Software Design Specification

for

JointheC

1.0

Prepared by

La Shawn Williams	**620106685**	**lashawnwilliams98@outlook.com**
Chantay Whyte	**620110100**	**chantaywhyte97@gmail.com**
Jusaiah Smith	**620099392**	**jusaiah4g@gmail.com**
Jevoy Charvis	**620084694**	
Marc Grant	**620096801**	**marcgrant10@gmail.com**
Zoann Lyn	**620085748**	**zoejayelyn@gmail.com**

Course Instructor:	***Richard Anderson***
Course:	***COMP2140 – Introduction to Software Engineering***
Studio Facilitator:	**Phillipa Bennet**
Date:	***November 29, 2018***

TABLE OF CONTENTS

1.0 Project Overview

The client is the Children's Ministry of the Portmore New Testament Church of God which has a congregation of approximately 45- 70 children per Sunday. They previously had a system that was used to keep track of children that attends children's church as well as ministries activities. Consequently, that system crashed which led them to use a manual system which they are currently seeking to change. The ministry has weekly service and two annual events. Ministry workers are often volunteers with the responsibility of overseeing the children. The number of workers fluctuate for each event and each service, the same is with the number of children present. The approximate number of workers is 20,the total number of children per event is 150 and children per Sunday is 60.

The director of the Children's Ministry, Mellecia Smith, has realized there is a greater importance and need in keeping in contact with the children and their parents. She has a strong desire to implement a new system replacing the crashed one.

The aim of the system is to keep a centralised record of all events and data. The system shall provide the users with the latest updated contact information for each child and guardian (i.e any caretaker of the child) to be accessed by the respective worker(s). The system, JointheC, is intended to be used by youth directors, workers that are assigned to at least one child, and the administrator (acting also in the faculty of a receptionist and treasurer). The system should be developed to allow easy transition onto an intranet, allowing different ministries to have access to the same service, information and centrally updating such.

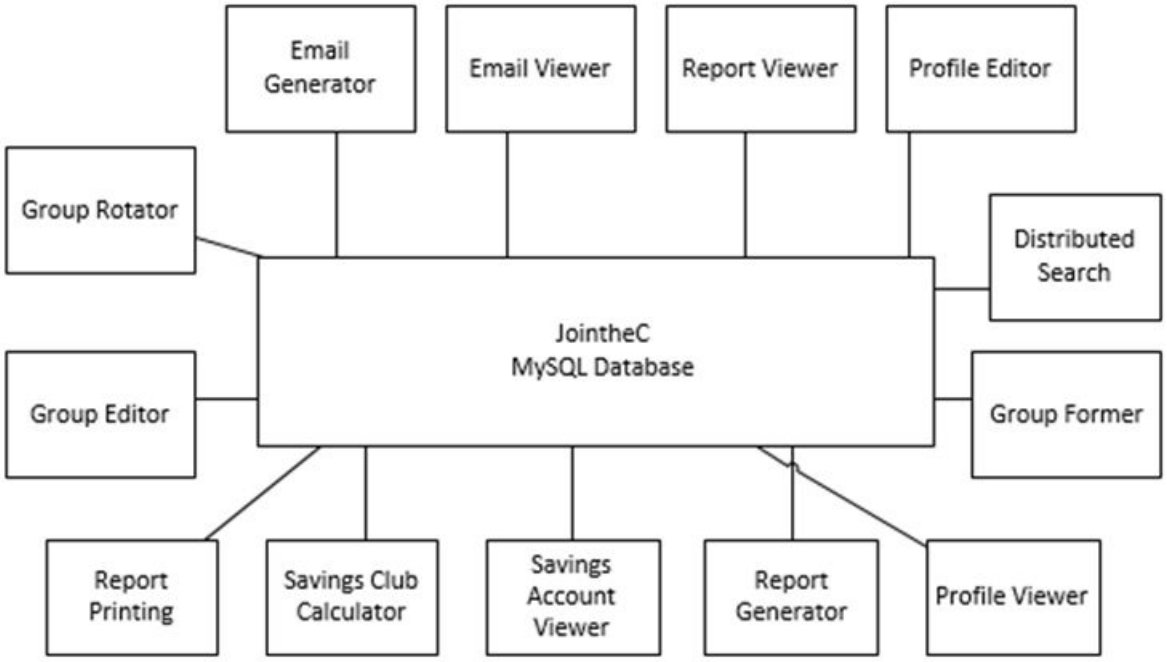
2.0 Architectural Design

The repository pattern was used to develop JoinTheC. The purpose of the repository pattern is to provide an abstraction layer for data access. With this design, all system components; email, reports, profiles, groupings, and savings club, share a database without directly communicating with other components. Sub-systems have their own databases which may still result in multiple databases being used for the entire system. However, the latter is not applicable to JointheC. Classes/components do not interact directly but are able to exchange data through the shared database.

2.1 General Constraints

The global limitation significantly impacting the system design of JointheC is memory recovery. There is no set size for the system as it expands depending on the number of workers and children added over the years. However, for effectiveness, a PC with at least 4GB RAM will be necessary. JoinTheC can only be operated on a desktops and laptops as it is made to run on the most common OS, Windows 8 or above. The system does not need an internet connection to be fully functional. However, internet connection will be necessary for a user to send an email.

2.2 System Architecture Diagram



N.B. Bidirectional flow between components and the database

2.3 Alternatives Considered

The Layered Architecture pattern, the Repository Architectural Pattern, and the Client-Server Architectural Pattern were all considered for JoinTheC.

The Client-Server Pattern was not applicable as it is best suited for a network dependent system. JoinTheC only utilises a network as an alternative means of contacting parents (email) and storing its data (data backup).

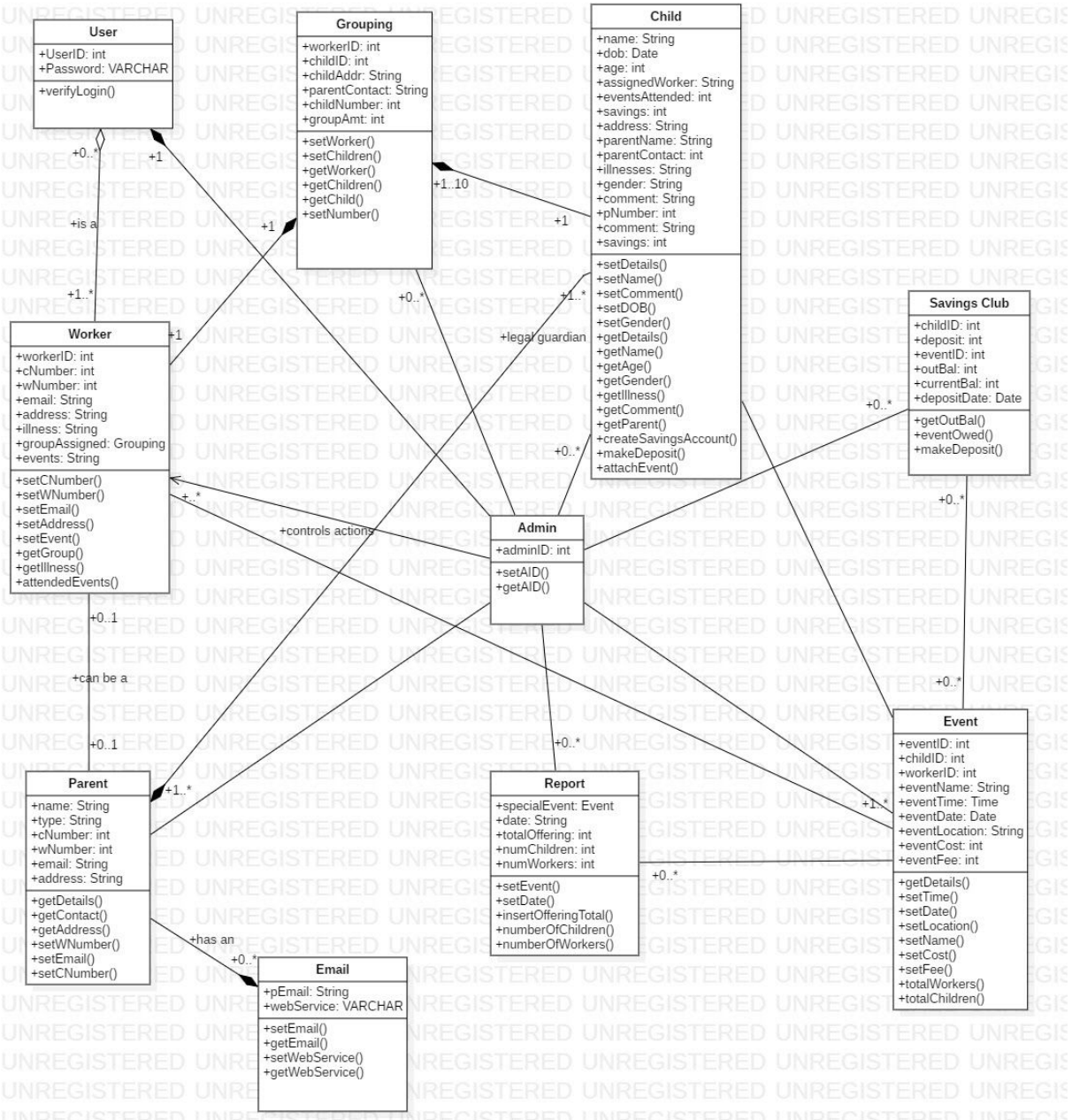
The other alternative considered was Layered Architecture pattern which organizes the system into layers with functionality associated with each layer with the lower layer providing services to the layer above it. This form of structure requires an existing system to operate off of but JointheC is in creation and will not include aspects of another system. Therefore the layered pattern cannot be applicable as it is suitable for a system being built on another existing system.
Additionally, components of the system all have to communicate with the database used by JointheC. By placing these components in layers, the system will be made more complex; thus, making the system less user friendly and increase difficulties in development.
Some requirements stated by our stakeholders are recording information, making queries, grouping students and reporting. In deciding on how a user will interact with

the system, based on the layered pattern, we noticed that in order to make a query the information would first have to be recorded, indicating that the recording of information would be the lower layer providing service to the upper layer (querying/search). This can be problematic in deciding what a layer may contain given that one requirement services the other but also it would be more suitable to have both in the same layer. However, the biggest disadvantage of the layered architecture pattern is that high level layer would have to interact directly and only with the layer below it. For example, our report layer would have to draw information from different components/layers but is restricted by having to wait on one layer to service other layers before its data can be received. This increases time spent as each layer interprets the request differently and consumes time in the multiple processes.
LTE chose the Repository pattern rather than the Layered pattern as it uses one central database through which each component communicates and needs no knowledge of the other to operate. It is also less time consuming compared to the Layered architecture. It is easier to add a component to a system using the Repository pattern, this allows a greater possibility of expanding the system in the future.

2.4 Architecture Justification

The Repository Model offers consistent modification of data through centralisation, while allowing the system to be developed for multiple other end systems (PCs) to be connected to the database through the system. The users of the system will require the most recent data at all points and at any given time, therefore, allowing immediate update of data which is highly beneficial to the users.

3.0 Class Diagram / Structure Chart



3.1 Design Notes

Notes on each class

Admin: The admin class is used to give administrators of the Children's Church full access to the system to view, edit, create profiles, and make changes wherever and whenever necessary. Therefore, an admin has an association with all other classes in the system.

User: This class is responsible for controlling the type of access being granted to users of the system. Access to the system is always had by an administrator and this person (a type of worker, and system developer) is responsible for adding workers, children, groups, and events to the system. As a result, users of the system exist as workers but a user cannot exist without an admin.

Worker: The worker class creates a new worker in the system, assigns them to a group of children and take their details (such as name and contact information). The class also takes note of any illnesses a worker may have. There is an association between the worker and parent class as they are not mutually exclusive; a parent can also be a worker and a worker a parent but this is not always the case.

Parent: A parent associated with a child must be in the system for a child to be added. This is to ensure that an emergency contact for the child will always be in the available. Though the word 'parent' is used it must be kept in mind that any legal guardian is included in this category. Multiple contact information is taken as the parent is the emergency contact for the child.

Child: This is dependent on a parent for legal and security purposes. A parent must give permission for a child to be included in the system. By giving contact information, the permission is granted.

Savings Club: A new object of a Savings Club is generated when a child is added to the system. This class provides methods for calculating the current balance after each deposit and after an event fee is applied to a child's account. This class also allows the system to keep record of outstanding balances for an event and the date each deposit is made on. When a child has been given permission to attend an event, the cost of that event is brought into the savings account, hence the association between the Savings Club and Event classes.

Report: The report class generates reports on the activities of the church for the month. It is dependent on the Event class as it will take information to generate a summary of the events held. It allows the admin to input the number of workers for the month and each Children Church Sunday service, number of attendants per services for the month, as well as the total offerings for the month.

Grouping: This class is responsible for grouping children and assigning each group to a different worker each month. It is dependent on the Child class as well as the worker class as the information from these classes are used to assign the different groups.

Event: The event class allows an object to be created representing a new event at any point in time. The event is stored within the database and it's details can be drawn upon for a report when necessary.

Discussion of Assumptions

Children, parents, workers, and events (within each category) may share particular details which may be used to search such as name, gender, and age. Because of the high probability of this, these classes will assign IDs to each new object.