**MAINTENANCE 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.

**INTRODUCTION**

This document provides the official maintenance plan for the Church Member Event Tracking System; a standalone desktop application developed for Touching Heart Christian Assembly. The system's purpose is to provide a reliable and efficient tool for managing member records and tracking event attendance.

While the Church Member Event Tracking System was developed and tested to be stable and free of critical defects upon deployment, a proactive maintenance plan is essential for several key reasons. Firstly, it ensures data integrity and longevity; regular, scheduled backups are the primary defense against catastrophic data loss due to unforeseen events like hardware failure, file corruption, or user error. Secondly, it guarantees system adaptability. The technological landscape is constantly changing, and this plan outlines a process for applying necessary updates to ensure the application remains compatible with future versions of the Windows Operating System and its dependencies. Lastly, it provides a structured support system. This plan defines a clear process for reporting bugs and requesting enhancements, ensuring that any issues are addressed in a timely and organized manner. In essence, this maintenance documentation is the commitment to the system’s continued success long after its initial installation.

**Scope of Maintenance**

The scope of this maintenance plan is comprehensive, covering all activities required to keep the system operational, secure, and effective. The plan encompasses four primary categories of maintenance:

* **Corrective Maintenance:** Addressing and fixing any bugs or errors reported by the user.
* **Adaptive Maintenance:** Applying necessary updates to ensure the application remains compatible with future versions of the Windows Operating System or .NET Framework.
* **Preventive Maintenance:** Performing regular, proactive tasks like database backups to prevent future problems.
* **Perfective Maintenance:** A process for logging and evaluating user requests for new features or usability improvements for future development cycles.

**MAINTENANCE PLAN**

The overall maintenance strategy for this system is straightforward, focusing on proactive data protection and responsive support. Given its nature as a standalone application, the plan prioritizes user-led preventive tasks and developer-led corrective actions.

* **Corrective Maintenance (Bug Fixes):** This is a reactive form of maintenance, initiated when a defect or error is discovered by the end-user. The primary goal is to restore the system to its correct operational state. The process involves the user reporting the bug with detailed information, the developer replicating and diagnosing the issue, implementing a code fix, testing the fix, and deploying a patch or updated version of the application. All bug fixes will be tracked through an issue log.
* **Adaptive Maintenance (System Updates):** This maintenance type addresses the need for the system to adapt to changes in its external environment. For this application, the most likely scenarios would involve updates to the Windows Operating System or the .NET Framework. If a future Windows update causes a compatibility issue, the developer will be responsible for modifying the application's code to ensure it continues to run correctly on the updated platform.
* **Perfective Maintenance (User-Requested Enhancements):** This type of maintenance is not about fixing errors, but about improving the system's functionality and usability based on user feedback. As the church staff uses the system over time, they may identify opportunities for new features or optimizations (e.g., adding new data fields, requesting a new type of on-screen summary). These requests will be formally logged and evaluated for inclusion in future major version releases (e.g., v1.1, v2.0).
* **Preventive Maintenance (Proactive Care):** This is the most critical type of maintenance to be performed by the user. It involves proactive tasks aimed at preventing future problems. For this system, the single most important preventive task is the regular manual backup of the database. By consistently creating and securely storing copies of the database, the church administrator can prevent permanent data loss, which is the biggest long-term risk for the system.

**MAINTENANCE SCHEDULE**

The following table outlines the recommended schedule for routine maintenance tasks.

|  |  |  |  |
| --- | --- | --- | --- |
| **Task** | **Description** | **Frequency** | **Responsible Person** |
| Database Backup | Manually creating a backup copy of the MemberInfo.mdf and MemberInfo\_log.ldf database files and storing it on a separate external drive. | Weekly (or after every major data entry session) | Church Administrator |
| System Health Check | Running the application, checking for any slowdowns, and ensuring the computer's disk space is not critically low. | Quarterly | Church Administrator |
| OS & Framework Updates | Installing official security patches and updates for the Windows Operating System and .NET Framework. | As Recommended by Microsoft | Church Administrator |
| Bug Fixes | Addressing and resolving reported software bugs. | As Needed | Developer / Support |

**Table 1.** Maintenance Schedule

**ISSUE TRACKING & BUG REPORTS**

A log will be maintained by the developer to track all reported issues. When a user reports a bug, it will be documented using the following format.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Issue ID** | **Description** | **Severity** | **Reported By** | **Date Reported** | **Status** |
| **BUG-001** | Example: "Search function freezes when using special characters." | Medium | Church Admin | MM/DD/YYYY | Open |
| **BUG-002** | Example: "Profile picture appears stretched on the ID card." | Low | Church Admin | MM/DD/YYYY | Resolved |

**Table 2.** Issue Tracking & Bug Reports

**BACKUP & RECOVERY PLAN**

This section details the critical procedures for backing up and restoring the system's data.

**Backup Procedures**

The backup process for this system is a manual procedure that must be performed regularly by the church administrator. This is the most important maintenance task for ensuring the long-term safety of the church's data.

* **Frequency:** It is strongly recommended to perform a full back up at least once a week, or more frequently (e.g., daily) during periods of heavy data entry.
* **Files to Back Up:** A SQL Server database consists of two primary files that must be backed up together:
* **The Main Data File (.mdf):** This file contains all the actual data, including all tables, member records, event details, and attendance logs.
* **The Log Data File (.ldf):** This file is a transaction log that records all modifications made to the database. It is essential for data recovery and integrity.
* **Step-by-Step Backup Procedure:**
  1. **Close the Application:** Ensure the Church Member Event Tracking System application is fully closed on the computer.
  2. **Navigate to the SQL Server Data Directory:** Open File Explorer and go to the directory where SQL Server stores its database files. The typical location is:  
     C:\Program Files\Microsoft SQL Server\MSSQL15.SQLEXPRESS\MSSQL\DATA\
  3. **Copy the Database Files:** Locate and copy the following two files:
     + MemberInfo.mdf
     + MemberInfo\_log.ldf
  4. **Store the Backup Securely:** Paste these two files into a secure, separate storage location. This must not be on the same computer. Recommended locations include:
     + A dedicated external hard drive.
     + A high-capacity USB flash drive.
     + A secure network drive or cloud storage location, if available.

**Recovery Steps**

In case of data loss, hardware failure, or database corruption, follow these steps to restore the last good backup. This procedure should be performed by the developer or a technically proficient person.

1. Stop the SQL Server service.
2. Navigate to the SQL Server data directory.
3. Replace the corrupted MemberInfo.mdf and MemberInfo\_log.ldf files with the backup copies from the external drive.
4. Restart the SQL Server service.
5. Run the application to verify that the data has been restored successfully.

**PERFORMANCE MONITORING**

Performance monitoring for this standalone system is straightforward and can be done manually by the administrator.

|  |  |  |
| --- | --- | --- |
| **Metric** | **Description** | **How to Monitor** |
| **Application Load Time** | The time it takes for the application to open and for the Login Form to appear. | Observe if there is a noticeable delay (more than a few seconds) when launching the application. |
| **Data Grid Load Time** | The time it takes for the Member List or Attendance List to display after clicking the button. | Check if the lists still load quickly as more data is added over time. A significant slowdown may indicate a need for database optimization. |
| **Free Disk Space** | The available storage on the computer's C: drive. | Periodically check "This PC" to ensure there is at least 1 GB of free space for the OS and database to operate smoothly. |

**Table 3.** Performance Monitoring

**SECURITY MEASURES**

This section outlines the security policies and measures relevant to the maintenance of the Church Member Event Tracking System. The focus is on protecting the integrity and confidentiality of the church's data.

* **Access Control:**
  + Physical access to the computer where the system is installed should be limited to authorized personnel only, primarily the church administrator.
  + Database access is restricted. The system connects using a trusted connection, and direct access to the SQL Server database should only be performed by a technically proficient administrator for maintenance tasks like recovery.
* **Authentication Mechanism:**
  + The application is protected by a mandatory login screen requiring a unique username and password. This ensures that only users with valid credentials can access the system's data and functions.
  + A password recovery mechanism using a security question and answer is in place to allow legitimate users to securely reset their password without developer intervention.
* **Data Protection:**
  + As a standalone, offline application, the system is inherently protected from external online threats such as hacking and network-based attacks.
  + The most critical security measure is the regular backup of the database. This protects the church's data from being permanently lost due to hardware failure, accidental deletion, or ransomware attacks.
* **Authentication Mechanism:**
  + The application is protected by a mandatory login screen requiring a unique username and password. This ensures that only users with valid credentials can access the system's data and functions.
  + A password recovery mechanism using a security question and answer is in place to allow legitimate users to securely reset their password without developer intervention.
* **Data Protection:**
  + As a standalone, offline application, the system is inherently protected from external online threats such as hacking and network-based attacks.
  + The most critical security measure is the regular backup of the database. This protects the church's data from being permanently lost due to hardware failure, accidental deletion, or ransomware attacks.

**DOCUMENTATION UPDATES**

This Maintenance Documentation, along with all other related technical documents (SDD, SRS, etc.), is considered a "living document." It must be updated whenever significant changes are made to the system to ensure it remains an accurate and useful reference.

* **Procedure for Updates:**
  + Any time a software update is released (e.g., a new version with bug fixes or enhancements), this document must be reviewed.
  + The Revision History section of the relevant document will be updated with a new entry detailing the date, new version number, and a summary of the changes made.
  + If a new feature is added, the User Manual and other relevant sections must be updated accordingly.
* **Responsibility:** The original developer is responsible for updating the technical documentation in conjunction with any software updates.

**CONCLUSION & RECOMMENDATIONS**

This document has established a comprehensive yet practical maintenance plan for the Church Member Event Tracking System. It outlines a clear strategy that balances user-led preventive tasks, such as regular database backups, with developer-led corrective support. By following the procedures for maintenance, issue tracking, and recovery detailed in this guide, Touching Heart Christian Assembly can ensure the long-term stability, security, and usefulness of their new digital record-keeping system. The plan is designed to be straightforward and effective for a standalone application environment.

**Recommendations for Future Maintenance**

To further enhance the system's robustness and ease of maintenance in the future, the following improvements are recommended:

* **Implement an In-App Backup Feature:**
  + **Recommendation:** For a future version (e.g., v2.0), develop a "Backup Database" button within the application itself. This would allow the administrator to create a backup with a single click, simplifying the manual procedure and reducing the chance of user error.
* **Automate Update Notifications:**
  + **Recommendation:** While the system is offline, a future version could include a simple mechanism to check a shared network folder or a file for an "update available" notice, which would then prompt the administrator that a new version is ready for installation.
* **Enhance Security with Password Hashing:**
  + **Recommendation:** A critical security enhancement for a future version would be to implement a one-way hashing algorithm (like SHA-256) for storing user passwords. This would provide a much stronger layer of protection for user credentials.