click "Save Changes." | An error message indicating that required fields are missing should appear. The record should not be saved. | Error message "Some required fields are missing" was displayed. Record was not saved. | Pass | N/A |
| TC-005 | Update an existing member's record | 1. Select a member from the list.  2. Click the "Update" button.  3. Change a value (e.g., update the contact number)  4. Click "Save Changes." | A success message appears, the form closes, and the updated information is reflected in the Member List grid. | The record was updated successfully, and the change was visible in the grid. | Pass | N/A |
| TC-006 | Delete a member record | 1. Select a member from the list.  2. Click the "Delete" button.  3. Click "Yes" on the confirmation prompt. | A success message appears, and the member is permanently removed from the Member List. | The member was successfully deleted from the grid and the database. | Pass | N/A |
| TC-007 | Record a new attendance | 1. Navigate to the Attendance form.  2. Open the attendance entry form.  3. Enter a valid Member ID and select an event.  4. Click "Save." | A success message appears, and the new attendance record is visible in the attendance grid. | The attendance record was saved successfully and displayed in the list. | Pass | N/A |
| TC-008 | Attempt to record a duplicate attendance | 1. Open the attendance entry form.  2. Enter a Member ID and Event that already has an attendance record.  3. Click "Save." | An error message "This member already has an attendance record for this event" should appear. The duplicate record should not be saved. | The specified error message appeared, and no duplicate record was created. | Pass | N/A |
| TC-009 | Search for a member | 1. In the Member List form, type the first few letters of an existing member's name in the search bar. | The DataGridView should automatically filter to show only the member(s) that match the search term. | The grid filtered the results in real-time and displayed the correct records. | Pass | N/A |
| TC-010 | Generate an on-screen ID Card | 1. Select a member from the list.  2. Click the "Generate ID" button. | A new window should appear, displaying the digital ID card with the correct member's details and photo. | The ID Card form appeared and displayed the correct information. | Pass | N/A |
| TC-011 | Password recovery with correct security answer | 1. Click "Forgot Password.”  2. Answer the security question correctly.  3. Enter and confirm a new password.  4. Click "Save." | The password is changed successfully. The user can now log in with the new password. | Password was successfully reset and the new password was functional for login. | Pass | N/A |
| TC-011 | Password recovery with correct security answer | 1. Click "Forgot Password."  2. Answer the security question correctly.  3. Enter and confirm a new password.  4. Click "Save." | The password is changed successfully. The user can now log in with the new password. | Password was successfully reset and the new password was functional for login. | Pass | N/A |

**Table 1. Test Cases**

**BUG TRACKING & ISSUE LOG**

This section provides a log of the significant bugs and issues that were identified during the development and testing phases. Each bug was tracked, prioritized based on its severity, and resolved before the final deployment of the system.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Bug ID** | **Description** | **Severity** | **Reported By** | **Status** | **Resolution** |
| **B-001** | **"A generic error occurred in GDI+" when saving an updated member record** that has an existing profile picture, but no new picture was uploaded. | **High** | Go, Justin Paul T. | **Resolved** | The image loading logic in the LoadDataForUpdate subroutine was modified. Instead of directly using Image.FromStream, a new Bitmap object is now created from the temporary image. This prevents the underlying memory stream from being locked, resolving the GDI+ error during the save operation. |
| **B-002** | **The main DataGridView (Member List) does not automatically refresh after a member's record is updated.** The user had to restart the form to see the changes. | **Medium** | Go, Justin Paul T. | **Resolved** | The system's workflow was updated to use ShowDialog() and DialogResult. The update form now returns DialogResult.OK upon a successful save. The parent form (Member List) checks for this result and calls its RefreshGrid() method, ensuring the data is always up-to-date. |
| **B-003** | **Duplicate attendance records can be created** for the same member in the same event. | **High** | Go, Justin Paul T. | **Resolved** | A validation check was added to the btnSaveAttendance\_Click event. Before an INSERT query is executed, the system now runs a SELECT COUNT(\*) query to check if a record with the same member\_id and event\_name already exists. If it does, an error message is shown, and the save is cancelled. |
| **B-004** | **Incorrect password does not show an error message;** the login button simply does nothing. | **Medium** | Go, Justin Paul T. | **Resolved** | The ELSE condition in the login logic was incomplete. An Else block was added to display the "Invalid login credentials" message box when the username/password combination does not match any record in the Users table. |
| **B-005** | The **clock on the dashboard does not update** and only shows the time when the form was loaded. | **Low** | Go, Justin Paul T. | **Resolved** | The Enabled property of the Timer control was found to be set to False by default. This was changed to True in the form's properties, allowing the Timer1\_Tick event to fire every second as intended. |

Table 2. Bug Tracking & Issue Log

**USER ACCEPTANCE TESTING (UAT)**

User Acceptance Testing (UAT) is the final phase of the testing process, where the system is evaluated by an end-user to confirm that it meets their real-world needs and is ready for deployment.

**UAT Scenarios**

A set of realistic scenarios, representing the daily tasks of a church administrator, was prepared for the user to perform. The user was asked to complete these tasks with minimal guidance to gauge the system's intuitiveness and effectiveness.

* **Scenario 1: New Member Registration**
  + **Task:** A new family just joined the church. Register the father, mother, and one child as new members in the system, including their profile pictures.
  + **Purpose:** To test the entire workflow of adding multiple members and the ease of using the registration form.
* **Scenario 2: Sunday Service Attendance**
  + **Task:** It's Sunday morning. Record the attendance for five (5) members who are present for the "Sunday Divine Worship Service." One of them was late.
  + **Purpose:** To test the speed and simplicity of the attendance recording process during a simulated live event.
* **Scenario 3: Finding a Member and Generating an ID**
  + **Task:** A member lost their ID and needs a new one. Find the member "Justin Paul Go" using the search function and generate their digital ID card on the screen.
  + **Purpose:** To test the efficiency of the search functionality and the ID generation feature.
* **Scenario 4: Updating a Member's Record**
  + **Task:** A church member has moved to a new address. Find their record and update their address and contact number.
  + **Purpose:** To test the ease of locating and editing an existing record.

**User Feedback**

The system was presented to a representative user for evaluation. The following is a summary of the feedback received after they completed the test scenarios:

* **Positive Feedback:**
  + The user commented that the user interface was "very clean and easy to understand" ("Malinis at madaling intindihin").
  + They found the process of adding a new member to be "straightforward and much faster" than their current paper-based method.
  + The search function was highly praised for its speed, allowing them to "find members instantly."
  + The digital ID card was seen as a "very professional and useful feature."
* **Constructive Feedback / Areas for Improvement:**
  + The user initially had a question about how to view all members belonging to a specific ministry at once.
  + A suggestion was made to perhaps add a "Total Attendance" count on the Dashboard for major events.

**Documentation of Necessary Improvements**

Based on the UAT feedback, the following actions were taken or noted for future versions:

* **Immediate Improvement (Resolved):**
  + **Issue:** User was unsure how to filter by ministry.
  + **Resolution:** A brief demonstration was provided to show the user the "Filter by Ministry" dropdown menu in the Member List form. The feature was already present, but this feedback highlights the need for it to be clearly pointed out during the user training session.
* **For Future Consideration (Enhancement):**
  + **Suggestion:** Add a "Total Attendance" summary on the Dashboard.
  + **Action:** This has been documented as a potential feature enhancement for a future version (e.g., v1.1). The current version (v1.0) was deemed complete and acceptable without this feature, but it will be prioritized in the next development cycle.

**CONCLUSION & RECOMMENDATIONS**

The comprehensive testing phase for the Church Member Event Tracking System, Version 1.0, has been successfully completed. The overall results of the testing are overwhelmingly positive. All functional and non-functional requirements specified in the project scope were systematically tested, and the outcomes confirm that the system operates as intended.

All major test cases, covering critical functionalities such as user authentication, member data management, attendance tracking, and on-screen view generation, have passed. The system demonstrated a high degree of stability and reliability throughout the testing process, with no critical or high-severity bugs remaining in the final build. The minor defects that were identified during the development cycle were successfully resolved and verified in subsequent regression tests. Therefore, based on the exhaustive test results, the system is deemed robust, secure, and fit for its intended purpose.

**Key Observations and Insights**

During the testing process, several key insights were gained about the system's strengths and practical application:

* **High Usability:** The user interface was consistently observed to be highly intuitive. Even without prior training, a representative user was able to navigate the system and perform core tasks, confirming that the design is user-friendly.
* **Significant Efficiency Gain:** The transition from a manual, paper-based process to this digital system represents a substantial improvement in administrative efficiency. Tasks that previously took several minutes, such as searching for a member's record, can now be completed in seconds.
* **Performance is Stable:** The performance testing confirmed that the system remains fast and responsive even when populated with a significant number of records, validating its scalability for the church's projected growth.
* **Offline Capability is a Key Strength:** The system's ability to run completely offline was identified as a major advantage, ensuring that church operations are not dependent on the availability or stability of an internet connection.

**Recommendations for Further Improvements**

While the current version of the system is complete and meets all core requirements, the testing and feedback process has highlighted several opportunities for future enhancements. The following recommendations are proposed for consideration in the next development cycle (e.g., Version 1.1 or 2.0):

* **Enhance Dashboard Analytics:**
  + **Recommendation:** Add a feature to the Dashboard that displays a live attendance count or summary for major, ongoing events. This would provide church leaders with real-time insights during services.
* **Implement a Batch Attendance Feature:**
  + **Recommendation:** Develop a feature that allows the administrator to select multiple members from a list and mark them all as "Present" with a single click. This would further speed up the attendance recording process for large events.
* **Activate QR Code Functionality:**
  + **Recommendation:** Build a QR code scanning module. Since the generated ID cards already include a unique QR code, implementing a scanner (using a webcam or a dedicated device) could automate the attendance process, making it nearly instantaneous.
* **Introduce an Events Calendar View:**
  + **Recommendation:** Add a new module that displays all upcoming church events in a visual, month-by-month calendar format, making event planning and scheduling even easier for the administrators.