**TECHINICAL 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 serves as a complete technical guide for the Church Member Event Tracking System. It explains the system’s design, installation, configuration, usage, and maintenance. The goal is to provide developers, administrators, and end-users with clear instructions to ensure proper implementation and smooth operation of the system.

This project is a Church Member Event Tracking System. The main purpose of this system is to help the church manage records of members, ministries, and attendance in a faster and more organized way. Before, most of these tasks were done manually by writing details in paper notebooks or logbooks. This process often took a lot of time and could lead to errors, lost files, or missing details. By creating this digital system, these problems are solved because all information can be stored safely in the computer, making the work faster, more accurate, and more secure. The system was developed using VB.NET and SQL Server. These tools allow the system to store and manage data properly while also keeping it safe. The design of the system makes it simple and easy to use, even for people who may not be very familiar with computers. It provides features such as adding new members, recording their attendance, checking which ministry they belong to, and tracking their involvement in church events. With these features, the system becomes a very useful tool for church administrators and staff. This technical documentation serves as a complete guide for everyone who will use or maintain the system. It explains how the system works, how to install it, how to set it up, and how to troubleshoot when there are problems. Both technical users and non-technical users can use this document. Developers can use it as a guide to understand the system’s design, database structure, and code. Administrators can use it to help them during installation, configuration, and fixing of errors. End-users such as church staff can also use it to learn how to perform daily tasks like adding new members, recording attendance, creating reports, or updating passwords and security questions.

The purpose of this document is not just to explain what the system does, but also to show how and why each part of the system was made. This makes it easier for future developers to improve or update the system if needed. It also makes it clear for administrators how they can manage it properly, and for end-users how they can use it without confusion. By giving one complete reference, the system can be maintained and used smoothly over time. The Church Member Event Tracking System is mainly designed to help the church become more organized and accurate in keeping its important records. It allows administrators to check the details of every member, monitor their participation in ministries, and keep track of their attendance during events and services. Since everything is digital, information can be updated anytime and retrieved quickly whenever it is needed. This reduces mistakes, saves time, and makes sure that all records are kept safe and easy to access.

In simple terms, this project provides a solution to the common problems of manual record-keeping in churches. By using this system, the church can focus more on its mission and activities while having peace of mind that all important data about its members and events are stored securely and can be managed efficiently.

**Overview of the Software System**

The system is designed as a Church Member Information and Event Tracking System that allows church administrators to keep accurate records of members, monitor their participation in various ministries, and track attendance during events or services. Traditionally, such tasks are done manually through paper records, which can be time-consuming and prone to errors. This system addresses those problems by providing a digital platform where data can be stored, updated, and retrieved efficiently.

* **Member Management.** Adding, updating, and viewing member information such as name, gender, address, contact number, and assigned ministry.
* **Event Management.** Creating and managing church events with details like event name, date, location, and description.
* **Attendance.** Recording member attendance during events and services, with the ability to mark who is present or absent.
* **User Management.** Allowing administrators to manage system, and security features such as, passwords, re-type new password and security questions.
* **ID Generation.** Creating digital ID cards for members with their photo, assigned ministry, and residence location.

The system is built using VB.NET with SQL Server as the database. The application runs on Windows and is designed to work as a standalone system, meaning it can be installed on a single PC without requiring an internet connection. This makes it highly practical for small to medium-sized churches that do not yet have large-scale IT infrastructure but want to digitize their record-keeping process.

**Scope of the Technical Documentation**

This documentation covers every major aspect of the system, from setup to daily use. It is structured in such a way that different users can easily find the information they need.

* **System Overview.** A description of how the system is structured, including its components and deployment setup.
* **Installation Guide**. Step-by-step instructions for installing the software on a Windows computer, including requirements and dependencies.
* **Configuration Guide**. Detailed instructions for customizing the system, setting up security features, and adjusting parameters based on church needs.
* **Database Documentation**. Technical references for developers, showing how data is stored, how the database is structured, and how system functions interact.
* **User Manual**. A guide for end-users, explaining how to use the software for everyday tasks.
* **Troubleshooting and Maintenance**. A section for administrators to diagnose problems, apply fixes, and maintain the system over time.
* **Testing Documentation**. A record of how the system was tested, including test plans, cases, and results.
* **Revision History and Approval**. A log of changes made to the documentation and sign-off from key stakeholders.

**SYSTEM ARCHITECTURE**

The system follows a client-server architecture where the front-end is developed in VB.NET (Windows Forms) and the back-end uses Microsoft SQL Server for data storage.

The Church Member Event Tracking System is designed with VB.NET as the main program and SQL Server as the database. It allows storing and managing information such as member details, attendance, and events in one place. The system can run on a single computer or be shared through a local network. Its design makes sure that data is organized, secure, and easy to access by church administrators and users. The Church Member Event Tracking System is designed to make the management of church records simple, organized, and secure. It replaces the old way of keeping records on paper by using a digital platform that stores all important information inside a database. The system is built with a three-layer architecture: the presentation layer, the business logic layer, and the data layer. The presentation layer is the part that the user sees and interacts with, such as forms for adding members, recording attendance, or viewing reports. The business logic layer is responsible for handling the rules and processes of the system, like checking if the input is valid, verifying login details, and saving records properly. The data layer is where all information is stored and managed inside the SQL Server database. This separation makes the system easier to maintain, safer to use, and more reliable over time.

The high-level components of the system work together to create a smooth flow of data. The first component is the User Interface, which is created using VB.NET Windows Forms. This allows church administrators and staff to interact with the system in a simple and user-friendly way. The second component is the Database, which is handled by SQL Server. The database contains all the records such as member details, ministries, attendance logs, user accounts, and security questions. The third component is the Application Logic, which acts as the bridge between the user and the database. It makes sure that when a user enters or requests data, the correct process is followed. For example, when adding a new member, the application logic checks if the required fields are filled out before saving it to the database. When logging in, it verifies the username, password, and security details before allowing access. These components interact with each other to make sure that all functions of the system work smoothly and correctly.

For deployment, the system is designed as a standalone desktop application that runs on a Windows computer. It requires VB.NET to run the application interface and SQL Server to store the database. Both can be installed on the same computer, making the system easy to set up for small churches that only need one computer to manage their records. In larger setups, the database can be installed on a separate server while the application is installed on multiple computers used by different staff members. This setup allows several users to connect to the same database over a local network, making it possible to share records in real time. Whether used as a single-user or multi-user system, the deployment remains simple and flexible, giving the church options depending on its needs and resources.

Overall, the system is designed to balance simplicity and reliability. Its architecture ensures that data is stored securely, its components work together in an organized way, and its deployment is flexible enough for both small and large church environments. This structure makes the Church Member Event Tracking System not only practical to use today but also adaptable for future improvements and expansions.

**INSTALLATION GUIDE**

**System Requirements**

**Hardware:**

* Processor: Intel i3 or higher
* RAM: Minimum 4GB
* Storage: At least 500MB free space

**Software:**

* Operating System: Windows 10 or a newer version
* Database Engine: Microsoft SQL Server Express 2019 (or newer)
* Framework: .NET Framework 4.7.2 or higher

**Step-by-Step Installation**

**Install SQL Server Express:**

* Download and run the SQL Server Express installer.
* Choose the "Basic" installation type.
* Accept the license terms and follow the on-screen instructions to complete the installation. Note the instance name (usually SQLEXPRESS)

**Set up the Database:**

* Open SQL Server Management Studio (SSMS).
* Connect to your local instance (e.g., (local)\SQLEXPRESS).
* In the Object Explorer, right-click on "Databases" and select "New Database."
* Name the database MemberInfo.
* Run the provided SQL script (schema.sql) to create all the necessary tables (Users, Members, Events, Attendance).

**Deploy the Application:**

* Copy the entire application folder (containing the .exe file and other supporting files) to a permanent location on the computer (ex., C:\ChurchSystem\).
* Create a shortcut for the main executable file (Church Event.exe) and place it on the Desktop for easy access.

**Configuration Settings**

* Database connection string: "Server=DESKTOP-XXXX\SQLEXPRESS; Database=MemberInfo; Trusted\_Connection=True;"
* Ensure SQL Server Authentication is enabled if using a username/password login.

**CONFIGURATION GUIDE**

This section provides detailed instructions for the initial setup and configuration of the Church Member Event Tracking System after its installation. Proper configuration is crucial for ensuring the application connects correctly to its database and that the initial administrative access is established securely.

**Configuring the Database Connection**

The most critical configuration step is ensuring the VB.NET application can communicate with the SQL Server database. The connection string, which contains the address and credentials for the database, is hard-coded within the application's source for simplicity and to prevent tampering by unauthorized users.

**Default Connection String:**

The system is pre-configured to use the following connection string format, which relies on Windows Authentication: Database connection string: "Server=DESKTOP-XXXX\SQLEXPRESS; Database=MemberInfo; Trusted\_Connection=True;”

**Steps for Developers to Modify (if needed):**

1. Open the project in Visual Studio.
2. Locate the module or class where the database connection is established (e.g., a DatabaseConnection module).
3. Update the Server parameter in the connection string to match the correct computer and instance name.
4. Recompile the application to apply the changes.

**Initial Administrator Account Setup**

For security reasons, the system does not have a public registration page. The very first administrator account must be created manually and directly within the database. This ensures that only an authorized person with database access can create the initial superuser.

**Step-by-Step Guide for Initial Account Creation:**

1. Open SQL Server Management Studio (SSMS).
2. Connect to the SQL Server instance where the MemberInfo database is located.
3. In the Object Explorer, expand the MemberInfo database, then expand the Tables folder.
4. Right-click on the dbo.Users table and select "Edit Top 200 Rows".
5. A new row will appear. Enter the following details for the first administrator:

* username: Choose a secure username (e.g., admin).
* password: Enter a strong initial password. Note*:* As passwords are not hashed in this version, it is critical that database access is strictly limited.
* security\_question: Provide a recovery question.
* security\_answer: Provide the corresponding answer.

1. Close the table editor. The record will be saved automatically. The administrator can now use these credentials to log into the application for the first time and can change their password and security question through the application's interface.

**Best Practices for System Configuration**

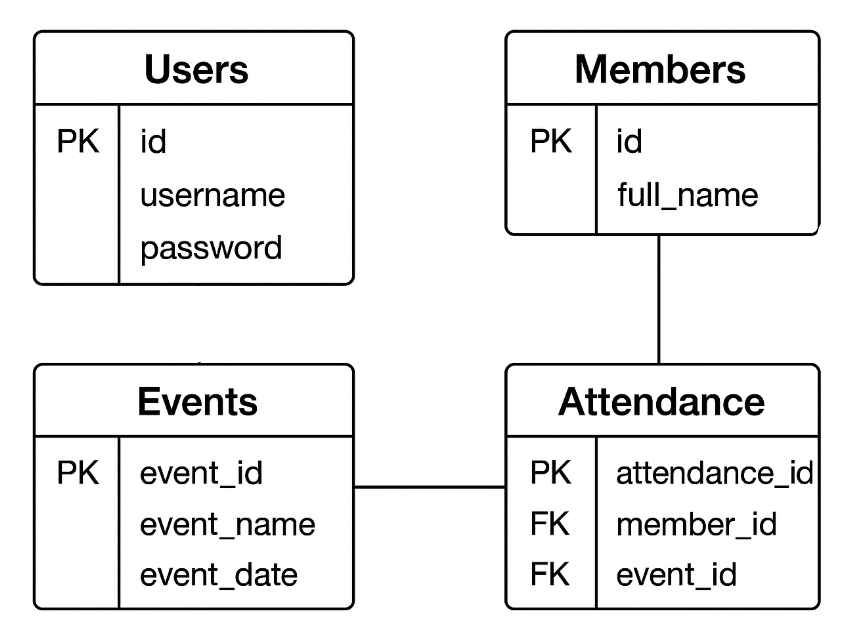
To ensure the long-term stability and security of the system, the following best practices are strongly recommended:

* **Limit Database Access:** The SQL Server instance should be configured to limit access. Only the machine running the application and authorized administrative personnel should be able to connect to the database.
* **Regular Backups:** While not a direct application configuration, establishing a routine for backing up the MemberInfo database is the most critical maintenance task. Backups should be performed regularly (e.g., weekly) and stored on a separate physical device (e.g., an external hard drive) to protect against hardware failure, data corruption, or ransomware.
* **Dedicated Machine:** For optimal performance and security, it is recommended that the system be installed on a dedicated computer in the church office that is used primarily for administrative tasks.

**DATABASE DOCUMENTATION**

The Members table contains detailed information about each church member such as name, birthdate, address, and ministry involvement. The Events table records details about church activities including event name, date, and location. The Attendance table acts as a link between Members and Events by storing the attendance record of each member, including their participation status and time of entry. The Accounts table manages user login credentials and roles, ensuring secure access to the system.

The Church Member Event Tracking System utilizes a relational database managed by Microsoft SQL Server. The structure of the database is defined by the Entity-Relationship Diagram (ERD) below, which illustrates the core entities and the relationships between them. The four main entities are: Users, Members, Events, and Attendance.

 **Figure 1:** Entity-Relationship Diagram

Each table is related through primary and foreign keys that establish connections. For example, the member\_id in the Members table links directly to the Attendance table, while the event\_id in the Events table also connects to Attendance. These relationships allow the system to efficiently generate attendance reports, member participation summaries, and event statistics.

To maintain data integrity and security, the system includes procedures for data migration and backup. During migration, existing records from manual files or other digital sources are validated before being transferred into the SQL database. Regular backups are scheduled to secure a copy of the database, preventing data loss in case of system errors or hardware failure. Backup files can be restored when needed to recover the most recent stable version of the system’s records.

**Description of Database Tables and Relationships**

* **Users Table:** This table is dedicated to system security and access control. It stores the username and password for each administrator. For simplicity and security, this system implements a single-role model, and this table is kept separate from the member data.
* **Members Table:** This is the master list of all church members. It contains detailed personal information for each individual, such as their full\_name, birthdate, address, and ministry involvement. Each member is assigned a unique member\_id which serves as the Primary Key (PK).
* **Events Table:** This table holds the records for all church activities and services. It includes details such as event\_name, event\_date, and location. Each event has a unique event\_id (PK).
* **Attendance Table:** This crucial table acts as a junction table (or bridge) that connects members to the events they participate in. It resolves the many-to-many relationship between Members and Events. It contains its own Primary Key (attendance\_id) and two Foreign Keys (FK): member\_id (linking to the Members table) and event\_id (linking to the Events table). It also stores the status of participation (e.g., Present, Late).

**Data Migration and Backup Procedures**

* **Data Migration:** For initial setup, existing member records from paper-based forms or other digital files must be manually entered into the system. It is critical that this initial data entry is done carefully to ensure accuracy and consistency. The system does not include an automated data import feature.
* **Backup and Restore:** To maintain data integrity and prevent data loss, a manual backup procedure must be established by the church administrator. This involves regularly creating a backup of the MemberInfo.mdf and MemberInfo\_log.ldf files and storing them on a separate, secure device (e.g., an external hard drive). In the event of a system or hardware failure, these backup files can be used to restore the database to its last known good state using SQL Server Management Studio.

**USER MANUAL**

Welcome to the User Manual for the Church Member Event Tracking System. This guide is designed to provide church administrators with a clear and comprehensive walkthrough of all the system's features. The following sections will explain the entire workflow in a narrative format, from logging into the system to managing member and attendance records, ensuring a smooth and intuitive user experience.

**Getting Started: Accessing the System**

To begin using the system, the administrator must first open the application, which will present the main Login Form. Here, the user is required to enter their assigned username and password into the respective fields. Clicking the Login button will validate these credentials and grant access to the system's main dashboard. For security purposes, it is highly recommended to log out after each session. This can be done by clicking the LOG OUT button located on the main sidebar, which will securely terminate the session and return the user to the Login Form.

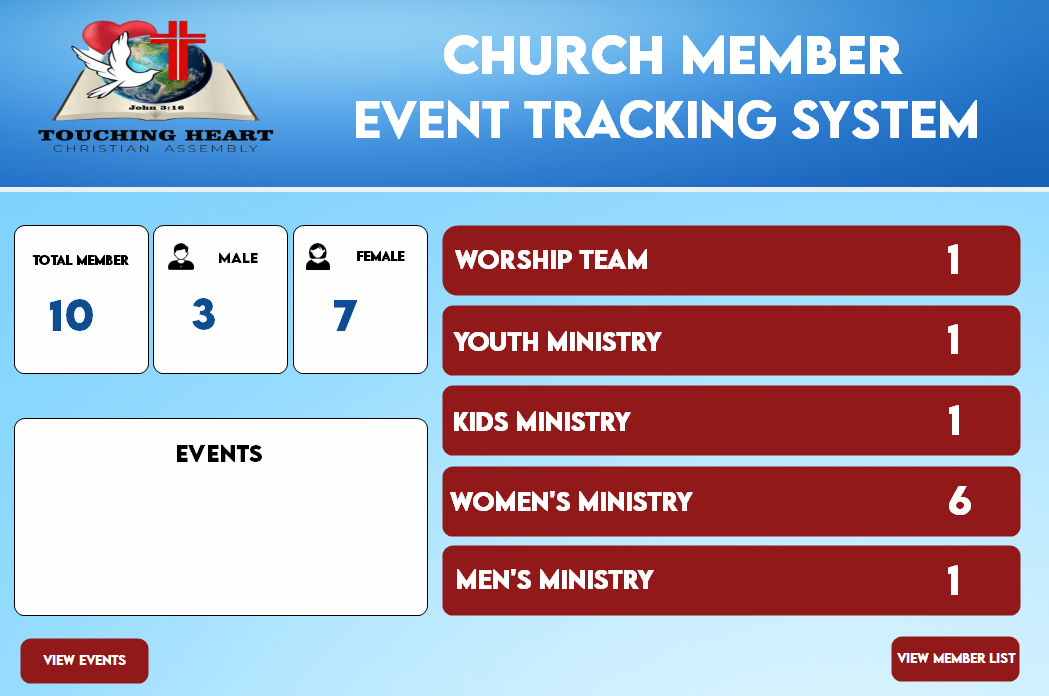


In the Events section, users can add upcoming church events by entering the event title, date, and location. Events are then automatically stored and can be connected to attendance records. The Attendance feature allows users to record member participation during services or special events. The AttendanceFillUp form can be used during live gatherings, where names of members are quickly marked as present or absent. This makes the tracking process faster compared to manual listing.

**Figure 2**. Login Form for Church Member Event Tracking System

**Navigating the Main Dashboard**

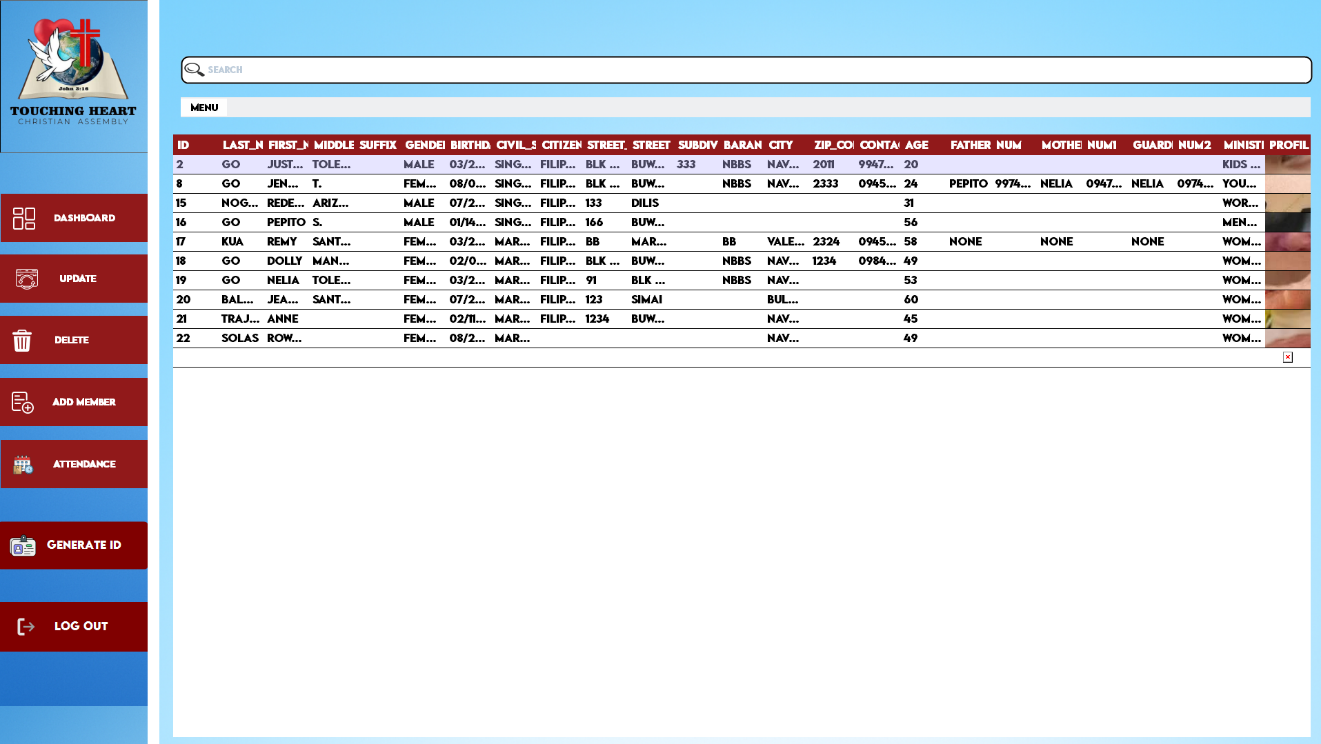
Upon a successful login, the user is immediately directed to the Main Dashboard, which serves as the central hub and control center of the application. This screen provides an at-a-glance overview of key church metrics, including real-time statistics on the total number of members and a demographic breakdown by gender. The dashboard also features a dynamic summary of member distribution across the various church ministries. From this central point, the administrator can easily navigate to other core modules by clicking the View Member List button to access the complete member database.

****

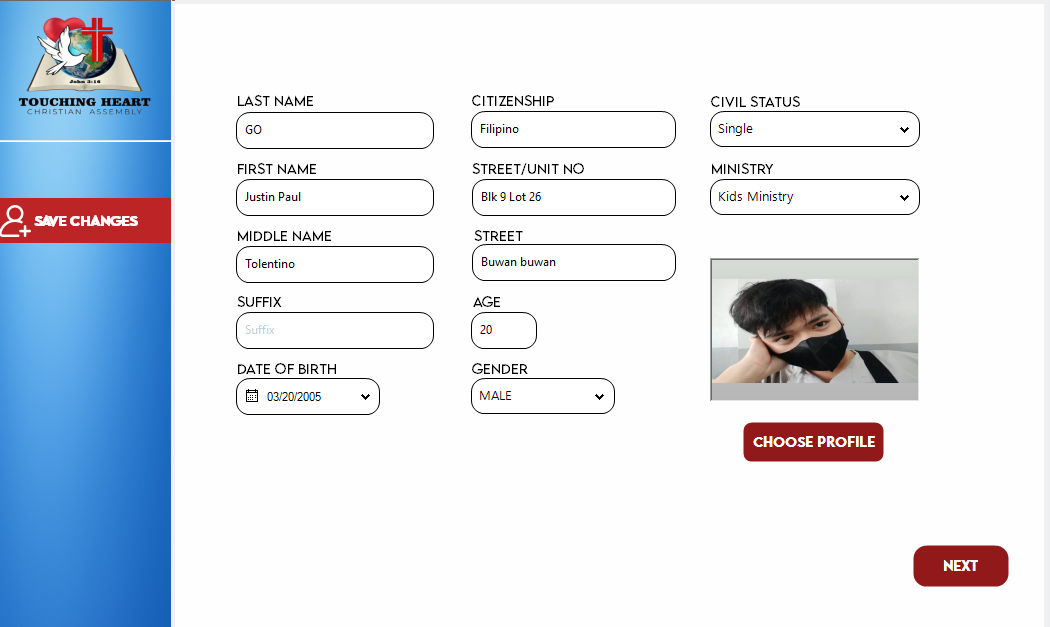
**Figure 3**. Dashboard Form for Church Member Event Tracking System

**Managing Member Records**

The management of all member information is handled within the Member List Form, which can be accessed from the dashboard or the main sidebar. This comprehensive interface displays all member records in a detailed and organized grid. A powerful search bar is located at the top of the screen, allowing the administrator to instantly filter the list by typing a member's name. To add a new member, the administrator simply clicks the ADD MEMBER button on the sidebar. This action opens the Fill Up Form, where all the new member's personal details, contact information, and ministry affiliation can be entered. An image can also be uploaded by clicking the "Choose Profile" button. Once all details are complete, clicking "Save Changes" permanently adds the new record to the database. For updating existing information, the process is just as straightforward. The administrator first selects a member from the list and then clicks the UPDATE button. The system will open the same form, but pre-filled with the member's current data, allowing for quick edits. To delete a record, the administrator selects a member and clicks the DELETE button, confirming the action in a subsequent prompt to prevent accidental removal.



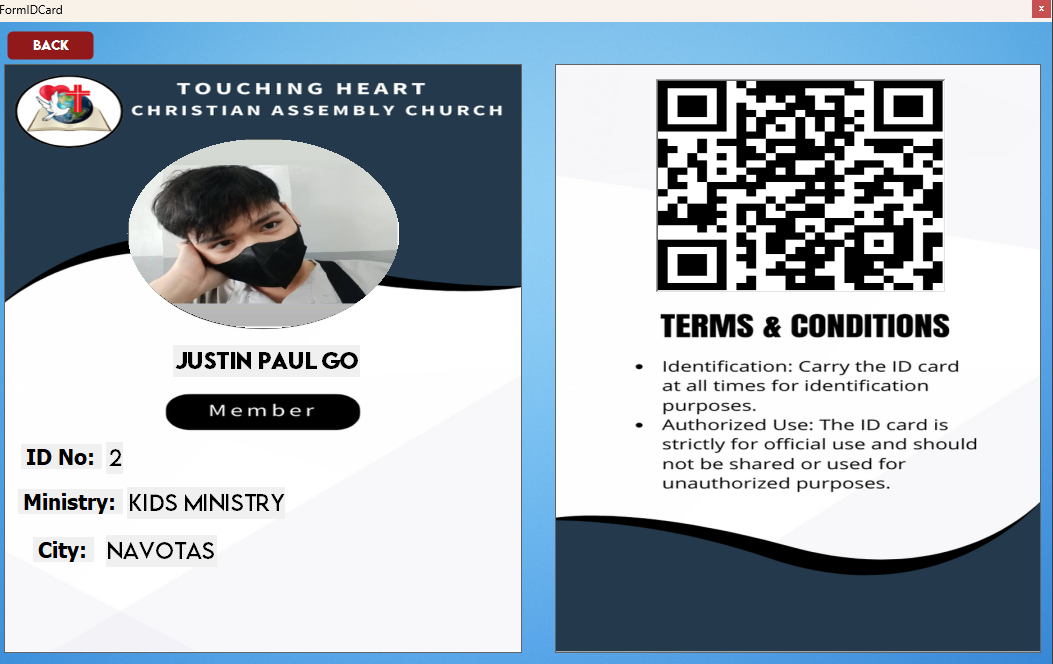
**Figure 4**. Member List Form for Church Member Event Tracking System

****

**Figure 5**. Fill Up Form for Church Member Event Tracking System

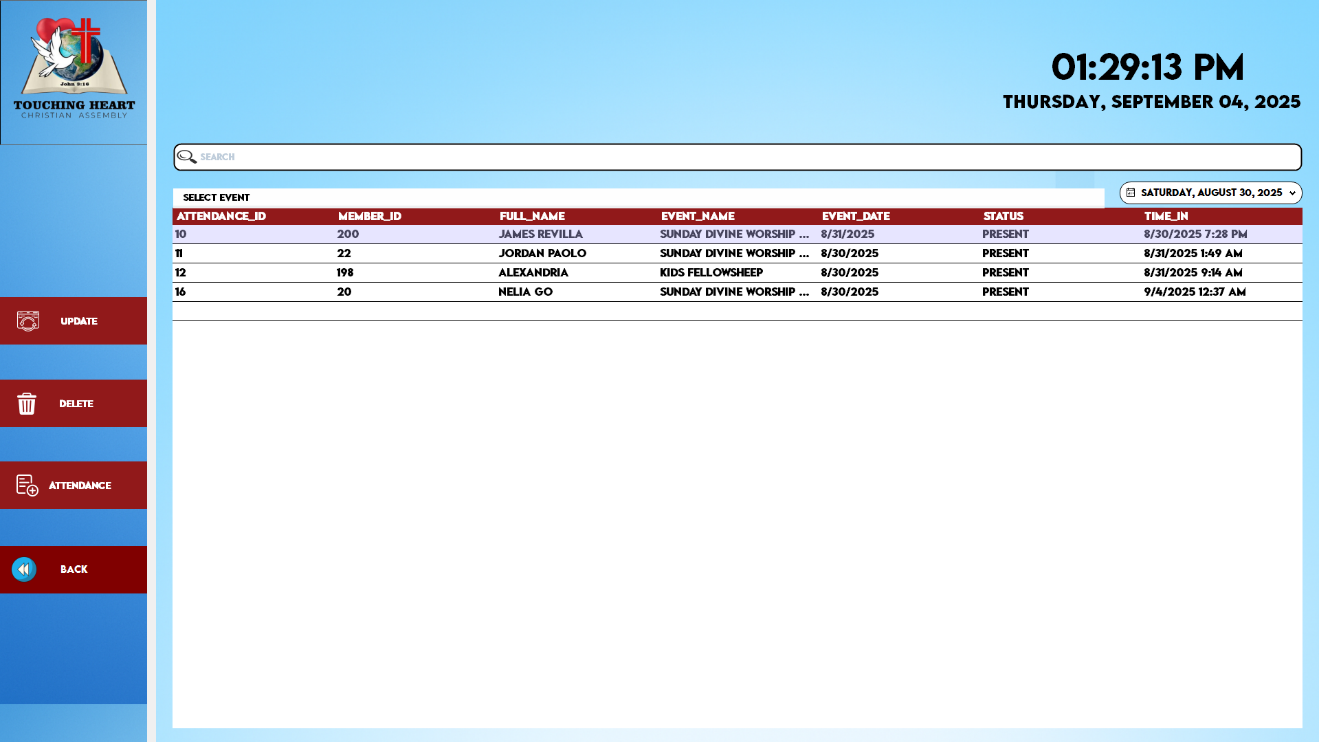
**Generating an On-Screen ID Card**

The system includes a feature to generate a professional, on-screen digital ID card for any registered member. From the Member List screen, the administrator selects a specific member and then clicks the GENERATE ID button on the sidebar. A new window will immediately appear, displaying a two-sided digital ID that includes the member's photo, full name, ID number, and other relevant church details, ready to be presented or viewed.

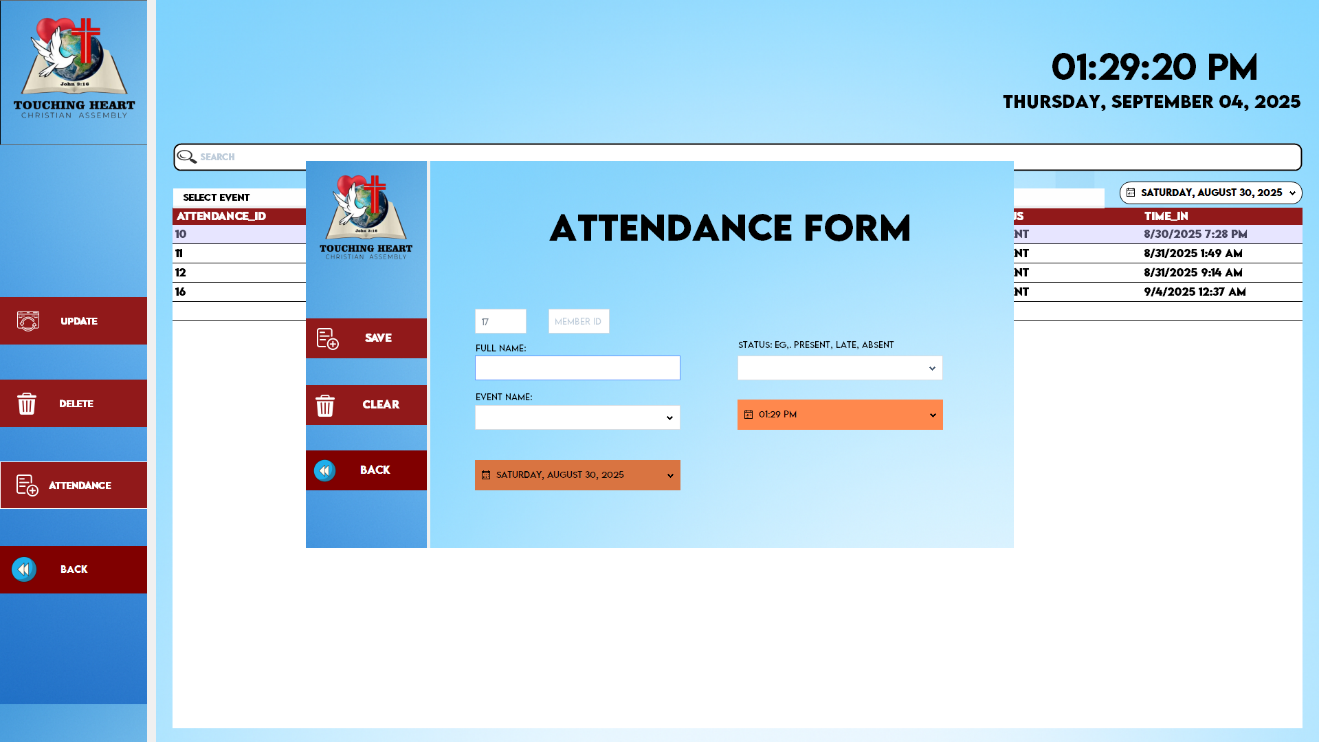
****

**Figure 6**. ID Form for Church Member Event Tracking System

**Managing Attendance Records**

 Tracking event participation is a core function of the system, managed through the Attendance Form. This screen, accessible via the main sidebar, displays all historical attendance records in a clear grid. The administrator can quickly find specific records using the search bar or filter the entire list to show attendance for a particular day by using the date picker tool. To record new attendance, the administrator clicks the main ATTENDANCE button, which opens the attendance entry form. In this form, the administrator enters the MEMBER ID, and the system automatically fetches the member's full name for verification. They then select the appropriate EVENT NAME from a dropdown list, set the attendance STATUS (e.g., Present, Late, Absent), and confirm the date and time. Clicking the SAVE button logs the attendance record into the database. A similar process is followed for updating an existing record, which begins by selecting a record from the list and clicking the UPDATE button.

**Figure 7**. Attendance Form for Church Member Event Tracking System

**  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
Figure 8**. Attendance Form2 for Church Member Event Tracking System

**TROUBLESHOOTING GUIDE**

This section provides solutions to common issues that may arise while using the Church Member Event Tracking System. It is designed to help the administrator diagnose and resolve problems quickly.

**Common Issues and Error Messages**

|  |  |  |
| --- | --- | --- |
| **Issue / Error Message** | **Possible Cause(s)** | **How to Fix** |
| Error: "Invalid login credentials. Please try again." | 1. Incorrect username or password was entered. The  2. Caps Lock key is on. | 1. Double-check the spelling of the username.  2.Carefully re-type the password, making sure Caps Lock is turned off.  3.If the problem persists, use the "Forgot Password" link to reset your password using your security question. |
| Error: "Cannot connect to the database." | 1. The SQL Server service is not running on the computer.  2. The computer is not connected to the local network (if the database is on a separate server). | 1. Restart the computer. This often restarts the SQL Server service automatically.   2. (Advanced) Open "Services" from the Windows Start Menu, find "SQL Server (SQLEXPRESS)," and click "Start" if it is stopped.   3. Ensure all network cables are properly connected. |
| Error: "A generic error occurred in GDI+.” | 1. This error is related to image processing, often occurring when saving a record with a picture that is corrupted or in an unsupported format. | 1. Try re-uploading the member's profile picture using a standard format like .JPG or .PNG.   2. Ensure the image file is not corrupted by trying to open it in a standard photo viewer.   3. If the error persists, try saving the record without a picture. |
| The application runs slowly or freezes. | 1. The computer has low memory (RAM) because many other programs are open.   2. The database has grown very large, and a specific search or filter is taking a long time. | 1. Close all other unnecessary applications (like web browsers, games, etc.) before using the system.  2. Restart the computer to clear its memory.   3. When searching, try to be more specific with your keywords to narrow down the results. |
| The Member's Full Name does not appear after entering Member ID in the Attendance Form. | 1. The Member ID entered does not exist in the database.   2. There might be a typo in the Member ID. | 1. Double-check the Member ID for any typing errors.   2. Go back to the Member List screen and use the search function to verify that the member and their ID exist. |

**Table 1**. Common Issues and Error Messages

**General Troubleshooting Steps**

If you encounter an issue not listed above, follow these general steps:

1. **Restart the Application:** Close the Church Member Event Tracking System completely and open it again. This resolves many temporary issues.
2. **Restart the Computer:** A full computer restart can fix problems with the database service or other background processes.
3. **Take a Screenshot:** If an error message appears, take a screenshot or write down the exact message. This information is very important for technical support.

**Contact Information for Technical Support**

If the issue cannot be resolved using this guide, please do not hesitate to contact the developer for technical support. Please provide the following information:

* A clear description of the problem.
* The steps you took that led to the error.
* The exact error message or a screenshot of the error.

**Developer Contact:**

* **Name:** [Go, Justin Paul T.]
* **Email:** [justingo006@gmail.com]
* **Contact Number:** [09947247515]

**CODE DOCUMENTATION**

This section provides an overview of the source code for the Church Member Event Tracking System. It is intended for developers who may need to maintain, update, or troubleshoot the application in the future.

**Code Structure and Organization**

The project is developed in **Visual Basic .NET (VB.NET)** using the Windows Forms framework. The source code is organized into multiple Form files, with each form representing a major screen or function of the application.

The primary forms and their responsibilities are as follows:

* **Form1.vb (Member Registration/Update Form):**
  + Handles the user interface and logic for adding a new member and updating an existing member's record.
  + Contains validation for input fields and manages the multi-page tab control for data entry.
* **Form2.vb (Member List Form):**
  + Displays the comprehensive list of all members in a DataGridView.
  + Contains the logic for searching, filtering by ministry, and initiating the Add, Update, and Delete operations.
  + Acts as a central navigation point to other features like ID Generation and Attendance.
* **Form3.vb (Login Form):**
  + The main entry point of the application.
  + Handles user authentication by verifying credentials against the Users table.
  + Provides access to the password recovery workflow.
* **AttendanceForm.vb:**
  + Displays the list of all historical attendance records.
  + Includes functionality to filter records by date, event, or search query.
* **AttendanceFillvb.vb:**
  + The data entry form for adding or updating a single attendance record.
  + Includes validation to prevent duplicate attendance entries for the same member in the same event.
* **DashboardForm.vb:**
  + The main screen after a successful login.
  + Contains the SQL queries and logic to calculate and display statistics (e.g., total members, gender breakdown, ministry counts).
* **Module1.vb (Database Connection Module):**
  + A central module that contains the reusable GetConnection() function.
  + This function holds the database connection string, making it easy to manage and update the database connection from one central location.

**Inline Comments and Key Logic**

The source code is documented with inline comments to explain complex or critical sections of logic. Comments are used to:

* Explain the purpose of key functions and subroutines: Each major Sub or Function begins with a brief comment describing what it does.
* Clarify complex SQL queries: Queries with multiple joins or conditions are explained.
* Highlight important validation logic: Sections that check for user input, prevent errors, or handle specific business rules (like the duplicate attendance check) are clearly commented.
* Mark areas for future improvement: TODO comments may be used to indicate sections that could be enhanced in future versions.

**Coding Standards and Conventions**

To ensure code readability and maintainability, the following standards and conventions were followed during development:

* **Naming Conventions:**
  + **Controls:** A three-letter prefix is used to denote the control type (e.g., txtLname for a TextBox, btnSave for a Button, dgvInfo for a DataGridView).
  + **Variables:** Camel case is used for local variables (e.g., studentIdToUpdate), while Pascal case is used for public properties and methods.
  + **Forms:** Form names are descriptive of their function (e.g., LoginForm, MemberListForm).
* **Code Formatting:**
  + Proper indentation is used to clearly show the structure of loops, conditional statements (If...Then), and Try...Catch blocks.
  + Code is logically grouped into regions where applicable (e.g., === FORM LOAD ===, === CRUD OPERATIONS ===) for easier navigation.
* **Error Handling:**
  + All database operations and potentially problematic code sections are wrapped in Try...Catch blocks to handle exceptions gracefully.
  + User-friendly error messages are displayed using a standardized ShowError() function, preventing the application from crashing and informing the user of the issue.
* **Database Interaction:**
  + The Using statement is consistently used for all SqlConnection, SqlCommand, and SqlDataReader objects. This ensures that database connections are always properly closed and disposed of, preventing resource leaks.
  + Parameterized queries are used for all SQL commands that involve user input. This is a critical security measure to prevent SQL Injection attacks.

**TESTING DOCUMENTATION**

This section documents the testing process conducted on the Church Member Event Tracking System to ensure its quality, functionality, and reliability before deployment.

**Test Plan Outlining Testing Objectives and Strategies**

The test plan for the Church Member Event Tracking System was designed to systematically validate the application's quality, functionality, and readiness for deployment. The primary objective of this phase was to rigorously verify that all functional requirements were implemented correctly and that the system met its specified non-functional attributes, particularly in performance, usability, and security. To achieve this, a multi-layered testing strategy was employed throughout the development lifecycle. The process began with Unit Testing, where individual code components, such as database connection functions and input validation logic, were tested in isolation to ensure their correctness. This was followed by Integration Testing, which focused on verifying the seamless interaction between different modules, for example, ensuring that a newly created member record correctly populates in the attendance form's selection list. Once all components were integrated, comprehensive System Testing was performed, simulating real-world administrative workflows from end to end, such as logging in, adding a new member, creating an event, and recording attendance for that event. Finally, User Acceptance Testing (UAT) was conducted by simulating the perspective of a church administrator to confirm that the system is intuitive, meets the practical needs of the end-user, and is fit for its intended purpose. All tests were executed in a controlled environment that mirrored the client's setup, utilizing a Windows 10 operating system, Microsoft SQL Server Express, and the .NET Framework 4.7.2 to ensure the results were accurate and relevant. This structured approach aimed to identify and resolve any defects prior to delivery, guaranteeing a stable, reliable, and effective software solution.

**Test Cases and Results**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Feature / Requirement Tested** | **Test Steps** | **Expected Result** | **Actual Result** | **Status** |
| **TC-001** | **FR-05: Secure Login** | 1. Enter a valid username and password  2. Click "Login." | The user is successfully authenticated and the Dashboard appears. | The user was successfully authenticated and the Dashboard appeared. | **Passed** |
| **TC-002** | **FR-05: Invalid Login** | 1. Enter an invalid username or password.  2. Click "Login." | An error message "Invalid login credentials" is displayed. | The specified error message was displayed. | **Passed** |
| **TC-003** | **FR-01: Add New Member** | 1. Navigate to Member List.  2. Click "Add Member."  3. Fill in all required fields and upload a picture.  4. Click "Save Changes." | The new member is saved to the database and appears in the Member List. | The new member was saved and appeared correctly in the grid. | **Passed** |
| **TC-004** | **FR-01: Update Member** | 1. Select a member from the list.  2. Click "Update."  3. Change a detail (e.g., address).  4. Click "Save Changes." | The member's information is updated in the database and the change is reflected in the list. | The information was updated and reflected correctly. | **Passed** |
| **TC-005** | **FR-01: Delete Member** | 1. Select a member from the list.  2. Click "Delete."  3. Click "Yes" on the confirmation prompt. | The member is removed from the database and no longer appears in the list. | The member was successfully removed from the list. | **Passed** |
| **TC-006** | **FR-03: Record Attendance** | 1. Navigate to Attendance.  2. Click "Attendance" button.  3. Enter a valid Member ID and select an event.  4. Set status and click "Save." | The new attendance record is saved and appears in the attendance list. | The attendance record was saved and appeared correctly. | **Passed** |
| **TC-007** | **FR-03: Duplicate Attendance Check** | 1. Attempt to record attendance for a member who already has a record for the same event. | The system displays a "Duplicate Record Found" error and does not save the new record. | The specified error message was displayed and no duplicate was saved. | **Passed** |
| **TC-008** | **FR-06: Search Functionality** | 1. In the Member List, type a part of a member's name in the search bar. | The grid filters and displays only the members that match the search query. | The grid filtered the results instantly and accurately. | **Passed** |
| **TC-009** | **NFR-01: Performance** | 1. Load the Member List form with 500+ dummy records.  2. Perform a search. | The form loads in under 3 seconds. The search results appear almost instantly. | Form load time was ~1.5 seconds. Search was instantaneous. | **Passed** |
| **TC-010** | **NFR-02: Usability** | 1. A test user was asked to add a new member without prior instruction. | The user can successfully navigate and complete the task with minimal to no assistance. | The test user completed the task successfully, commenting on the clarity of the buttons. | **Passed** |

**Table 2**. Test Cases and Results

**Test Summary:**

The testing phase was conducted successfully over a one-week period. All major functional and non-functional requirements were tested, and the results indicate that the system is stable and performs as expected. All test cases listed above passed without critical issues.

**Defect Report:**

During the development and testing cycle, several minor defects were identified and subsequently resolved. These included:

* **Defect #01 (Resolved):** A "GDI+ generic error" occurred when updating a member record without changing the profile picture.
  + **Resolution:** The image loading logic in LoadDataForUpdate was modified to create a new Bitmap object, preventing the file stream from locking.
* **Defect #02 (Resolved):** The data grid in the parent form did not automatically refresh after a record was updated.
  + **Resolution:** Implemented DialogResult.OK to signal a successful update from the child form to the parent form, triggering a grid refresh.

As of the final testing date, there are no known critical or major defects in the system. The application is deemed ready for deployment.

**MAINTENANCE GUIDE**

This section outlines the procedures and best practices for the ongoing maintenance and support of the Church Member Event Tracking System. Proper maintenance is essential for ensuring the system's long-term stability, security, and reliability.

**Procedures for Maintaining and Updating the Software**

Routine maintenance is critical to keep the system running smoothly. The primary maintenance tasks are to be performed by the church's designated system administrator.

* **Database Backup:** This is the most critical maintenance procedure. A full backup of the MemberInfo database should be performed on a regular basis (e.g., weekly or after every major data entry session). Backup files must be stored on a separate, secure physical device (ex., an external hard drive or a different computer) to protect against data loss from hardware failure, corruption, or ransomware.
* **System Monitoring:** The administrator should periodically monitor the computer's performance and available storage space. If the system begins to slow down, it may be an indication that the computer's resources are becoming limited.
* **Software Updates:** The system itself is standalone, but its underlying dependencies require updates. The administrator should ensure that the **Windows** Operating System and the .NET Framework are kept up-to-date by installing regular updates from Microsoft. This helps maintain system security and compatibility.

**Version Control and Release Management**

All changes to the application's source code are managed through a structured release process to ensure stability and traceability.

* **Version Control:** The source code for the project is managed in a version control system. This practice allows for a complete history of all code modifications to be stored. Every change is documented, which is crucial for collaborative development, tracking down the source of bugs, and safely reverting to a previous stable version if a new update introduces problems.
* **Release Management:** Each new version of the application released to the client is assigned a unique, incremental version number (ex., v1.0, v1.1, v1.2). A new version is released only after it has undergone thorough testing in a separate development environment. This process ensures that any updates or bug fixes deployed to the church are stable and will not disrupt their daily operations.

**Guidelines for Handling Bug Fixes and Enhancements**

Over time, users may discover minor bugs or request new features (enhancements). These are handled through a systematic process.

* **Reporting Bugs:** The administrator should collect detailed reports for any bugs encountered by users. A good bug report includes:
  1. A clear description of the problem.
  2. The exact steps to reproduce the bug.
  3. Any error messages that appeared (screenshots are highly recommended).
* **Requesting Enhancements:** All requests for new features or changes to existing ones should be formally documented. Each request will be evaluated based on its technical feasibility, development time, and overall benefit to the church's workflow.
* **Implementation and Deployment:**
  1. **Bug Fixes:** Critical bugs will be prioritized and fixed by the developer. A "patch" or a new minor version (ex., v1.0.1) will be released containing the fix.
  2. **Enhancements:** New features will be grouped together and implemented as part of a major or minor version update (ex., v1.1 or v2.0).
  3. All bug fixes and enhancements will be deployed during a scheduled maintenance window (ex., a time when the church office is not busy) to minimize disruption. The administrator will be responsible for installing the updated application files.

**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. |
| 2025-09-04 | 2.0 | -Added "Event Management" module to User Manual.  - Updated database schema with Events table.  - Revised System Overview to include new features. | Go, Justin Paul T. |

This section tracks the changes and updates made to this technical documentation over time. Each entry records the version number, the date of the revision, a summary of the changes made, and the author of those changes.

**Table 3**. Revision History