**A SYSTEM DESIGN OF A**

**CHURCH MEMBER EVENT TRACKING SYSTEM**

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

By:

Go, Justin Paul T.

**INTRODUCTION**

This document presents the design of the Church Member Event Tracking System, a desktop application that helps the church manage member records, track attendance, and organize events. It provides a clear plan for the system’s structure, database, and interface, making processes easier, faster, and more accurate for administrators.

This document is the Software Design Document (SDD) for the Church Member Event Tracking System. It explains the overall plan of how the system will be built, including its structure, features, and design. The goal is to guide the development process, make sure it follows the project’s objectives, and provide a clear reference for both developers and church leaders.

The Church Member Event Tracking System is made to help the church manage member records, keep track of attendance, and organize events more efficiently. Instead of relying on manual record-keeping, the system will store everything in one place and make it easier for administrators to access information. With this, the church can save time, avoid errors, and have a smoother process in handling members and events.

This design document explains the system’s structure, database, user interface, data flow, security, and how it will be set up on a computer. It also covers how errors will be handled, how the system will be maintained, and how performance will be kept smooth. The system is designed to work as a desktop application for church use, with the possibility of expanding to other platforms in the future if needed.

This design document details the system’s structure, database design, user interface layout, data flow, security mechanisms, and deployment setup. It also covers error handling strategies, maintenance and support guidelines, and performance optimization methods to ensure the system runs smoothly over time.

Furthermore, the system is designed as a scalable and maintainable desktop application using VB.NET and SQL as core technologies. While it primarily supports church administrators, it can be extended to include more features such as report exports, and potential integration with mobile or online platforms in the future

**SYSTEM ARCHITECTURE**

The system is a standalone desktop application that runs locally in the church office. It includes modules for Member Management, Event Management, Attendance Tracking, and User Authentication, all connected to a local SQL database. Communication occurs internally through SQL queries, ensuring smooth and secure offline operation.

**High-Level Components and their Interactions**

The system is made up of key modules such as Member Management, Event Management, Attendance Tracking, and User Authentication. These modules work together through a shared database, where all records are stored and updated. The desktop application interacts directly with the database using SQL queries, ensuring smooth data flow between the user interface and the stored information.

**User Interface**

* **User Authentication Module**. secure login for system administrators.
* **Admin Dashboard.** Displays all the statistics like all member active member and, events included.
* **Member Management Module.** register, update, and search member profiles.
* **Event Management Module.** create and schedule church events.
* **Attendance Tracking Module.** record member attendance during events.
* **Database Layer.** stores member, event, and attendance records.

**Deployment Architecture**

The system is designed for a standalone deployment utilizing a two-tier architecture. Both the client-tier, which is the VB.NET desktop application, and the data-tier, the SQL database, are installed and run on the same local machine. This configuration ensures that all data processing and storage are handled locally, guaranteeing system availability and security without reliance on network connectivity. All data is centralized within the local database, providing a secure and easily accessible record-keeping solution for authorized staff.

* The system runs as a standalone desktop application on a local computer in the church office.
* Database (SQL Server) is installed locally.

**Client-Server Model**

* + **Client** - VB.NET desktop app on Windows PCs.
  + **Server** - (MySQL) on a local machine or networked server.

**Communication Protocol and Interfaces**

The Church Member Event Tracking System is a standalone VB.NET desktop application that communicates with a local SQL Server database using the ADO.NET framework. User interaction is handled through a Windows Forms–based Graphical User Interface (GUI), which includes modules for member management, event tracking, and attendance recording. All internal communication between the application and the database occurs through this secure, local connection. The system is self-contained and does not interface with external services or file formats. Security is maintained through a credential-based authentication system and the safe handling of database connections to protect all records.

**DATABASE DESIGN**

The Church Member Event Tracking System uses a relational SQL database to store and manage information related to church members, events, attendance, and user accounts. The database is designed to support accurate, efficient, and secure data handling within the system.

Core entities include:

* **Members**. Stores personal and contact information of church members, such as full name, birthdate, contact number, and membership status.
* **Events**. Contains information about church events, including event name, date, location, and description.
* **Attendance**. Records attendance of members for specific events, linking members and events through foreign keys.
* **Users**. Manages system users who can log in, with roles and secure authentication credentials.

**Description of Database Tables, Fields, and Relationships**

**User Table**

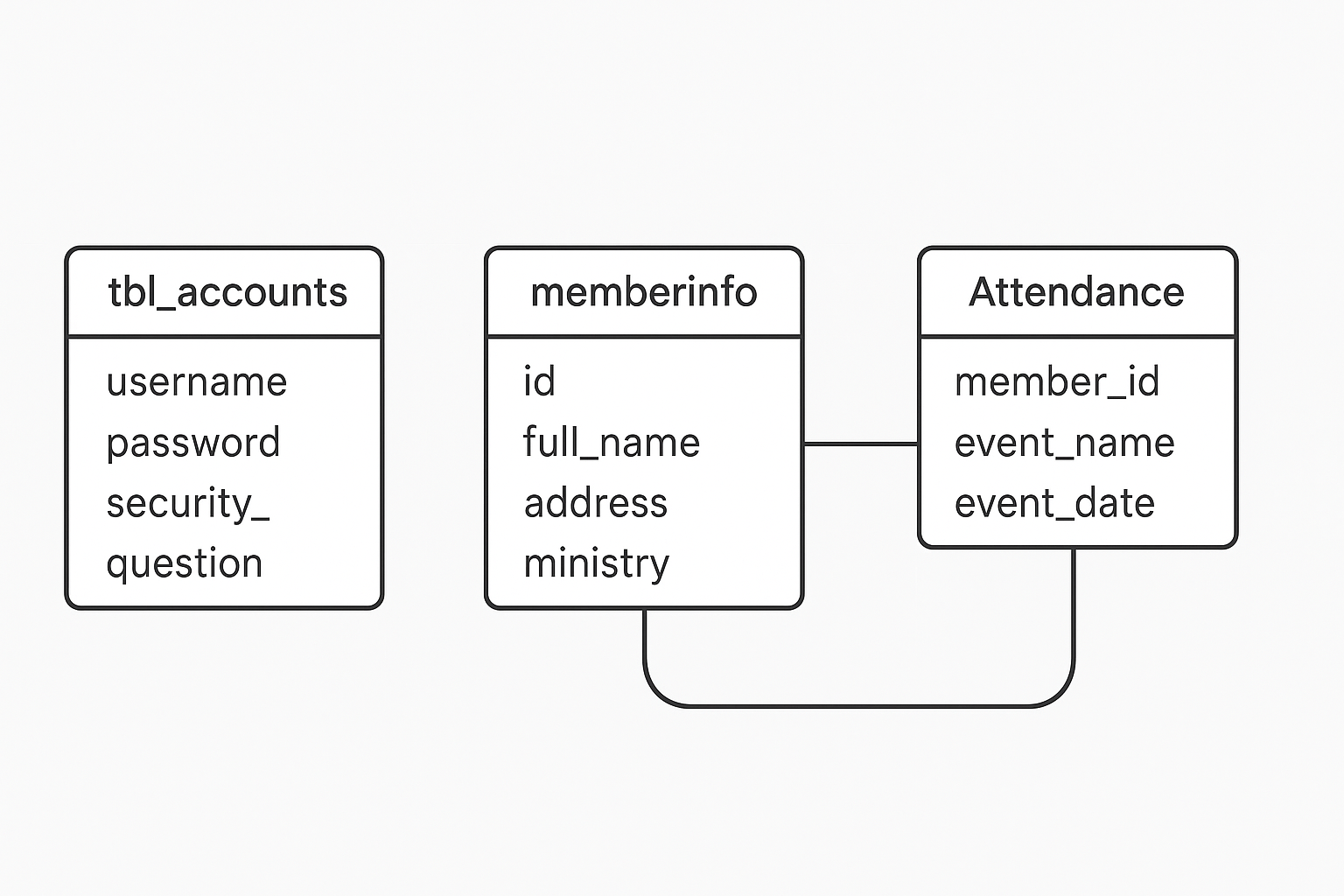
Fields include id (PK), username, password, security question, and security answer. This table stores the login credentials of system users. It is responsible for authentication and controlling access to the system.

**Members Table**

Holds detailed personal information of church members such as last name, first name, middle name, gender, birthday, civil status, address, contact numbers, parents’ names, guardian details, and ministry involvement. The key field id (PK) uniquely identifies each member and links to attendance records.

**Events & Attendance Table**

Contains attendance\_id (PK), member\_id (FK), full\_name, event\_name, event\_date, status (e.g., Present/Absent), and time\_in. This table records attendance for church activities and services, connecting members with the events they participate in.

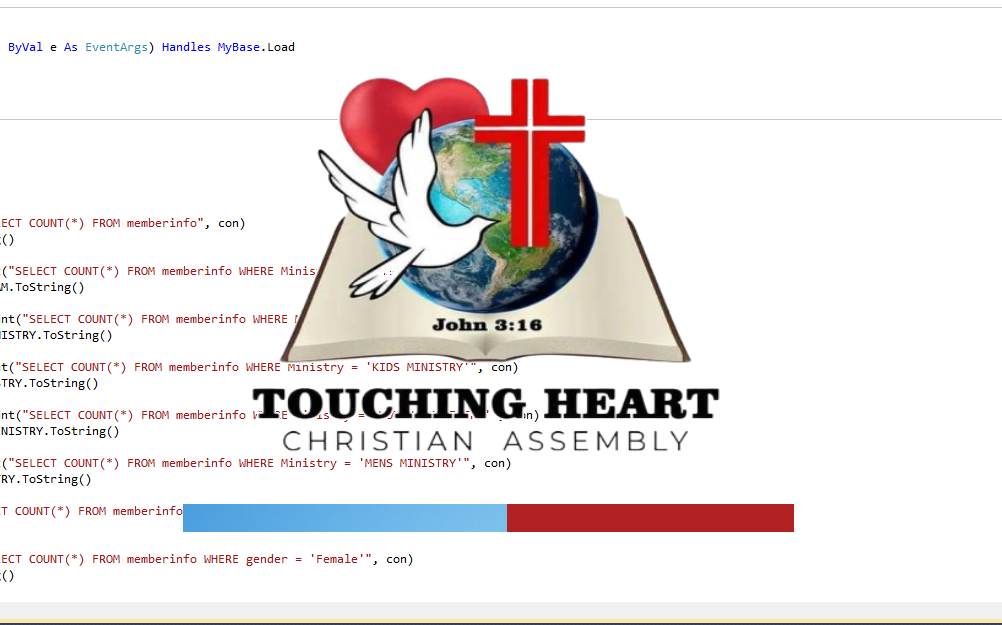
****

**Events &**

**Figure 1.** Database Schema

**USER INTERFACE DESIGN**

**Loading Screen**

This is the loading screen that will first to see before the login forms shows the purpose of a loading screen is to inform users that a process is underway and the system is working, helping to manage expectations and improve perceived performance.

***Figure 2.*** *Loading Form*

**Login Form**

****. The Login Form serves as the secure entry point to the system. Administrators must provide their registered username and password to gain access. The interface is minimalist, focusing on the core function of authentication while also providing a "Forgot Password" link for account recovery.

***Figure 3.*** *Login Form*

**SecurityQuestionForm**

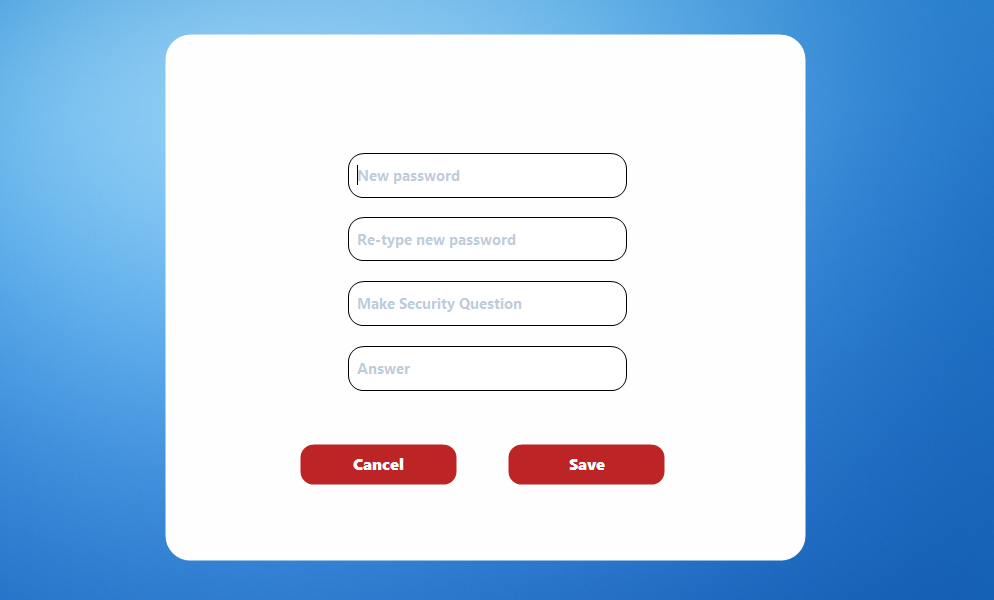
This form is the first step in password recovery. After confirming they wish to reset, the user is prompted to answer their pre-configured security question. This ensures that only the legitimate user can proceed.



***Figure 4.*** *SecurityQuestionForm*

**Password Reset Form**

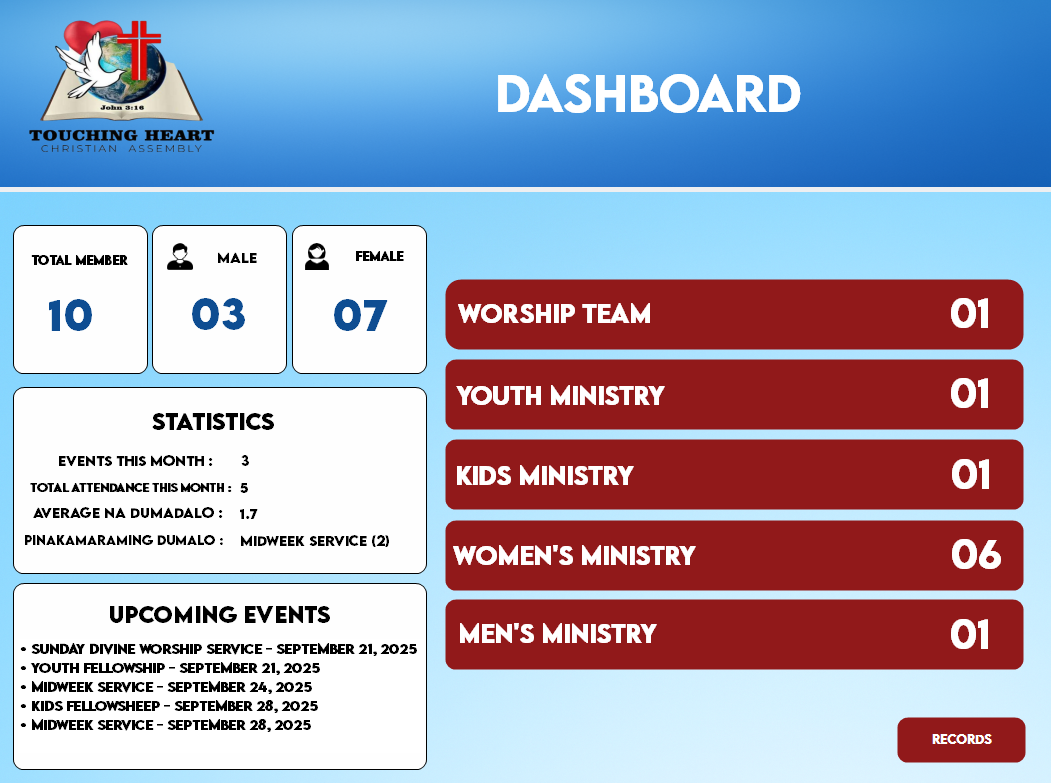
 Upon successful verification, this form allows the user to set a new password. To enhance security, the system requires the user to re-type the new password for confirmation. It also mandates setting a new security question and answer for future recovery needs before the changes can be saved.

****

***Figure 5.*** *Password Reset Form*

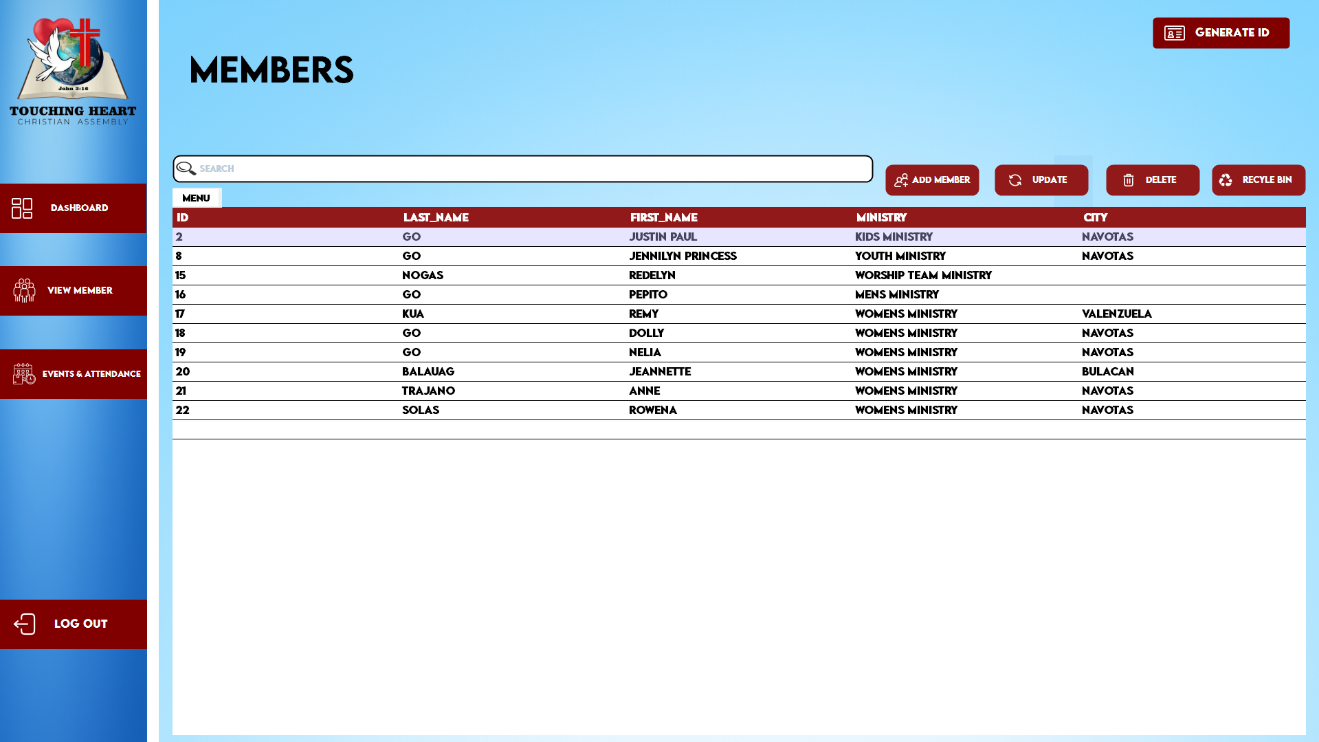
**Dashboard Form**

The Dashboard is the central hub of the application, providing administrators with an at-a-glance overview of key church metrics upon logging in. It presents real-time statistics, including the total number of members and a gender breakdown. It also features a summary of member distribution across various ministries. Navigation buttons provide quick access to view the full member list and event details.

  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
***Figure 6.*** *Dashboard Form*

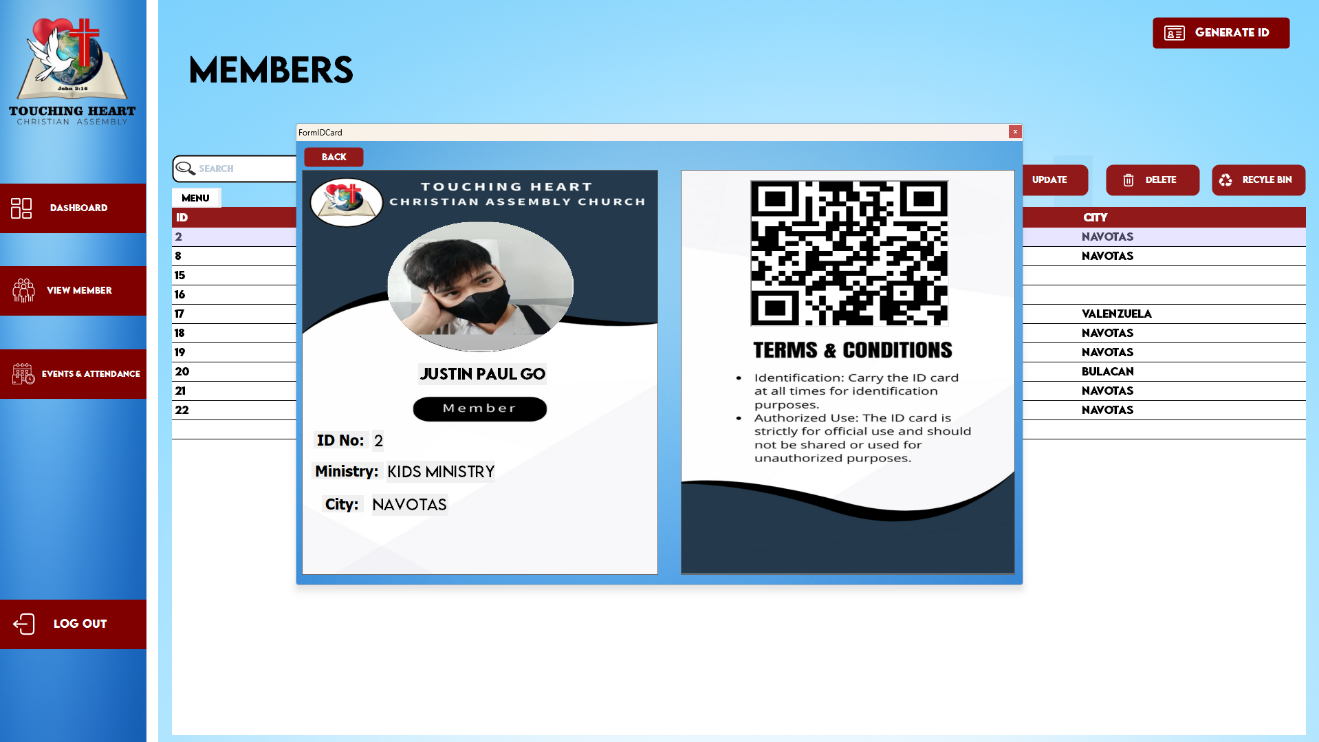
**Member List Form**

This form provides a comprehensive interface for managing the church's member database. A powerful search bar at the top allows for quick filtering of records. The main data is displayed in a detailed grid format, showing all member information. A dedicated sidebar offers quick access to all primary functions: returning to the Dashboard, updating or deleting selected records, adding a new member, managing attendance, and generating an ID card.

  
  
  
  
  
  
  
  
  
  
  
  
***Figure 7.*** *Member List Form*

**FormIDCard.Form**

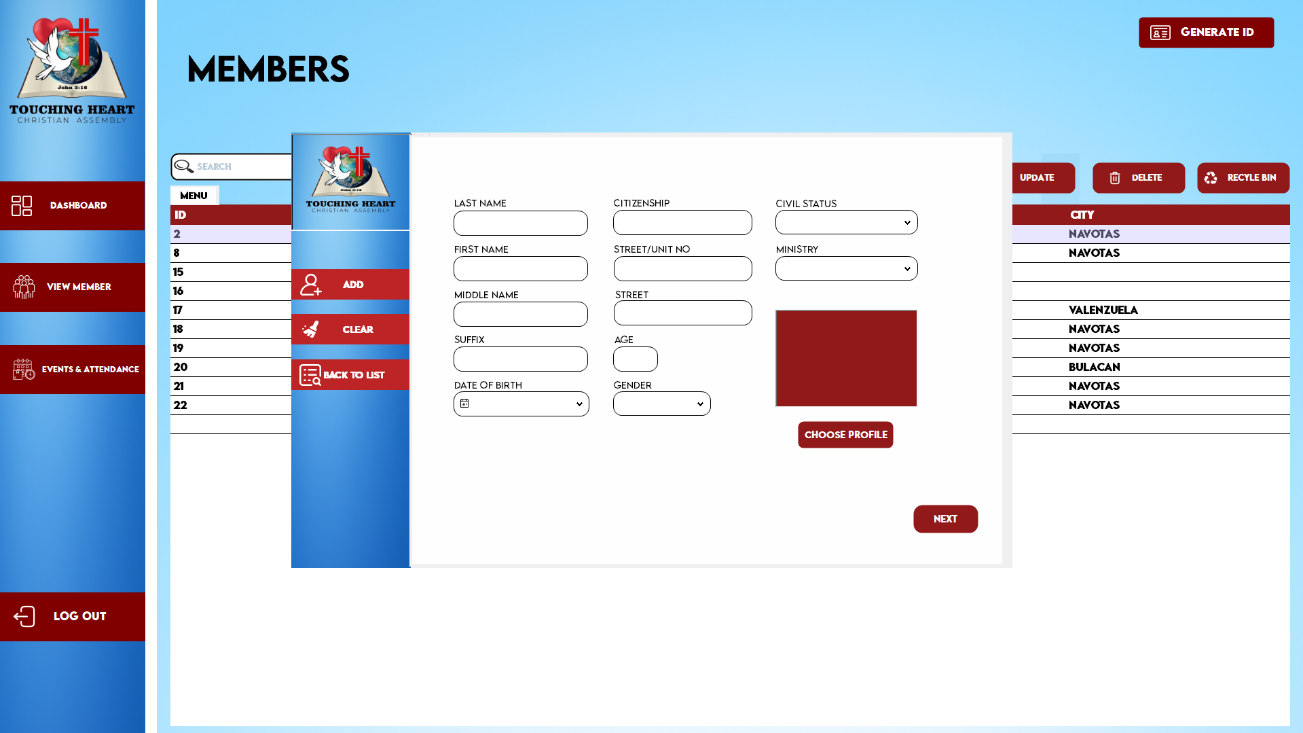
The FormIDCard.Form is a screen that shows the digital ID card of a church member. It displays the member’s personal details along with their profile photo, almost like a real ID card.

****

***Figure 8.*** *FormIDCardForm*

**Fill Up Form (Add Member)**

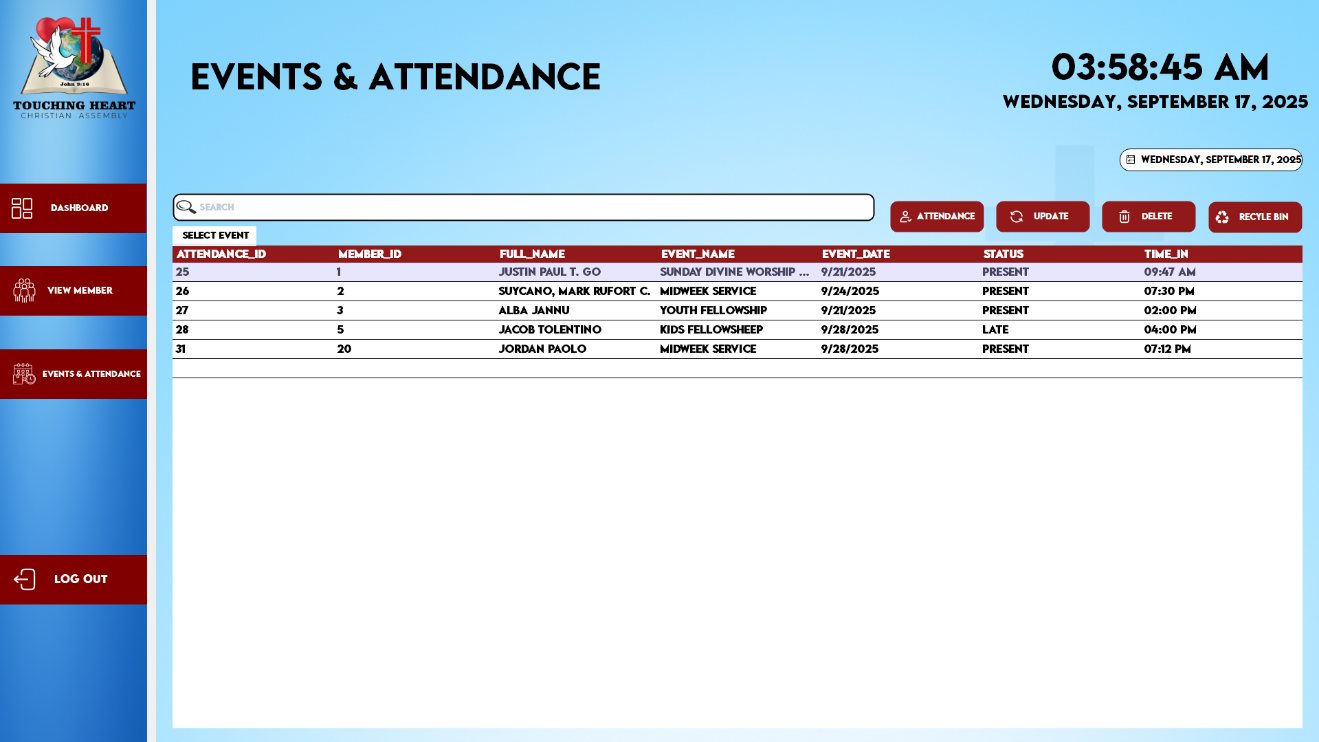
The Fill Up Form is the part of the system where new church members are registered. It allows the admin or authorized user to add a member’s personal details into the database.

****

***Figure 9****. Fill Up Form*

**Events & Attendance Form (Attendance)**

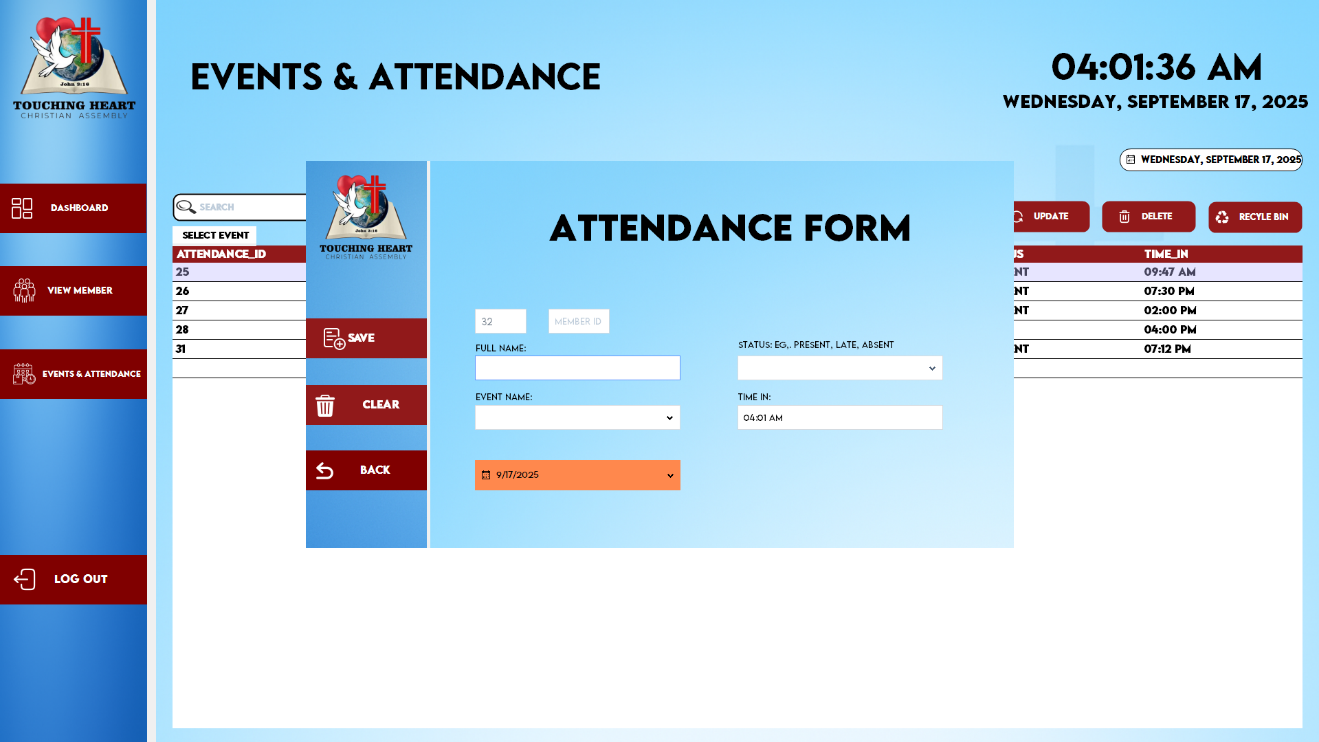
The Attendance Form is the part of the system where the attendance of church members is recorded and monitored during events, services, or gatherings.

**

***Figure 10****. Events & Attendance Form*

**Attendance Form (Attendance Fill up)**

The AttendanceForm2 (Attendance Fill up) is the form used to input or encode member attendance for a specific event. Unlike the first Attendance Form that mainly shows records, this one focuses on the actual filling up and updating of attendance data.

****

***Figure 11****. Attendance Form*

**COMPONENT DESIGN**

The Church Member Event Tracking System is divided into modular components to ensure maintainability, scalability, and clarity in development. Each component handles a specific responsibility and interacts with others through clearly defined interfaces.

**User Authentication Module**

This module provides a secure login system for administrators. It validates usernames and passwords against the database to ensure only authorized users can access the system. It also includes a security question mechanism for account recovery.

**Admin Dashboard**

The Admin Dashboard serves as the control center for system administrators. It displays important statistics such as the number of registered and active members, and a list of upcoming events. It also provides shortcuts to access other key modules, helping administrators quickly monitor church operations.

**Member Management Module**

This module allows staff to register new members, update existing member information, and search or filter member profiles. It also includes the ability to upload and store profile pictures. The module keeps all member records organized and easily accessible.

**Event Management & Attendance Tracking Module**

The Event Management Module is used to create and manage church events. Staff can enter event details such as name, date, time, location, and description. Events can also be edited or canceled as needed, making it easier to plan and coordinate church activities This module records member attendance during events. It allows staff to select an event, view the list of members, and mark who was present or absent. Attendance records are saved with links to both the event and the member, helping track participation over time.

**Database Layer**

The Database Layer stores all system data, including member profiles, event information, attendance records, and user credentials. It uses a local SQL database (such as MySQL or SQLite) and communicates with the application through secure queries. This layer ensures that data is centralized, consistent, and reliably stored**.**

**Dependency Management and Interaction Between Components**

The Church Member Event Tracking System is built with modular components that interact through a shared database. The User Authentication Module controls access, allowing only authorized users to enter the system. Once logged in, users are directed to the Admin Dashboard, which summarizes data from the member, event, and attendance modules. Each module; Member Management, Event Management, and Attendance Tracking perform specific tasks but depends on accurate data from each other to function properly.

The Member Management Module stores member details, which are used by the Attendance Module to track who attends events. The Event Management Module handles event creation and updates, which are also needed by the attendance system. All components rely on the Database Layer, which handles secure data storage and retrieval using SQL. This structure ensures smooth interaction between parts of the system while keeping responsibilities clear and manageable.

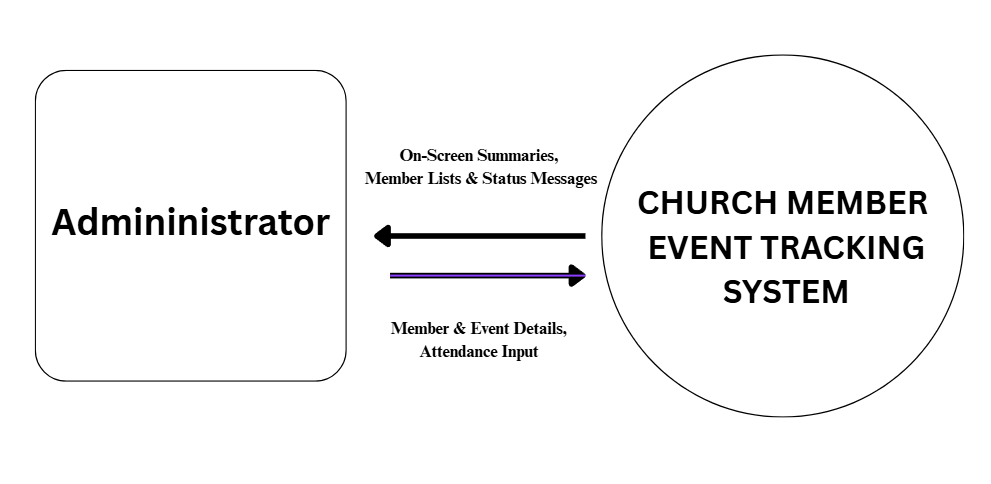
**DATA FLOW DIAGRAM**

The Data Flow Diagram (DFD) provides a visual representation of how information moves through the Church Member Event Tracking System. It illustrates the system's processes, the data they handle, and where that data is stored.

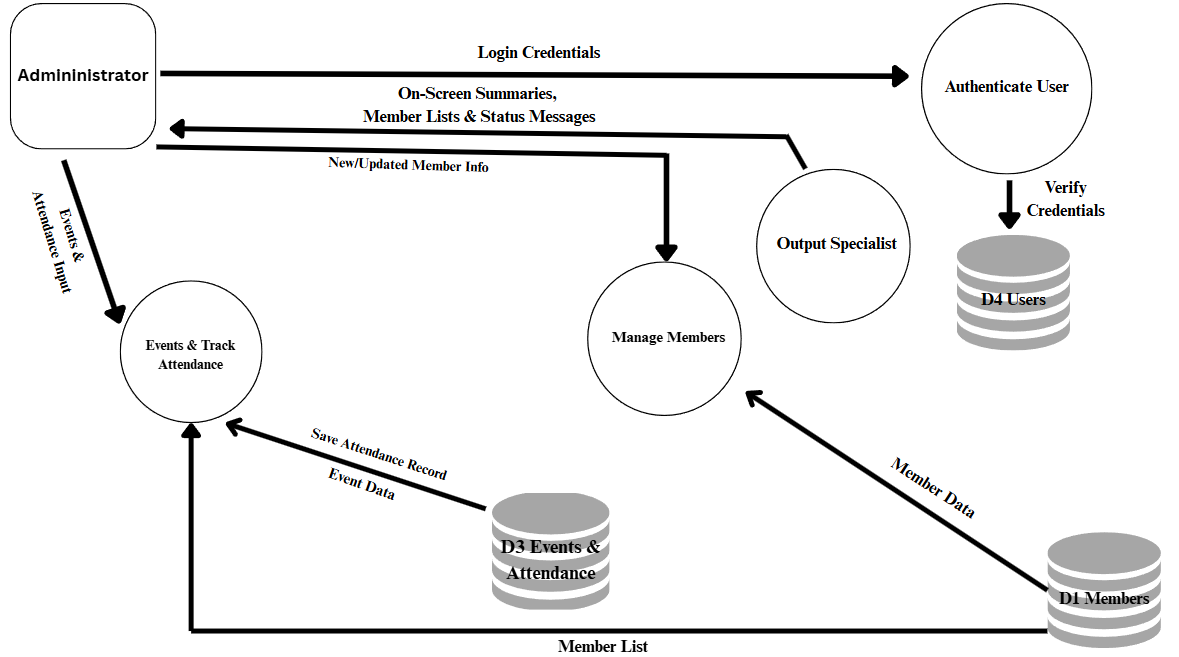
The Context Diagram (Level 0) presents the simplest, highest-level view of the system. It shows the entire application as a single process and highlights its interaction with its primary external entity.

In this diagram, the sole external entity is the Administrator. They provide all data inputs, such as new member details and attendance records. In return, the system processes this data and displays all outputs directly to the Administrator on-screen. These outputs include member lists, dashboard statistics, and attendance summaries. The subsequent presentation of this on-screen information to the Church Leaders is a manual process that occurs outside the system's boundaries.

.

**DFD - Context Diagram (Level 0)**

***Figure 12.*** *Data Flow Diagram (Level 0)*

****

**DFD - Context Diagram (Level 1)**

***Figure 13.*** *Data Flow Diagram (Level 1)*

**External Entities (Data Sources/Destinations)**

The Church Member Event Tracking System, while a standalone application, interacts with several external entities for data input and output. The main data source is the church administrator, who manually inputs all member information, event details, and attendance records into the system. Often, this data originates from physical external documents like printed registration forms or attendance sheets, which serve as references for data entry. For output, the system retrieves and stores member profile pictures from the local file system, acting as both a source and destination for image files. While the system displays all reports and summaries on-screen, this processed information is ultimately intended for the church leaders, who use it for planning and decision-making.

**Administrator/Staff**

* They are the primary and only direct users of the system.
* They are responsible for inputting and managing all data, including member details, event information, and attendance records.
* They receive and view all system outputs on-screen, such as the dashboard statistics, member lists, and attendance summaries.

**Member**

* Members are the subjects of the data managed within the system; they are not direct users.
* Their personal details, ministry affiliations, and event participation are digitally recorded and tracked.
* They may receive outputs indirectly when an administrator shows them their on-screen digital ID card or confirms their registration details from the system.

**Database (MemberInfo DB)**

* The main data storage where all member, event, attendance, and user information are saved.
* Acts as the central destination and source for all stored records.

**Ministry/Church Leaders**

* They are the beneficiaries of the information processed by the system; they are not direct users.
* They receive summarized information about events, attendance, and member participation verbally or by viewing the on-screen dashboard and reports presented by the administrator.
* They use this information for strategic decision-making and planning church activities.

**Data Flows**

Data Flows describe how information moves and is processed within the system. Key data such as member details, event schedules, and attendance records flow from the administrator into the database for storage. The system then processes this data and presents outputs like reports, digital ID cards, and summaries on-screen for the administrator and church leaders. Additionally, login and security details flow between the administrator and the database to control access.

**Member Information Flow**

* From Administrator/Staff → System → Database:  
  Admin enters and manages member details (name, gender, address, etc.).
* From Database → System → Administrator:  
  The system retrieves and displays member lists, individual profiles, and the on-screen digital ID card to the administrator.

**Event Information Flow**

* From Administrator/Staff → System → Database:  
  Admin inputs and manages event details (name, date, location, etc.).
* From Database → System → Administrator:  
  The system provides on-screen event lists and schedules to the administrator.

**Attendance Flow**

* From Administrator/Staff → System → Database:  
  Attendance is recorded (member, event, status: present/absent).
* From Database → System → Administrator:  
  Attendance reports and summaries are displayed on-screen to the administrator.

**User Login / Security Flow**

* From Administrator/Staff → System → Database:  
  Username, password, and security question/answer are verified.
* From Database → System → Admin:  
  Access is granted or denied depending on credentials.

**Reports Flow**

* From Database → System → Administrator:  
  The system processes data from the database to generate various reports.
* From System → Administrator:  
  Reports such as attendance summaries and member lists are generated and displayed on-screen for the administrator.
* From Administrator → Church Leaders:  
  The administrator verbally communicates or presents the on-screen reports to the church leaders.

**SECURITY DESIGN**

Security Design ensures that the system protects member information and event records from unauthorized access. It uses features like a secure login with a unique username and password, and security questions for account recovery. Only authorized users (like admins and staff) can add or edit data, while sensitive records are stored safely in the database. Regular backups and access control keep the system reliable and secure.

**Authentication and Authorization Mechanisms**

These are the security rules of the system. Authentication confirms a user's identity (e.g., username and password), while Authorization determines what they are allowed to do. Together, they ensure that only the right people can access and operate the system.

**Authentication** - “Are you really who you say you are?”

This is the process of verifying the identity of the person using the system. For example, when you log in, the system checks if the username and password you entered match a valid account stored in the database. The security question feature for password recovery is also part of the authentication process.

**Authorization** - “What are you allowed to do?”

Once a user is authenticated, the system authorizes them to perform actions. In this system, a single-role model is implemented. Any user who successfully logs in is considered an Administrator and is granted full authorization to access all features: adding and managing members, creating events, tracking attendance, and viewing all records. This ensures that only trusted individuals can operate the system, but does not differentiate permissions between them.

**Data Protection Measures**

The system implements several key measures to keep data safe. Sensitive information, particularly login credentials, is stored within the database and is protected from public access. The primary protection comes from secure database management, ensuring that only the application itself can interact with the data tables. Additional protection measures include strict access controls enforced by the login system, which prevents unauthorized users from viewing or modifying any records. The system is also designed as a standalone, offline application, which significantly reduces the risk of external online attacks. Finally, implementing a regular backup routine is a critical part of the strategy to safeguard against accidental data loss from hardware failure or user error. These safeguards ensure that member and event details remain private and secure within the church's local computer.

**PERFORMANCE DESIGN**

The system is built to run smoothly and quickly, even when handling many member records and events. It uses an optimized database structure, efficient SQL queries, and clean code to reduce delays and avoid crashes. Regular maintenance, data indexing, and resource management ensure that the system stays responsive, reliable, and able to handle daily church activities without slowing down.

**Performance Requirements and Objectives**

* Handle a growing number of member and event records without slowing down.
* Process attendance updates and searches quickly.
* Load forms, reports, and dashboards within a few seconds.
* The main objective is to keep the system fast, reliable, and responsive so staff can complete their tasks efficiently.

**Strategies for Optimizing System Performance**

* **Efficient Database Queries**. Use optimized SQL queries and indexing to make searches and reports faster.
* **Resource Management**. Minimize memory usage by closing unused forms and releasing resources.
* **Scalable Design**. While it’s a local desktop system, the structure allows future upgrades (ex., moving to a server or cloud if needed).

**Performance Testing Plan**

* **Load Testing.** Simulate a large number of member and event records to check if the system stays responsive.
* **Stress Testing.** Push the system beyond normal usage (e.g., bulk attendance updates) to see how it handles pressure.
* **Response Time Measurement.** Measure how fast forms load, searches return results, and reports are generated.
* **Optimization Review.** Identify bottlenecks and adjust queries, indexing, or code where needed.

**ERROR HANDLING AND LOGGING**

The Church Member Event Tracking System handles errors smoothly by showing clear, user-friendly messages instead of technical codes, preventing the application from crashing during use. It also keeps detailed error logs for administrators, making it easier to trace problems, maintain reliability, and ensure smooth operation of member, event, and attendance records.

**Error Handling Mechanisms and Strategies**

The system includes built-in error handling to prevent crashes and guide users when something goes wrong. Instead of showing confusing technical errors, it will display clear and friendly messages.

* **Validation Checks**. Input fields (e.g., phone number, date) are validated before saving to avoid invalid data.
* **Graceful Failure**. If an error occurs, the system will continue running instead of shutting down.
* **Fallback Options**. When possible, the system provides alternatives (e.g., retrying a failed operation or using cached data).

**Logging Requirements and Specifications**

The system maintains a secure error log file or database table to track technical issues for developers and administrators.

* Timestamp of the error
* Module or form where the error occurred
* Error type and description
* User actions leading to the error. These logs help in troubleshooting, system maintenance, and improving performance over time. Only administrators or authorized staff can access them to keep data secure.

|  |  |  |
| --- | --- | --- |
| **Error Code** | **Error Message (User-Friendly)** | **Description / When It Happens** |
| ERR-001 | Invalid login credentials. Please try again. | Shown when a user enters the wrong username or password. |
| ERR-002 | Cannot connect to the database. Please check your connection. | Appears when the system fails to connect to the SQL database. |
| ERR-003 | Some required fields are missing. Please complete all fields. | Triggered when the user leaves mandatory fields empty. |
| ERR-004 | You do not have permission to access this feature. | Shown when a staff account tries to access admin-only modules. |
| ERR-999 | An unexpected error occurred. Please contact the administrator. | Used as a fallback for unknown or unhandled errors. |

**Error Codes and Messages**

The system uses simple and understandable error messages for users, paired with error codes for developers.

**Table 1.** Error Codes and Messages

**THIRD-PARTY INTEGRATIONS**

The Church Member Event Tracking System is mainly a standalone VB.NET desktop application with an SQL database, but it also makes use of third-party tools and libraries to improve functionality and user experience. These integrations allow the system to go beyond basic data storage by supporting features like exporting, reporting, email notifications, and attendance tracking.

**List of Third-Party Services or APIs Integrated into the System**

The Church Member Event Tracking System is a self-contained application built entirely on the VB.NET Framework and a local SQL Database. All core functionalities, including the user interface, data management, validation, and the generation of the on-screen digital ID card, are handled natively within the application using built-in VB.NET libraries. The system does not rely on external third-party services or APIs for its primary operations, ensuring its stability and offline capability.

**Description of integration points and data exchange formats**

The primary integration point for the Church Member Event Tracking System is its database layer. The VB.NET application communicates directly with the local SQL database using standard SQL queries (INSERT, UPDATE, SELECT, DELETE). All data is exchanged and stored in a structured relational format of tables, rows, and columns. All other functionalities, including the on-screen display of reports and the digital ID card, are handled natively within the application. As a self-contained system, it does not export data to external file formats like PDF or Excel, nor does it connect to external APIs.

.

**DEPLOYMENT PLAN**

The Church Member Event Tracking System will be deployed as a standalone VB.NET application connected to an SQL database. The deployment involves installing the SQL database engine, configuring tables, setting up the .NET runtime, and installing the application on client machines. It requires a Windows 10 (or later) system with at least a dual-core CPU, 4GB RAM, and 500MB storage. Configuration management includes secure database connection strings and regular backups, while version control is handled through Git to track updates and ensure smooth maintenance.

**Overview of the Deployment Process**

* **Install SQL Database Engine (SQL Server/MySQL):** This is where all member, event, and attendance records will be stored.
* **Configure Database**: Create the necessary schema, tables, and insert initial setup data so the system can function properly.
* **Install .NET Framework / Runtime**: Required so the VB.NET application can run on the client machine.
* **Deploy VB.NET Executable (.exe):** Copy and set up the application file, then connect it to the database.
* **Testing**: Run the system after setup to verify that features like member registration, events, and reports are working correctly.

**Hardware and Software Requirements (Client System)**

* **Operating System**: Windows 10 or later - provides compatibility with VB.NET applications.
* **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.
* **Software Dependencies**: .NET Framework/Runtime and SQL Server/MySQL - core requirements for the system to operate.

**Configuration Management and Version Control Procedures**

* **Secure Database Connection:** Database connection strings are managed securely, ensuring that sensitive credentials are not exposed within the application's code.
* **Database Backup:** A clear procedure for performing regular database backups is established to prevent data loss and ensure data integrity.
* **Source Code Versioning:** The project's source code is managed using a version control system. This allows for systematic tracking of all modifications, creating a stable development history.
* **Version Releases:** Each release of the application is assigned a unique version number (ex., v1.0, v1.1). This standard practice helps in documenting changes, managing future updates, and providing version-specific support.

**MAINTENANCE AND SUPPORT**

The Church Member Event Tracking System will be maintained through regular database backups, system monitoring, and performance checks to ensure smooth operation. Software updates, patches, and bug fixes will be applied when needed, with version control used to manage changes safely. For unresolved issues, an escalation process will be followed, starting from user support, then to the system administrator, and finally to the developer if technical fixes are required.

**Guidelines for System Maintenance and Support**

* Perform regular database backups to prevent data loss.
* Monitor system performance and storage usage.
* Provide basic user support for common issues (e.g., login problems, report generation).

**Procedures for Handling Software Updates, Patches, and Bug Fixes**

* Test all updates on a separate environment before applying to the live system.
* Use version control to track all code changes, allowing for organized updates and safe rollbacks to a previous stable version if a new update causes issues.
* Apply patches and bug fixes during scheduled maintenance windows to minimize downtime.

**Escalation Process for Resolving Issues**

* Level 1: User reports issue to the system administrator.
* Level 2: If unresolved, the administrator checks logs and database for technical problems.
* Level 3: If still unresolved, the issue is escalated to the developer for debugging and code fixes.

**REVISION HISTORY**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 2025-08-31 | 1.0 | Initial draft of the Church Member Event Tracking System SDD | Go, Justin Paul T. |

This section records all changes made to the Church Member Event Tracking System SDD, including the date, version, description of updates, and the author responsible. It ensures proper documentation of revisions for tracking and accountability.

**Table 2**. *Revision History*