PiMask

Mobile File Manager

Functional Specification

Version 0.1

Anup Kher
anup.kher.1990@gmail.com

Version History

Version	Changes
0.1	Started the process for development of a prototype application

Introduction

The mobile file manager functionality is responsible for retrieving and arranging various types of files on the user's mobile device for the purpose of backups. There are two aspects of this functionality. First, accessing user's data on the mobile phone like photos, videos, contacts and documents. Second, preparing the user's data to be synchronized with the controller, the Raspberry Pi.

References

1. http://developer.android.com/training/contacts-provider/index.html
2. https://developer.apple.com/library/ios/documentation/ContactData/Conceptual/AddressBookProgrammingGuideforiPhone/Introduction.html

Figures

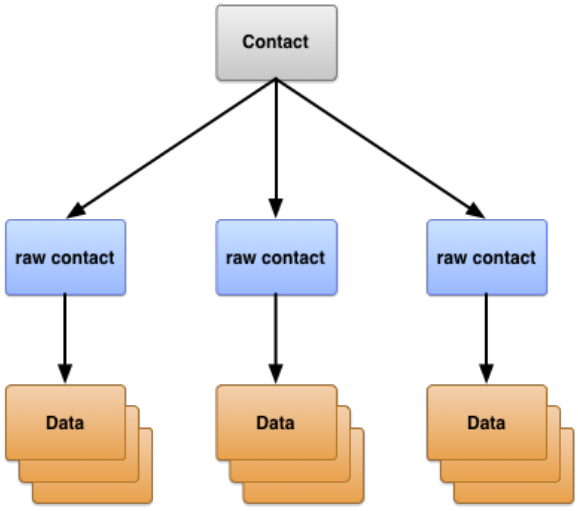

Fig. 1 Contacts Provider table structure (Android)

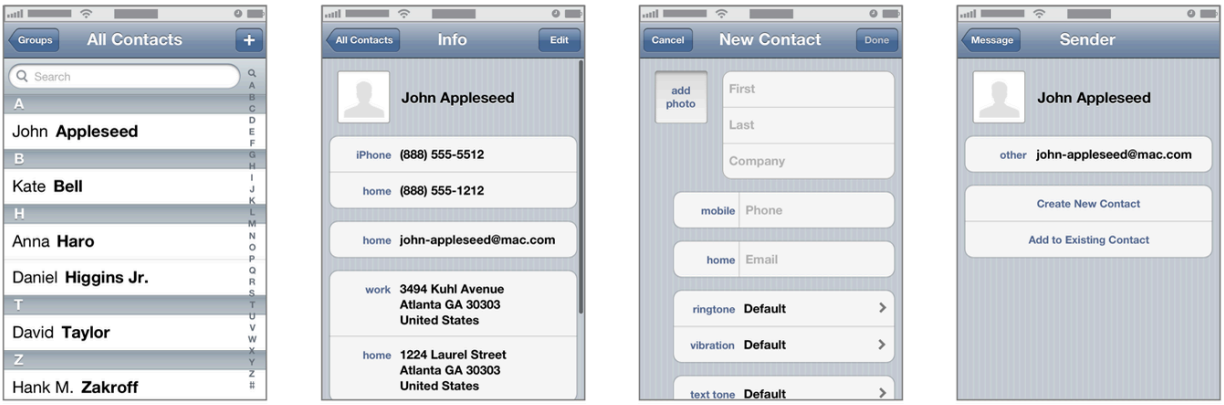

Fig. 2 Various view controllers for accessing contacts (iOS)

Requirements

- Mobile device
- Raspberry Pi
- Mobile application for accessing and organizing data on the mobile device
- WiFi / Bluetooth connection to Raspberry Pi
- XCode for iOS development
- Android Studio for Android development

Functional Overview

- A mobile application will be developed specifically for this purpose. This mobile application will implement the above mentioned features in a seamless manner.
- The primary function of the application is to gain access to user data in the mobile device. The data we will be concentrating on are photos, videos, contacts and important user documents. This data will then be organized and sorted using a timestamp.
- The previous step will make it possible for the data to be sent to the Raspberry Pi for

storage and further processing.

- Once the data is organized, it will be transferred to the Raspberry Pi where it will be stored for future use.
- Ideally, our controller, i.e. Raspberry Pi will hold this data in chronological order so that the user can retrieve it in the future.
- The data on Raspberry Pi can then be persisted on a secondary storage which will be a repository of backups.

Configuration/ External Interfaces

- No external configurations required for this functionality
- Finding relevant APIs on both mobile platforms for implementing the required features

Debug

Logging

The controller (Raspberry Pi) will be running a server which will be responsible for logging all the activities and interactions between the controller and the mobile device. Some of the activities which may be logged are:

- Status of the process of synchronization between the mobile device and the server
- The number of mobile devices currently in the network
- The devices currently connected to the controller

Implementation

1. The implementation of the functionality will be carried out using a fully-featured mobile application
2. There are two main sub-tasks which need to be concentrated on, data/files access and preparation for synchronization with the controller
3. For the purpose of developing the mobile application, the current mobile ecosystems need to be looked at
4. Currently, there are three dominant mobile ecosystems in the market. Apple's iOS mobile operating system, Google's Android mobile operating system and Microsoft's Windows mobile operating system. For the purpose of our development, we will be concentrating on iOS and Android ecosystems
5. Two different mobile ecosystems will require two separate applications
6. After careful research we concluded that it is not feasible to develop a single application which can be deployed on these two mobile platforms and thus will require the use of two separate programming languages
7. iOS will require the application to be developed using either Objective-C or Swift. We will be concentrating on Swift.
8. Android will require the use of Java as the primary programming language.

Testing

Performance Tests

The mobile device will undergo the following performance tests:

- Stress tests
- Network tests
    - WiFi connectivity tests
    - Bluetooth connection tests
- Mobile application benchmarking

Appendix

N/A