CHURCH MANAGEMENT SYSTEM

Stakeholder Requirements Specification

ABSTRACT

The church Management System was methodically designed to alleviate the inefficiencies surrounding the financial process in which a church goes through from the perspective of the collection of tithes and offerings to the generation of various expenditure reports as well as outreach goals progress reports.

TEAM MEMBERS

Cecil White
Andrew Titus
Matthew Hall
Peter-Jon Williams
William McDermott

COURSE

Advance Software Engineering, COMP6110

Contents

Introduction

Business Purpose

- To guide each individual to put his full faith and trust in the Lord.
- To grow in the experience of Christian prayer as it is defined and practiced in the Bible, to the glory of God.
- To minister to all persons regardless of race, national origin, or economic condition.
- To build the scriptural qualities of faith, hope and love in the lives of individual Christians and in the corporate life of the church.
- To have a service ministry to persons with special needs, more specifically, the sick, elderly, widows, orphans, the poor, the homeless, those away from home, and those in prison.

Business Scope

- Outreach Programme in the Communities
  - Free Garage Sale
  - Serving free lunches to community members
  - Back to School Drive
  - Donation of various supplies to Children and Nursing Homes

Business Overview

- To create environments where people are encouraged and equipped to pursue intimacy with God, community with insiders, and influence with outsiders.

Definitions

- EPS – Electronic Payment System
- CFS – Church Financial System
- CMS – Church Management System
- Electronic Donation/Payment/Transaction – Any financial transaction electronically transmitted.

Stakeholders

- Church Leader - The most influential member of the church who everyone reports to.
- Group Leader – A senior member of the church who has been chosen to lead a small subsection of the congregation. They may plan and execute various activities/events for the benefit/sake of the church.
- Office Administrator – A member of the church that carries out the church's routine tasks to ensure the smooth daily operation of the church.
- Auditor – A member or external professional who is independent of the regular management of the churches daily finances.
- System Administrator - Information Technology competent member of the church assigned to manage the system's IT components.
- Member – A registered member of the church.

Business Management Requirements

Business Processes

- Payment of bills, wages etc.
- Collection of donations
- Planning and execution of events
  - Outreach
  - Celebratory

Business Environment

- Non-Competitive

Goal and Objective

- Used tithes and offerings collected to:
  - Carry out various outreach projects and ministry the church have planned.
  - Help church member financially with difficulties they may be experiencing.
  - Upgrade/Enhance the church equipment and facilities.

Business Model

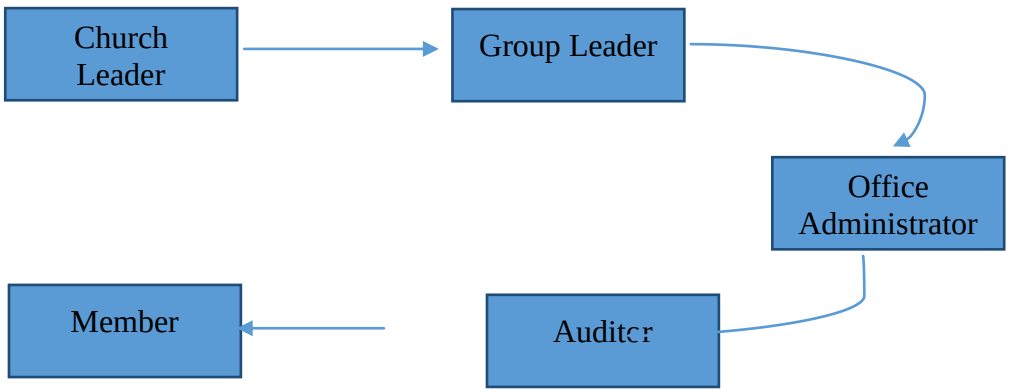

Illustration above depicts the Business Model of a church with the Church Leader being at the top of the hierarchy and a regular member of the church being the lowest.

Information Environment

The process in which organizational level decisions (that may/will affect the entire church) are decided upon ultimately by the church leader along with the input from group leaders of the church (direct subordinates).

- This is the only project planned to address the management of the church financial affairs and as such no constraints will be made on the current project portfolio.
- The Church Financial System (CFS), however, is a component of a long term project plan to develop a Church Management System (CMS) and subsequently be a constraint on that system.

Problem Statement

The ever increasing pervasiveness of technology in our lives, has allowed people to reduce their dependence on cash transactions. As such many persons today are less likely to be carrying around large sums of money. This presents a problem, as church members are limited by the options that are given to them to make large donations such as tithes. There is also an issue with record keeping as many local churches are currently manually recording financial records. This has the potential to introduce errors in the financial records while also making it difficult to quickly generate accurate reports on member donations and church expenditure.

In order to close the gap, churches can implement an electronic financial management system that accepts payments from members and keeps records of various transactions. This would promote financial prudence.

Implementing a financial management system, will help churches meet their financial requirements. Members will be able to donate larger sums of money without the church having to worry about securing the cash between the time it's donated and deposited at the bank. It also provides timely and accurate reports on financial status of the church to its leaders and members alike.

User Requirements

Each actor inherits all roles of the member category. The member Actor is therefore the “lowest” level user in the system/problem domain.

Only members shall be able to:

1. Donate money electronically.
2. View their transaction history.
3. View progress to reach event/cause goal. (summary report)
4. Add credit card payment information.
5. Delete credit card payment information.
6. View credit card payment information.
7. Update personal profile information.
8. Add profile information.
9. Delete profile information.
10. View profile information.
11. Update profile information.
12. View own transaction history
13. View summary of income vs. expenditure report

Only Office Administrators shall be able to:

14. Enter cash offering/transactions into the system.
15. Pay utility bills from the system.
16. View bill payment history.
17. Enter an invoice.
18. View invoices.
19. Pay-out approved invoices.
20. View various financial reports.
21. Make bill payments directly to bank via system interface.

Only System Administrators shall be able to:

22. Set-up groups/departments to be headed by leaders.
23. Add a user,
24. Delete a user.
25. View all users.
26. Update a user.
27. Add a group,
28. Delete a group.
29. View all groups.
30. Update a group.
31. Add Electronic Payment System (EPS) gateways.
32. Delete EPS gateways.
33. View all EPS gateways configured.
34. Update gateways to different banking systems.
35. View various system reports (error logs, user management etc.).

Only Auditors shall be able to:

36. Perform audits on all financial transactions.
37. Make requests for financial adjustments.
38. View various financial reports.
39. View audit report/general system event report

Group Leader shall be able to:

40. Add new events/causes for members to donate to, approved by Church Leader.
41. Discontinue events/causes they created, with church leader approval.
42. Update events/causes they created, with church leader approval.
43. Create unapproved budgets.
44. Delete unapproved budgets created by them.
45. View all budgets created by them.
46. Update unapproved budgets created by them.
47. Duplicate any budget created by them.
48. Create unapproved budget items.
49. Delete unapproved budget items created by them.
50. View all budget items created by them.
51. Update unapproved budget items created by them.
52. Group Leaders shall be able to enter invoices into the system.
53. Make a request for additional funding for out of budget expenditures, with Church Leader approval.

Church Leader shall be able to:

54. Reallocate funds from one Church Groups active budget to another.
55. Approve requests for financial data modification.
56. Reject requests for financial data modification.
57. Approve requests for additional funding.
58. Reject requests for additional funding.
59. Create unapproved budgets.
60. Delete unapproved budgets created by them.
61. View all budgets created by anyone.
62. Update all unapproved budget items created by anyone.
63. Duplicate any budget.
64. Cancel any active budget.
65. View various reports. (ie. transaction, progressions, etc…)

Only Church Leader and Group Leaders shall be able to:

66. View tithes report.
67. View events report.
68. View building/yard maintenance report.
69. View utilities (water, electricity, telephone and internet) report.
70. View salaries report.
71. View generous givers report.

Report Requirement Breakdown Summary for each actor

1. Member shall only be able to view their transaction history.
2. Auditor shall only be able to view audit reports
3. System Administrator shall only be able to view system reports.
4. Group Leader shall only be able to view general reports
5. Church Leader shall be able to view all reports excluding system reports.
6. All actors are members, hence can view member reports.

CFS Class Diagram

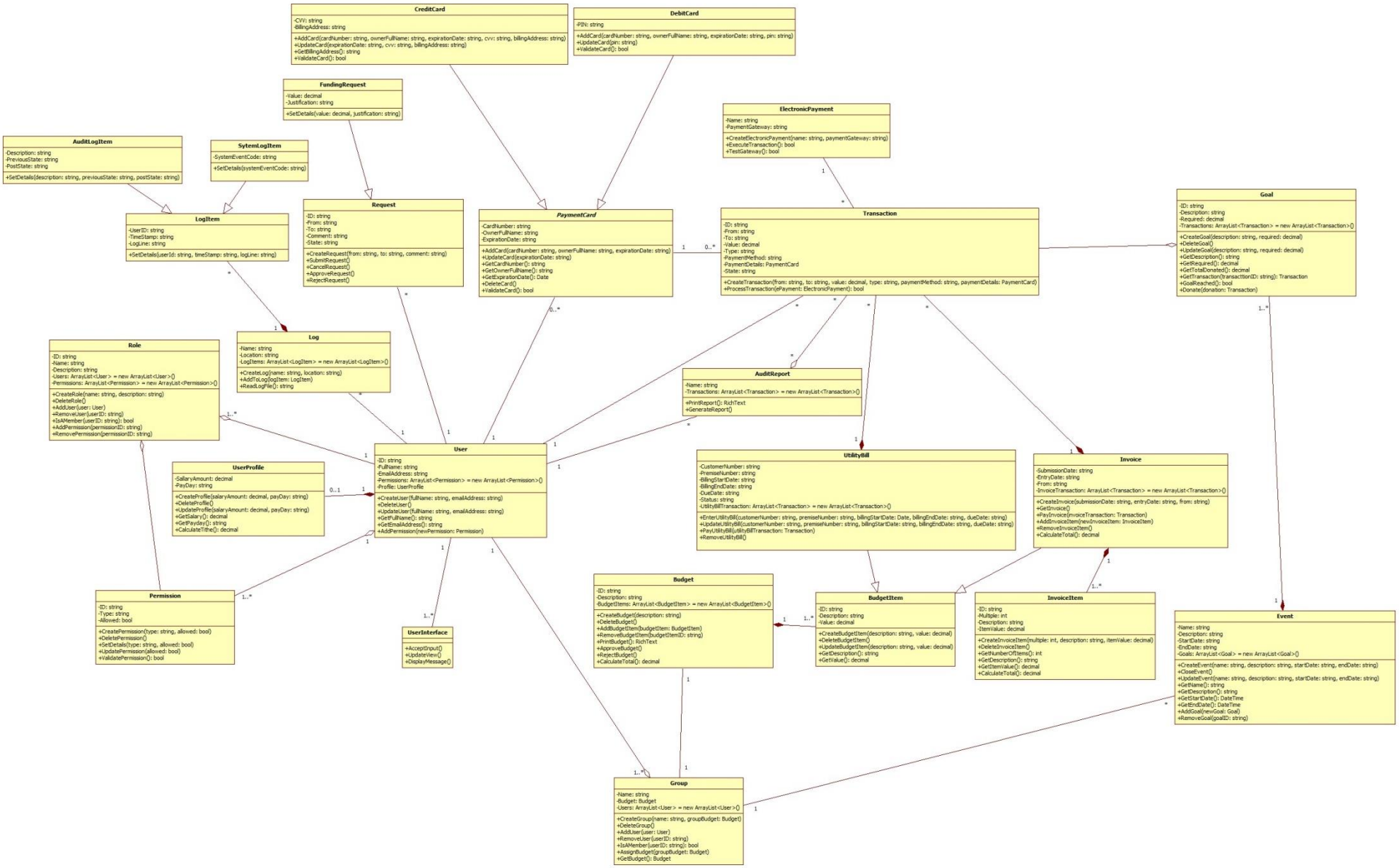

Figure 1: Illustrating an overview of the CFS relational class diagrams

CFS - OCL Diagrams

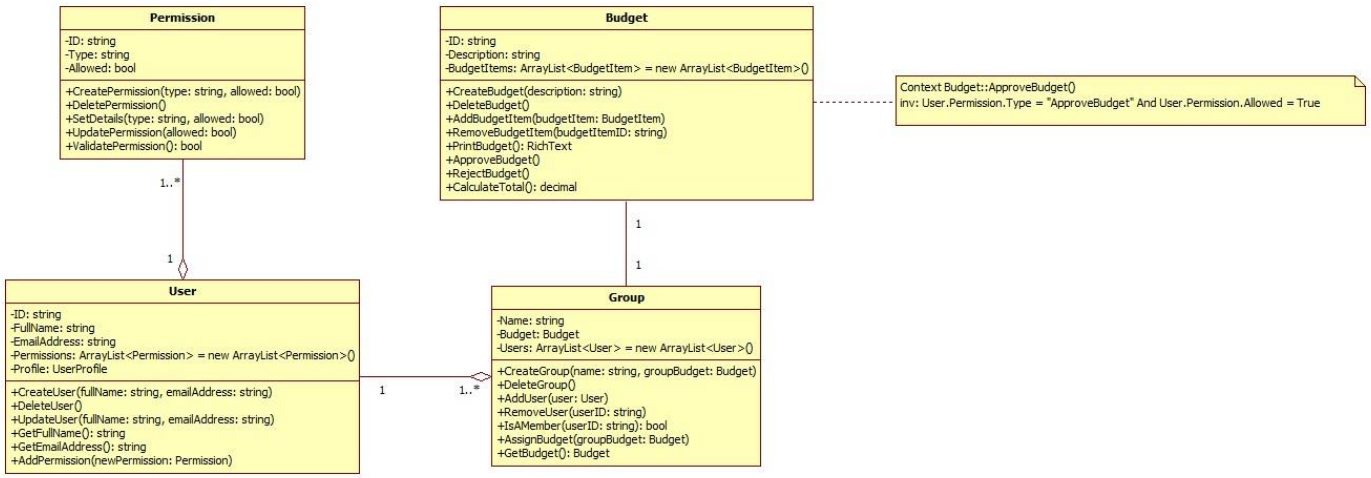

Figure 2: Illustrating the CFS Budget OCL diagram

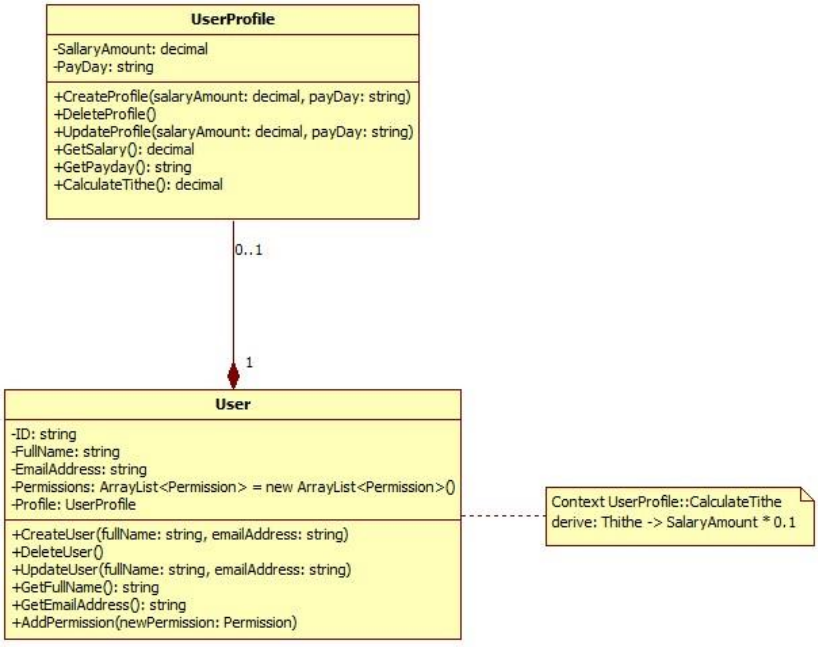

Figure 3: Illustrating the CFS Tithe Calculation OCL diagram

Use Case Diagrams

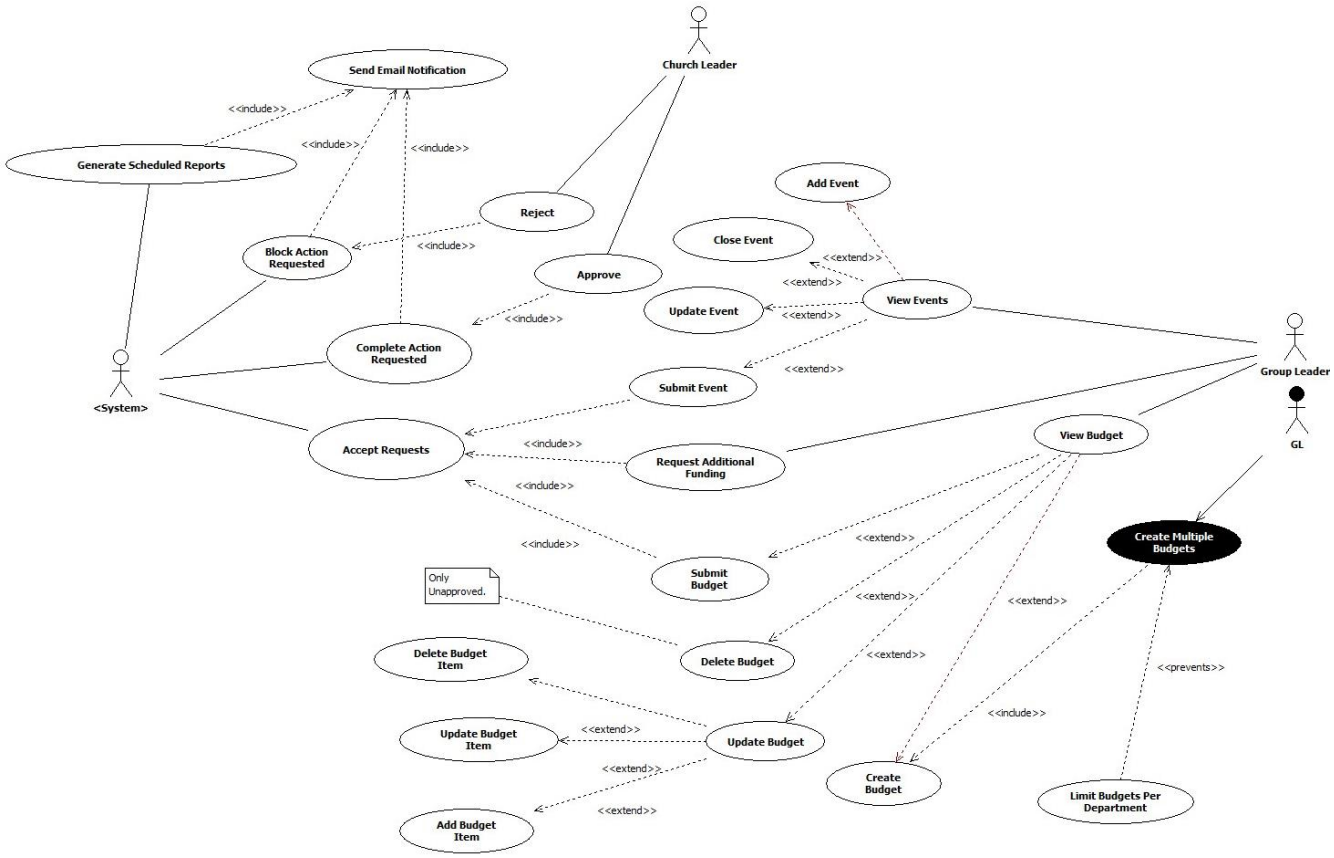

Figure 4: Illustrating the CFS Church Leadership Use case diagram

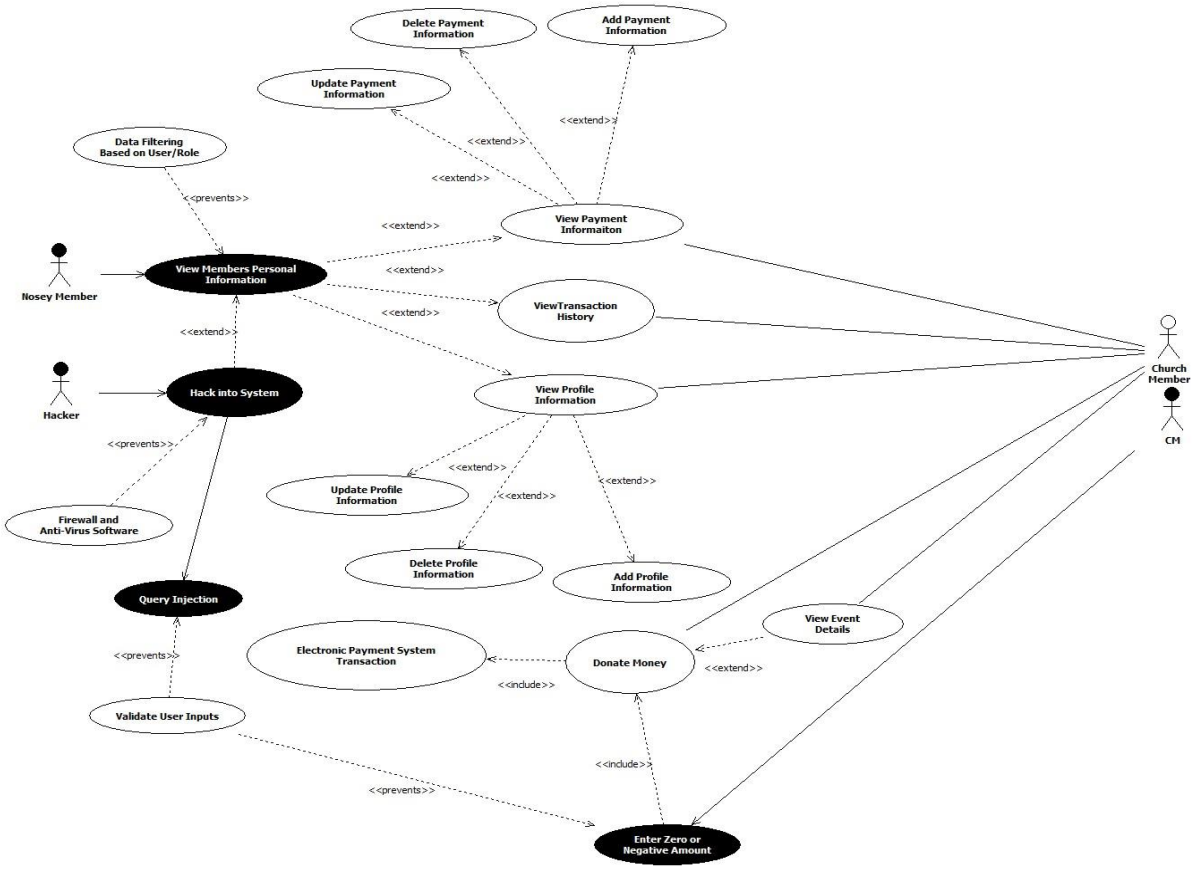

Figure 5: Illustrating the CFS Member Use case diagram

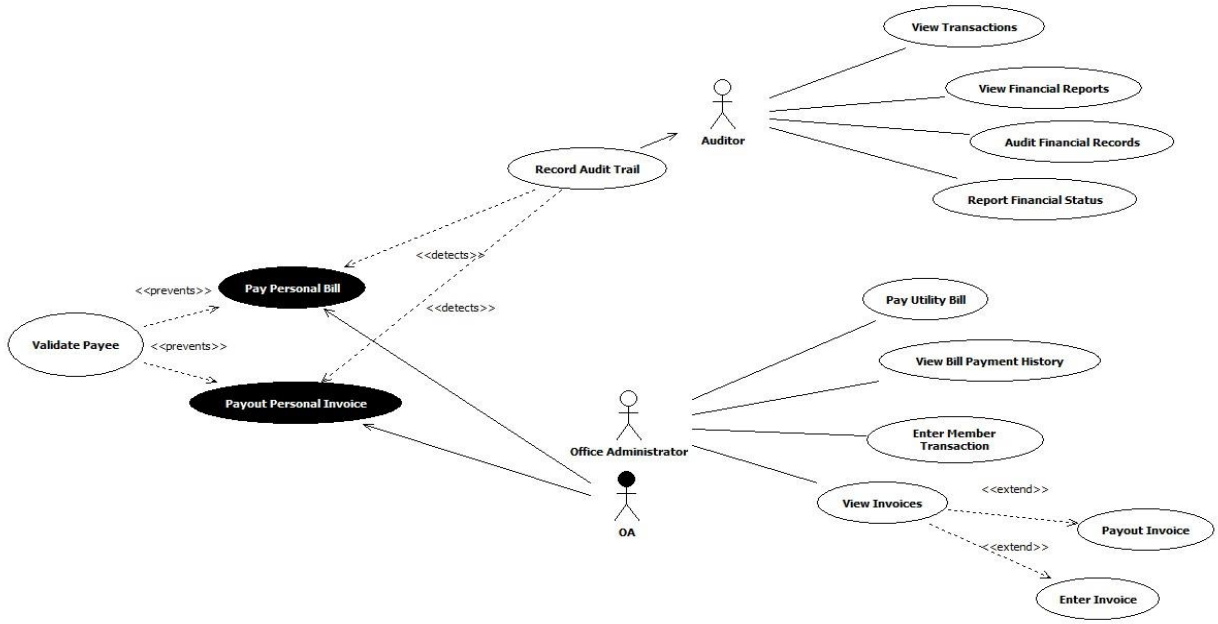

Figure 6: Illustrating the CFS Office Administrator Use case diagram

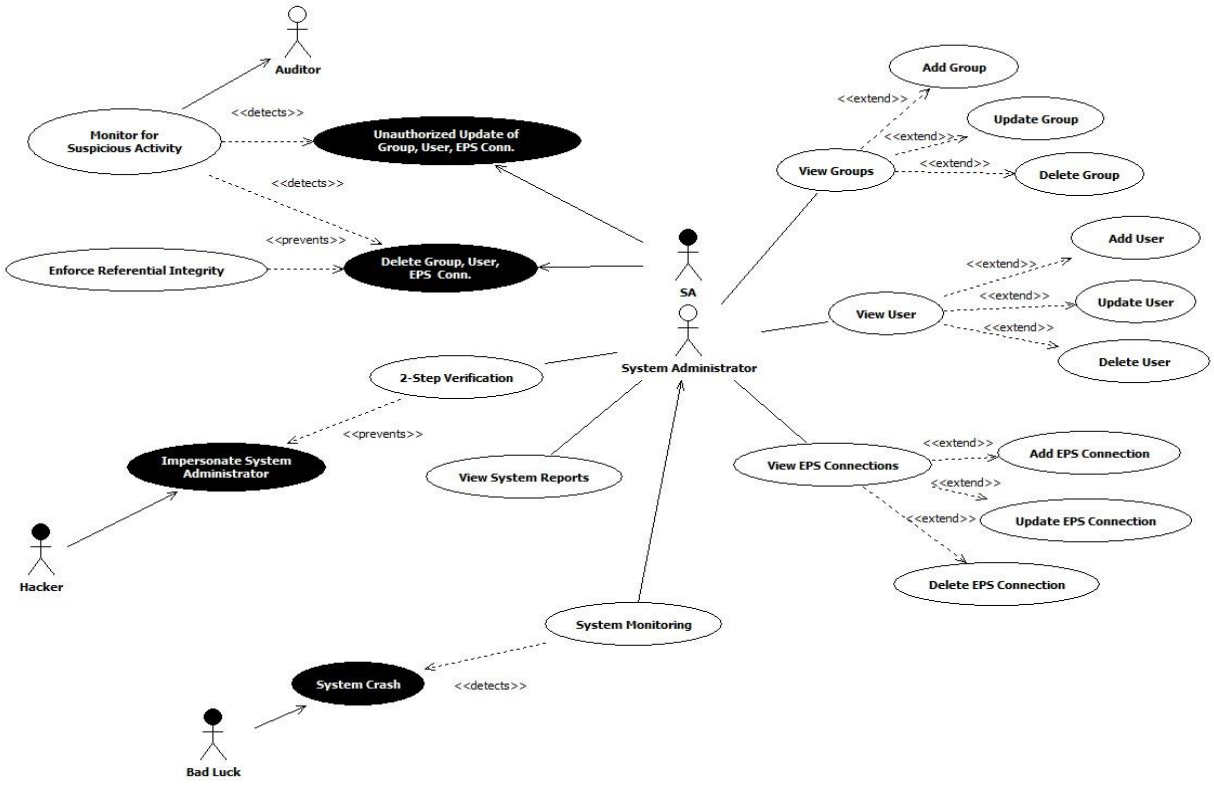

Figure 7: Illustrating the CFS System Administrator Use case diagram

Use Case Descriptions

The system contains the following actors:

- **Church Member**: A member is a person who is registered with the church as being a member of the general congregation.
- **Group Leader**: A group leader is a member of the church whose has been designated the responsibility over the management of a department of the church. Group leaders have the same use cases as church members in addition to their own.
- **Church Leader**: A church leader is a senior group leader that is the head of the church and has oversight over all the group leaders. A church leader has the same use cases as church members and group leaders; in addition to his. Church leaders make the final decisions on the most important church matters.
- **Office Administrator**: An external professional administrator or a church member who is familiar with administration duties. The responsibility of the administrator is to interface with the system on behalf of persons who don't have direct access to the system. These can include church members, contractors etc. Office administrators also dispatch the duties, such as paying bills, to keep the church operations running smoothly.
- **Auditor**: An external professional auditor or a church member who is familiar with the auditing process. This person is responsible for auditing all the transaction that pass through the system on a predetermined schedule.
- **System Administrator**: The system administrator is responsible for ensuring that the financial system runs with minimal downtime. They are also responsible for user management.

The system contains the following mis-actors:

- **Nosey Member**: A church member who deliberately try to view the personal information of another member without proper access rights or permissions.
- **Hacker**: A person either internal or external to the system that attempts or succeeds in circumventing the features of the system put in place to uphold data integrity and security, for malicious reasons.
- **CM**: A church member who unknowingly/knowingly uses the system in a way it was not intended to be used.
- **GL**: A group leader who unknowingly/knowingly uses the system in a way it was not intended to be used.
- **OA**: An office administrator who unknowingly/knowingly uses the system in a way it was not intended to be used.
- **SA**: A system administrator who unknowingly/knowingly uses the system in a way it was not intended to be used.

- **Bad Luck**: A security threat that arises from unexpected equipment failure, sudden operator illness and other negative factors.

The main use cases are as follows:

- **View Event Details**: Events created by group leaders can be viewed by church members. Members will be able to view details such as Goal, Percentage of Total Donation of Goal etc. For group and church leaders, they can then decide what modification to this information is required.
- **Donate Money**: Allows church members to use electronic payment systems to donate money to the church.
- **View Payment Information**: Allows church members to view all forms of payment they have configured. Examples of payment information include credit card and debit card information. The church member can then decide what modification to this information is required.
- **View Profile Information**: Allows church members to view their personal profile information they have entered. Examples of profile information are salary information and personal donation goals. This information is used to aid in tithe calculations and reminders of when to donate. The church member can then decide what modification to this information is required.
- **View Transaction History**: Allows church members to view all the transaction activity linked to their user account. Church members can use this to audit their own donations to the church.
- **Validate User Input**: Ensures that the strings of user input that are to be used as parameters in queries don't contain sub-queries or malicious data that can modify the intent of the query. This also ensures that input for donations are not (input <= 0) less than or equal to zero.
- **Firewall and Anti-Virus Software**: Prevents unauthorized access to the system from internal or external network or software vulnerabilities.
- **Data Filtering Based on User/Role**: The system will at all times present the user with the information they are allowed to access.
- **View Budget**: Displays the selected budget details to the user.
- **Submit Budget**: Marks the proposed budget as completed and notifies the church leader of a groups proposed budget is waiting on his approval.
- **Submit Event**: Submits the proposed event with goal to the church leader for approval.
- **Request Additional Funding**: A group leader can request additional funding from the church leader for funding small important tasks not covered under the active budget.
- **Approve**: A use case that only the church leader can execute in order to approve various requests.
- **Reject**: A use case that only the church leader can execute in order to deny various requests.

- **Accept Request**: The CFS system will hold all requests in a repository until the church leader can attend to them.
- **Complete Action Requested**: This is based on the action the church leader wishes to approve. The CFS system will call the required method of the object to execute the action.
- **Block Action Requested**: This is based on the action the church leader wishes to deny. The CFS system will call the required method of the object to execute the action.
- **Generate Scheduled Reports**: The user will be able to configure reports to be generated on a schedule and sent to different actors of the system.
- **Pay Utility Bill**: Allows the office admin to pay utility bills associated with the church.
- **View Bill Payment History**: Allow the office admin to view a history of all previous bills paid.
- **Enter Member Transaction**: This allows the office admin to enter a transaction that was not made by electrical means (i.e. Credit card, debit card)
- **View Invoices**: Allows the user to view the invoices entered by the office admin for work done for the church.
- **Record Audit Trail**: The system will log all events done by a user.
- **Validate Payee**: Prevents office administrators from either paying the incorrect bill or a bill that does not belong to the church.
- **View Groups**: View all groups.
- **View User**: View all users.
- **View EPS Gateway**: Give the ability to the system admin to view all e-payment gateways configured on the system.
- **View System Reports**: Gives the ability to users to view the system reports.
- **2-step Verification**: Prevents unauthorized access to a system admin's account by a hacker or malicious user.
- **System Monitoring**: Detects any system fault and reports it to the system admin.
- **Enforce Referential Integrity**: Ensures data consistency after the modification of one object.
- **Monitor for Suspicious Activity**: Detects any unusual activity base on previous use and reports it to the auditor.

Sequence Diagrams

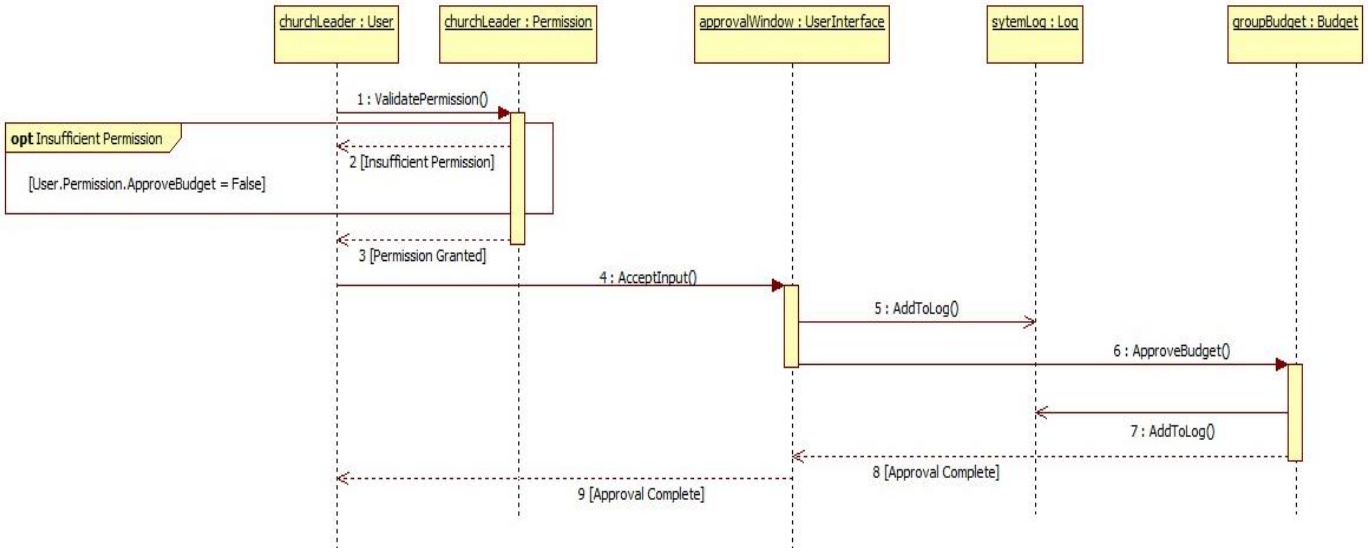

Figure 8: Illustrating the CFS Budget approval by Church Leader sequence diagram

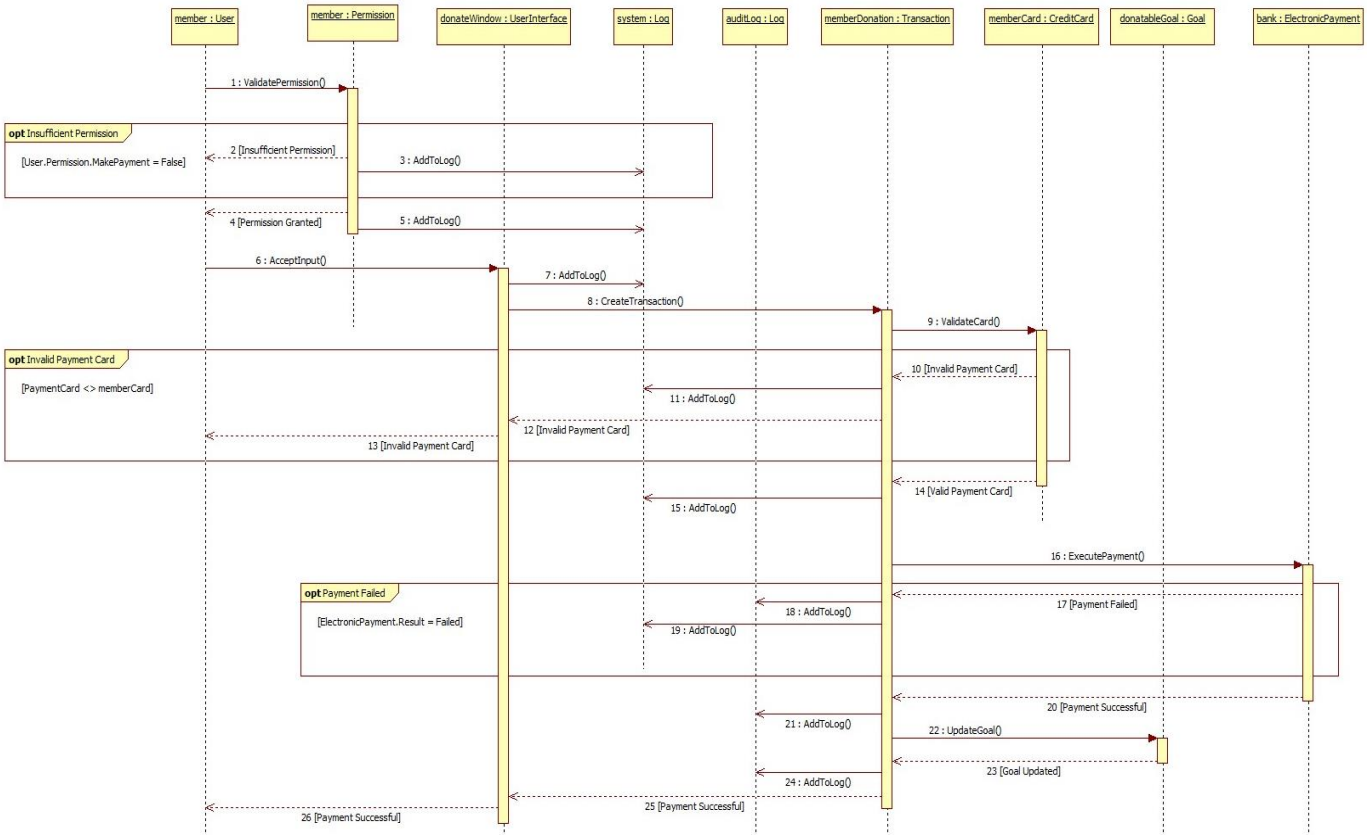

Figure 9: Illustrating the CFS member donation sequence diagram

System Screen Shots

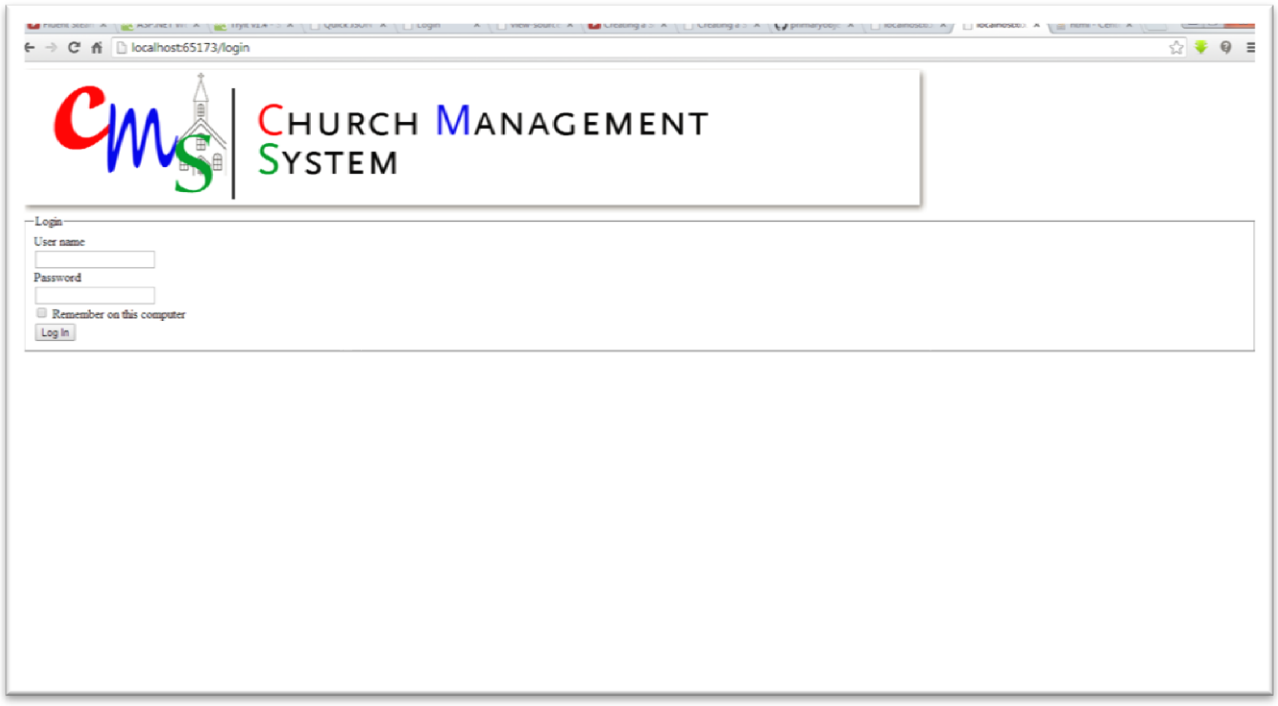

Figure 10: Illustrating CFS Login Screen

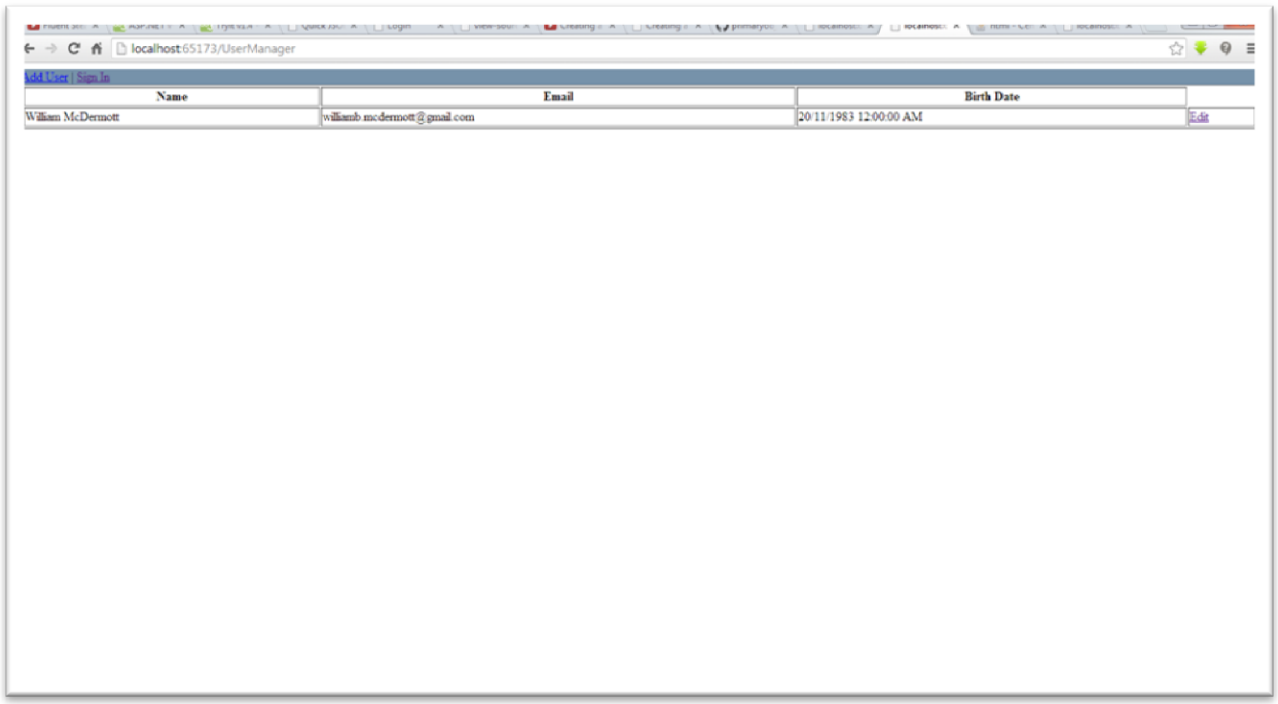

Figure 11: Illustrating CFS View, Add and Updating a User

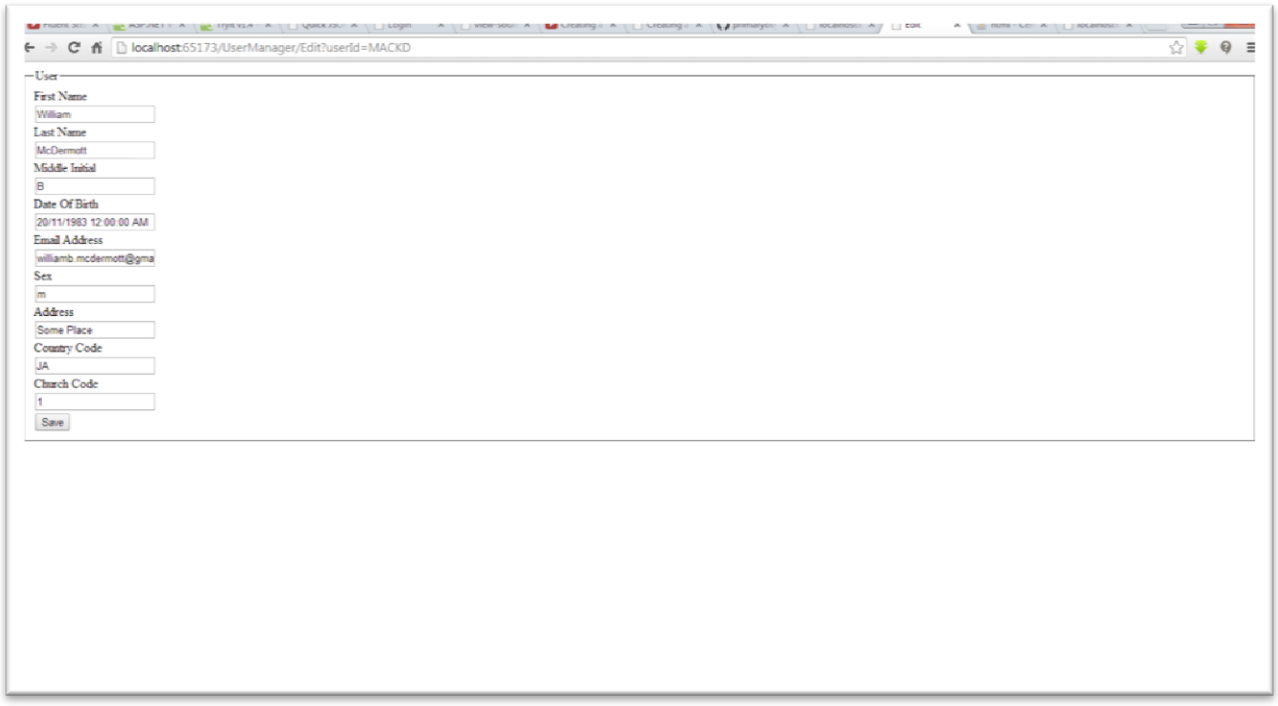

Figure 12: Illustrating Entering User Information

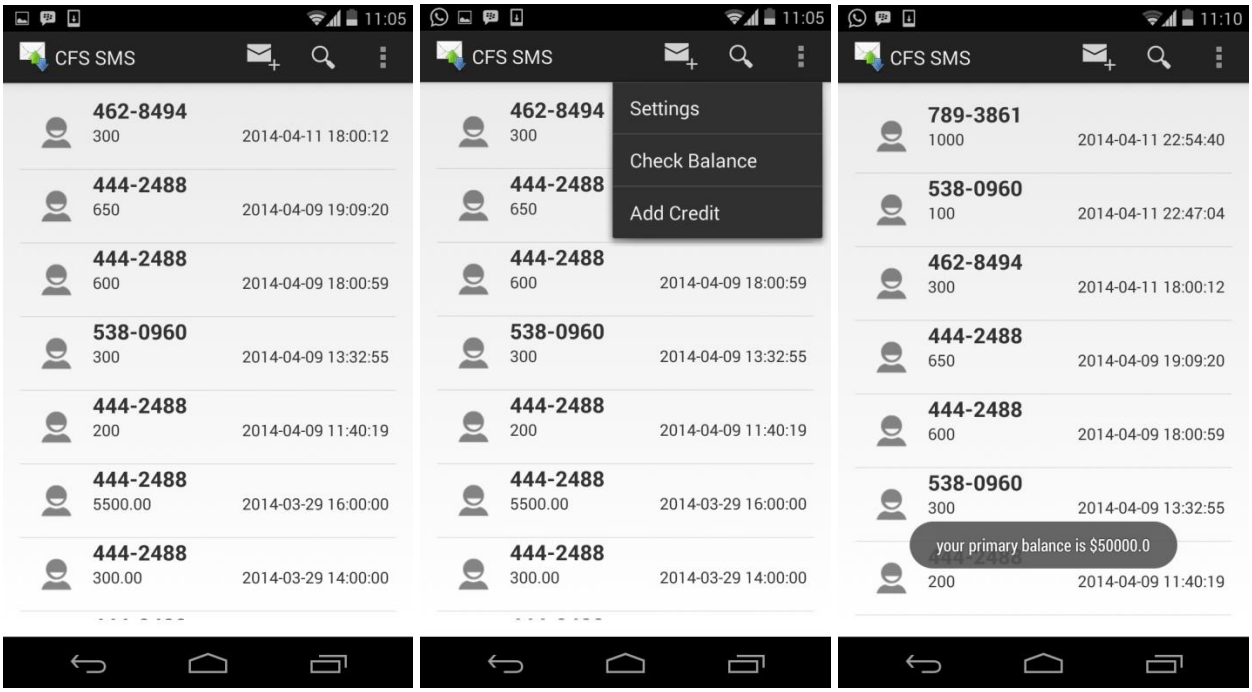

Figure 13: Illustrating CFS SMS Component Used to Donate and Check Balance

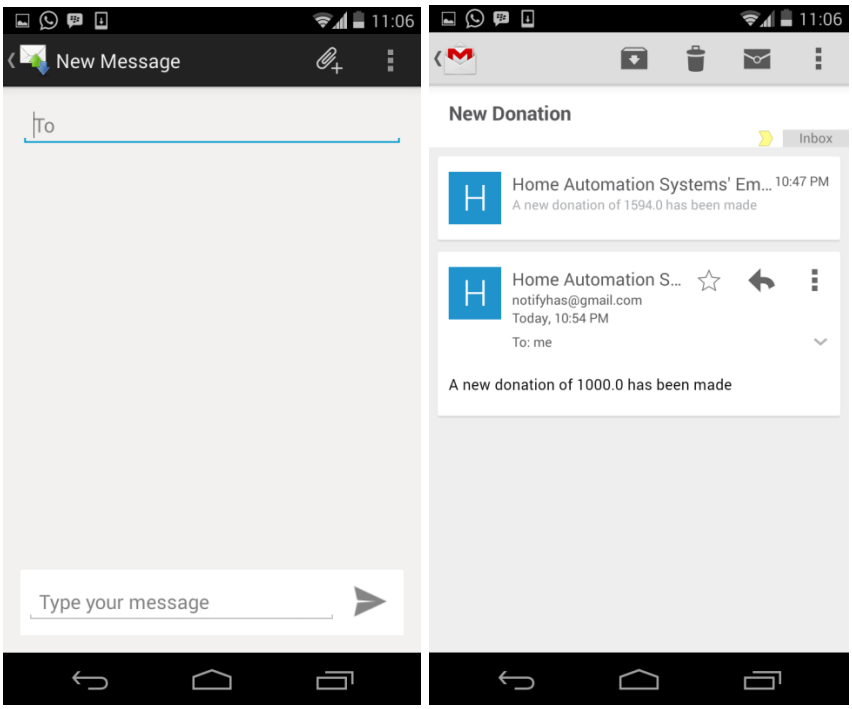

Figure 14: Illustrating Creating a New Donation Message and Receiving Confirmation Email Using Notification Component.

```
public boolean mailtoChurchLog(String receiver, String subject, String body)
{
    HttpClient httpClient = new DefaultHttpClient();
    HttpPost httpPost = new HttpPost(
            "http://ticketmanager.mysoftware.io:8100/component-email/src/send_email.php");
    httpPost.addHeader("Content-type",
            "application/x-www-form-urlencoded");
    BasicNameValuePair destination = new BasicNameValuePair(
            "To", receiver);
    BasicNameValuePair topic = new BasicNameValuePair(
            "Subject", subject);
    BasicNameValuePair content = new BasicNameValuePair(
            "Body", body);


    List<NameValuePair> nameValuePairList = new ArrayList<NameValuePair>();
    nameValuePairList.add(destination);
    nameValuePairList.add(topic);
    nameValuePairList.add(content);
```

Figure 15: Calling the Email Notification Component from CFS SMS Component

Architecture

Both components were developed using Model View Controller architecture. More specifically MVC4 (User Management Component) and MVC (SMS Component)

CMS Component – Database Management Services

Description

The purpose of this component is to provide a client with database management services such as CRUD operations with database tables, and Insert, Read, Update and Delete queries for database elements.

Arguments

- **server**: This is the string representation of your database server IP address or name
- **database**: This is a string representation of database name
- **user ID**: This is a string representation of your database user name
- **password**: This is a string representation of your database password
- **dbtype**: This string representation of database type this currently supports the following server types:
    - mssql: For Microsoft SQL server
    - mysql: For mySQL servers
- **query**: This is a string representation of your SQL query

Access End Point:

holycrosschurchjm.com/dbcomponent.php

Access Method:

To use the web service, simply POST each argument to the access end point. That's it.

Constraints

- This component assumes that your database is publicly available. If your database is stored locally, please clone this project, and add the php file to your local server. The access methods will remain the same. An alternative to this may be to pass your IP address as the server parameter; however this has not been tested by our group.

- Also, the remote endpoint provided works for MySQL users **ONLY**. For those using SQL Server, you MUST use the file locally. This was not our intention, but the remote server does not have the necessary extensions to support SQL Server connections.

Output

- **Code**
  - 200: success
  - 422: sql exceptions/ other exceptions
- **Data**
  - A JSON object with values retrieved from the database if a data is select query is performed or a status, delete or update with a description what happened.
- **Debug**
  - data: status if process failed
  - message: this gives exemption which may occurs from the database or otherwise

{"code" : 200/422, "data" : [JSON Object , JSON Object], "debug" : {"data" : "message"}}

Example Queries to be executed

Select id from logins;

Insert into logins values (‘nw-0001’,’Password1’) ; // this example can also work for other query types that are not returning specific data from the database (i.e. insert, update, delete, create, alter drop)

Example Output

Select

{"code" : 200, "data" : [{"id" : "aj3013"}, {“id” : ”mh1234”}], "debug" : {"data" : "No Error Occurred"}}

Insert

Success :

{"code" : 200, "data" : “Successful”, "debug" : {"data" : "","message" : "No Error Occurred"}}

Unsuccessful :

{"code" : 422, "data" : “Unsuccessful”, "debug" : {"data" : "","message" : " Error Occurred"}}

Appendix

Acronyms and Abbreviations

- GL – Group Leader
- CL – Church Leader
- OA – Office Administrator
- CM – Church Member
- SA – System Administrator
- EPS – Electronic Payment System
- CFS – Church Financial System
- CMS – Church Management System
- Electronic Donation/Payment/Transaction – Any form in which a transaction in the system is done that didn't require actual cash to change hands.

Table of Figures