Prayer Tracker Design Document

by Judah Sistrunk
Reed Lawrence
Scott Stoltzfus

Table of Contents

1. Introduction

This document is meant to describe the design of an app that helps its users keep track of their prayer life. The app takes prayer requests and stores them in an easy to read list. The user may check the app whenever they want, or they may turn on automatic alerts that will remind the user about a specific prayer. The app will keep a record of answered prayers.

The app must meet the following criteria:

- It should be able to take information about a specific prayer and display it to a user.
- It should be able to remind users to pray for a specific prayer.

- It should be able to keep a record of answered prayers
- It should let the user edit prayers.

1.1. Document Outline

- Introduction
- System Overview
- Design Considerations
    - Assumptions and Dependencies
        - Assumption - Linked lists are already implemented
        - Assumption - Interrupts or low cost timing/ delay/ scheduling functions are available
    - General Constraints
        - Memory
        - CPU speed
        - Screen size
    - Goals and Guidelines
        - Goal / guideline - Simple
        - Goal - Not very resource intensive
        - Goal - Fast
        - Guideline – must be available for android
    - Development Methods
- Architectural Strategies
    - runtime calculations from a database to conserve on memory
    - single graph memory area that is deleted then recreated when parameters change compromise between speed/reuse and memory
- System Architecture
- Policies and Tactics
- Detailed System Design
    - Prayer class – holds the information on individual prayers
    - Tracker class – holds lists of prayers, runs calculations for graphs/ statistics, sets or triggers alerts, displays different views, creates and stores prayer objects
- Glossary
- Bibliography

1.2. Document Description

1.2.1. Introduction

- The purpose of this document is to lay out the design specifications for the prayer app
- This document's intended audience is developers of the software
- The software will run on android phones, specifically running android OS 3.x.x and up
- Related documents:
    - Prayer Tracker SRS

1.2.2. System Overview

The app is a storage system for a specific class that we are calling a prayer and an interface that allows the user to view the prayers, track them, or schedule reminders based on them. The prayer holds a name, description, picture, date, etc. The system will be able to alert the user of the prayer at a specified time. Also it displays statistical data, based on the prayers, on an adjustable graph.

2. Design Considerations

This section describes many of the issues which need to be addressed or resolved before attempting to devise a complete design solution.

2.1. Assumptions and Dependencies

- The app will run on android phones
- The app will run on android operating system
- Users are Christians and know how to pray
- It is assumed that the app will be used as a prayer tracker, but it could potentially be used as a general calendar app.
- An alert class exists and is easy to implement
- GUI menu classes are built in to the android SDK and are easy to implement
- Images are supported and are easy to implement
- Date and time can be accessed by the app
- Linked lists are already implemented on Android systems
- Interrupts or low processor cost timing or delay or scheduling functions are available

2.2. General Constraints

- Hardware or software environment
- End-user environment
- Availability or volatility of resources

- Standards compliance
- Interface/protocol requirements
- Security requirements
- Memory and other capacity limitations
- Performance requirements
- Verification and validation requirements (testing)

2.3. Goals and Guidelines

- Simplicity
- Emphasis on speed versus memory use
- working, looking, or "feeling" like an existing product

2.4. Development Methods

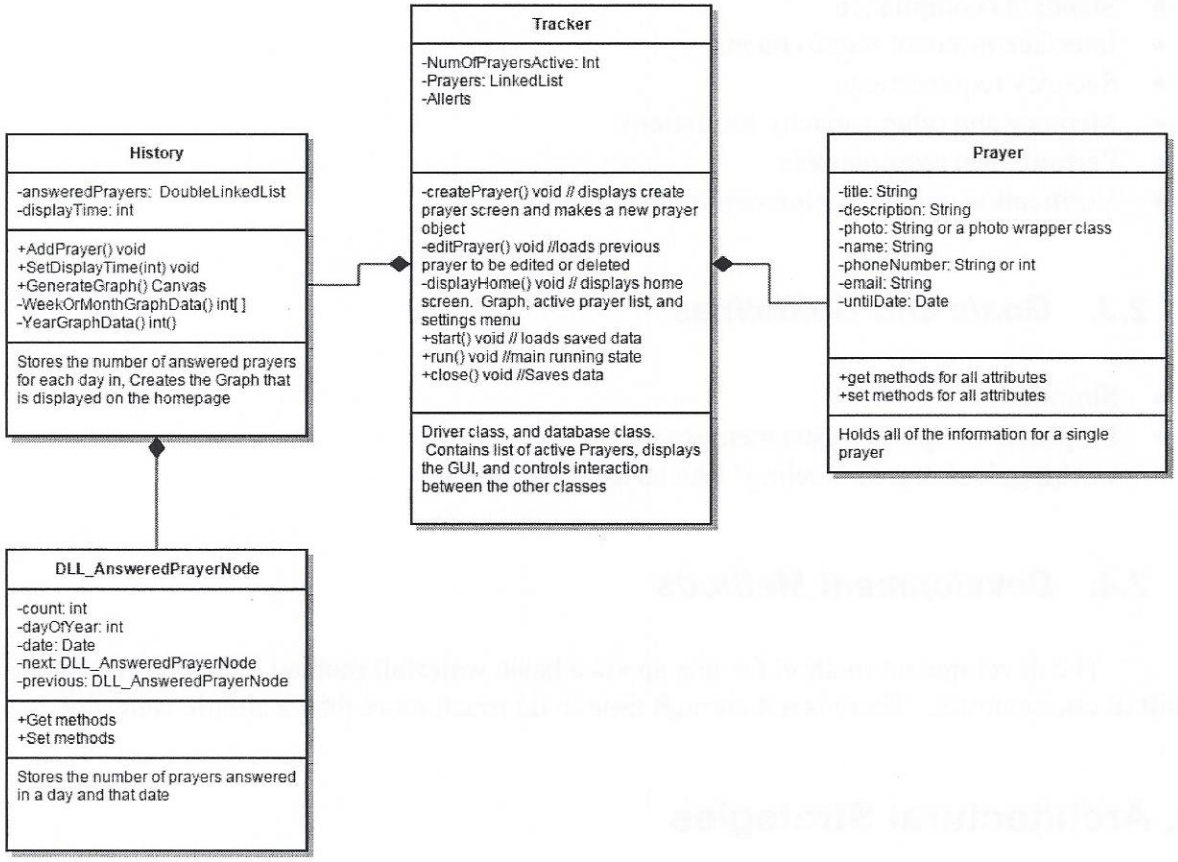
The development method for this app is a basic waterfall method. This app has to be built in one semester. There is not enough time to do much more than a simple waterfall.

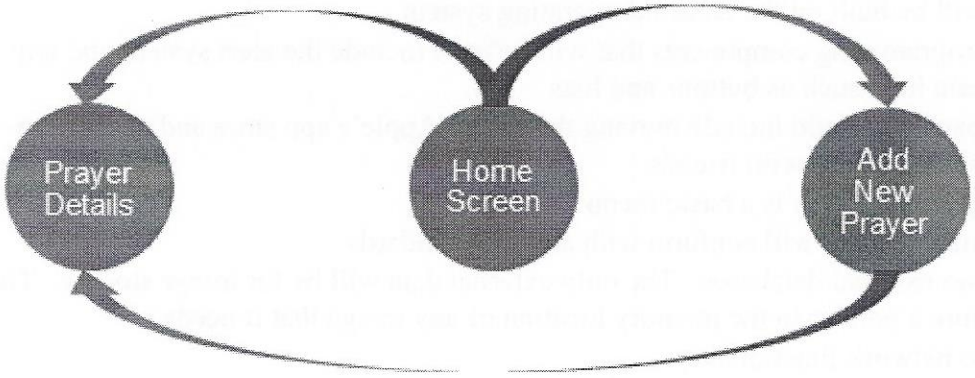
3. Architectural Strategies

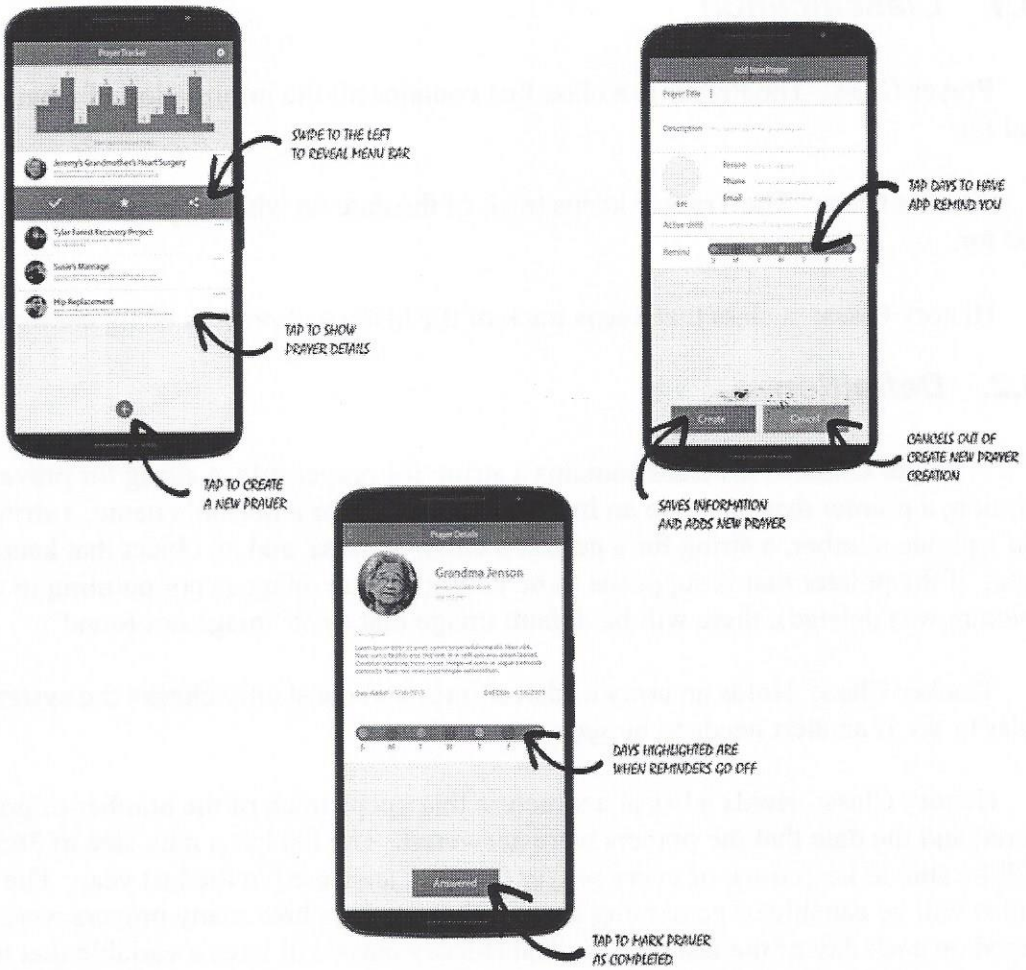
- This app will be built on the Android operating system
- Existing programming components that will be used include the alert system and any kind of menu item such as buttons and lists
- Future expansions could include moving the app to Apple's app store and adding the ability to share prayers with friends
- System input and output is a basic menu system
- Memory management will conform with android standards
- There are no external databases. The only external data will be for image storage. The app will store a pointer to the memory location of any image that it needs.
- There is no network functionality

4. System Architecture

The System must behave like a very simple database. It gets information about a prayer and displays it when requested. Also functions as a scheduler with reminders for prayers and other events. The system will also be able to keep track of prayer history.


Tracker
-NumOfPrayersActive: Int
-Prayers: LinkedList
-Allerts
-createPrayer() void // displays create prayer screen and makes a new prayer object
-editPrayer() void //loads previous prayer to be edited or deleted
-displayHome() void // displays home screen. Graph, active prayer list, and settings menu
+start() void // loads saved data
+run() void //main running state
+close() void //Saves data
Driver class, and database class. Contains list of active Prayers, displays the GUI, and controls interaction between the other classes
History
-answeredPrayers: DoubleLinkedList
-displayTime: int
+AddPrayer() void
+SetDisplayTime(int) void
+GenerateGraph() Canvas
-WeekOrMonthGraphData() int[ ]
-YearGraphData() int()
Stores the number of answered prayers for each day in, Creates the Graph that is displayed on the homepage
Prayer
-title: String
-description: String
-photo: String or a photo wrapper class
-name: String
-phoneNumber: String or int
-email: String
-untilDate: Date
+get methods for all attributes
+set methods for all attributes
Holds all of the information for a single prayer
DLL_AnsweredPrayerNode
-count: int
-dayOfYear: int
-date: Date
-next: DLL_AnsweredPrayerNode
-previous: DLL_AnsweredPrayerNode
+Get methods
+Set methods
Stores the number of prayers answered in a day and that date


Prayer Details
Home Screen
Add New Prayer



5. Policies and Tactics

- The Android dev kit will be used
- Plans for ensuring requirements traceability – a checklist
- Design several test interactions with product to ensure all requirements are met.
- No plans for maintainability, software complete upon delivery
- Standard Android Interface

6. Detailed System Design

6.1. Classification

Prayer Class: The Prayer is a class that contains all the information of what is being prayed for.

Tracker Class: The Tracker keeps track of the date on which a prayer should be prayed for.

History Class: a class that keeps track of the history of prayers being answered.

6.2. Definition

Prayer Class: This class contains a string for prayer title, a string for prayer description, a pointer that points to an image file, a string for a person's name, a string for a person's phone number, a string for a person's email address, and an object that keeps track of the date. If the pointer that is supposed to be pointing to an image is not pointing to an image (the picture was deleted), there will be default image that says "image not found".

Tracker Class: Holds an array of dates/times and constantly checks the system's calendar to see if an alert needs to be sent.

History Class: Holds a list of a structure that keeps track of the number of prayers answered and the date that the prayers were answered. The list has a max size of 365. The list will be able to keep track of every prayer that was answered in the last year. The History class also will be capable of generating a graph that displays how many prayers were answered on each day of the last period. The History class will have a variable that tracks how long a period is. A period can be a week, month, or year. The period can be decided by the user in the settings menu. The history class will also get rid of items if they are older than one calendar year. Otherwise the app could be able to store more than necessary.

6.3. Responsibilities

Prayer Class: Keeps information about a prayer.

Tracker Class: Keeps track of dates and alerts users of prayers.

History Class: Keeps track of how many prayers were answered on each day.

6.4. Constraints

Prayer Class: Since the prayer class is not actually storing an image file, it is dependent on an external image storage system. The actual file is not being stored as to save space in memory.

Tracker Class:

History Class: The history class will only keep up to 365 days of information as to preserve as much memory space as possible. The class will not store any data for days in which nothing happened. Thus the graph function must be able to recognize a gap in dates as days without activity.

6.5. Composition

Prayer Class: Stores the data for a single prayer

Tracker Class: master class for organizing and display prayers

History Class: master class for organizing and displaying info on answered prayers

AnsweredPrayer Class: Stores numbers of prayers answered on a given date.

6.6. Uses/Interactions

Prayer Class: When a prayer is answered it must be removed from the Tracker class. The class must also be able to interact with the phone's time and date system.

Tracker Class: The tracker class must be able to interact with the prayer class and delete a prayer when the prayer is answered. The tracker must also be able to interact with the time and date system, so that it will be able to notify the user when a prayer needs to be prayed. Class will also delete prayers once they have been answered and pass the necessary info to the History class for processing.

History Class: When a prayer is answered it must be received by the history class from the tracker class and created an answered prayer object. Class will also generate and display a graph of answered prayers.

AnsweredPrayer Class: Keeps track of answered prayers and the date they were answered.

6.7. Processing

Prayer Tracker class will provide a list of all active Prayers (Prayer Class) for display on the main menu. This list will not actually be stored, but generated each time the user starts the program, it will traverse the linked list of prayers and compiled and displayed at runtime. Marking a prayer answered will immediately delete the prayer from the active prayer list and generate an Answered prayer and pass that to the History class.

History class will keep track Answered Prayer class double linked list. The History class will generate a graph of the answered prayers by year, subdivided into months; by month, subdivided into days; or by week, subdivided into days. History class will also have to accept any prayers passed to it as "answered" by the Prayer Tracker Class. Finally the History class will have to delete prayers more than 1 year old, grouped by the month. E.G. if it is March then all prayers answered in February of the previous year should be deleted.

Prayer class will have the ability to store all information associated with a single prayer. This class will also function as a node in a linked list. The only operations it should preform are constructors. By the current requirements no editing operations are needed although these options would greatly increase functionality.

Answered Prayer class will represent a single day in which any number of prayers can be answered. This class will function as a node in a double linked list. This class will need to store the number of prayers answered a given day and the date of that day.

WeekOrMonthGraphData function Algorithm

Returns an array of the number of prayers for the last 7 or 30 days

```
WeekOrMonthGraphData()
    //create variables
    int endDate
    int graphDate[ ]
    Date currentDate
    DLL_AnsweredPrayerNode currentNode

    endDate = current date from system converted to DayOfYear // check calendar class
    If(displayTime = = 1)
        graphData = int[7]
    If(displayTime = = 2)
        graphData = int[30]

    currentNode = first AnsweredPrayers node
    For( i = graphData.lenght() ; i>0 ; i--)
        If(endDate> currentNode.getDayOfYear())
            graphData[i] = 0
            endDate - -
        elseif(endDate = = currentNode.getDayOfYear())
            graphData[i] = currentNode.getCount()
            endDate - -
            currentNode = AnsweredPrayers.getNext()
        if(endDate = = 0)
            endDate = 365
    return graphData
```

6.8. *Interface/Exports*

7. Glossary

Prayer: usually used to refer to a religious action of calling upon God. Here it is used to describe a class that holds information that the user of the app wants to pray about.

8. Bibliography