**Project Abstract:**

Design a Java GUI to allow entry and retrieval of weekly contributions to a local church. The program should allow the user to print a report listing each donor and the amount and ministry area for the current week and a summary page with total weekly deposit information. The purpose of this program is to allow the counters to communicate efficiently with the treasurer of the church while only allowing them to access the current week’s donor information (a privacy issue).

**Project Team Members:**

Kevin Lancaster

Jedidiah Bird

Dondra Richards

Brian Hinkle

Timia Davis

**Project Technologies:**

This project will use Java and JavaFX/Swing technologies. Although Swing allows easier GUI design, JavaFX allows greater control over some specifics of GUI elements.

**Project Responsibilities:**

Kevin Lancaster:

Overall team lead

Design and create GUI elements except Report Views

Jedidiah Bird:

Design Donor class/object

Design NewDonor class/object

Create methods used to manipulate Donor and/or NewDonor data

Brian Hinkle:

Design Cross-checking algorithms/class

Create methods for ensuring data has been properly entered

Dondra Richards:

Design Reports class for viewing/correcting data prior to printing

Reports Object designs

Timia Davis:

Design Printing class for printing data once it has been verified to be correct

Reports Object designs

Notes:

Two people are assigned to the reports object design and printing due to the complexity of the reports and interfacing with the printer. Additionally, there are at least two different reports that must be created, and this provides a person for each report rather than one person trying to accomplish both in the time allotted.

**Who did what for this document:**

Kevin:

Created UML diagram with input from all team members

Finalized formatting of this document

Jed:

Provided UML for Donor and New Donor – slightly modified after offline discussion

General sounding board for overall plan – assisting with overall design

Brian:

Provided UML for Cross-checking

Provides additional support for design especially regarding cross-checks

Dondra:

Provided UML for weekly Report sub-classes

Timia:

Provide UML for NewDonorReport class – some portions incorporated into Donor, EntCont, and both Report sub-classes in primary UML diagram. Also used to generate a report of any new donors added, potentially with extra information

**Class Descriptions:**

Main\_Selection – GUI button set that is the initial screen seen by the user. Opens sub-forms for user input.

EntCont – Allow user to input donation information. Names and Offering Types are validated by combo-box, and this form also opens sub-forms to allow the user to add a new name or offering type to the list if needed.

AddNewDonor – Allows user to input names and up to 5 generic notes (could be address, phone number, email, etc.) for new giving units. This will also update the text file that is the source for the Name combo-box in the EntCont class’s form.

EntMisc – Allows user to input the 2 counters’ names, the deposit ID, and the quantities of each denomination of coins and currency. This information instantiates a single object (:MiscInfo) that is used by the Cross-Check class to verify that the information entered in the EntCont class matches what the actual count of the total cash and checks is.

ReportSelect – GUI button set to allow the user to view/print the two (or three) reports.

CrossCheck – Compares the total donations entered with the total by Offering Type as well as the total donation reported as being by cash to the total cash by denomination. Also, counts the number of checks and allows the user to verify that the correct number of items is recorded on the summary report. Returns a Boolean TRUE if all cross-checks are satisfied.

NewDonor – Takes information from the AddNewDonor form and updates the Name file for the EntCont.name combo-box. Also called by the Contributor class to generate a report if new donors have been added.

Donor – Takes information from the EntCont form and stores it for the CrossCheck and Reports classes.

ContributionReport – Creates a report listing all donations sorted by Donor.name and then by Donor.offType with a section of subtotals for all offTypes used. Also has blanks that can be signed by the two counters after the report is printed.

SummaryReport – Creates a report listing the total of all donations by offType and service, with separate totals of cash and checks, and an overall total. Additionally, this report provides all the miscellaneous data about the deposit (bag number, number of items, etc.) The overall total should be the same regardless of method of adding, which is part of what the CrossCheck class does. Also has blanks for the two counters to sign after the report is printed.

NewDonorReport – Creates a report listing any new donors and the notes associated with them. Could also track the donations made the first week the new giving unit is added to the list, although this may be extra, unneeded information, depending on the users’ desires.

**UML Class Designs:**

**Timia’s UML for Contributors:**

|  |
| --- |
| NewDonorReport |
| - contributor: String  - amountFromContributor: double  - time: String  - offeringType: String  - total: double  - subtotal: double  - payment: String |
| + Contributor()  + printResults(): void  + getOfferingType(): String  + getAmount (): String  + getTypeFromContributor(): String  + getTime(): String  + getAmount(): double  + calculateTotal(): double  + displayOfferings(): void  + getSubtotal(): double  + displayDeposit(): void  + getDeposit(): double  + totalItems(): int  + getDate(): void |