Google Medical Records: Android Application

Siddhartha Abburi, Computing Studies, Arizona State University

Abstract—Mobile devices are offering many ways to improve the health care through the application of technology by allowing users to access their health care data and services in a secured and easy to use environment at any location and time. With a Google Health account and GMR mobile application, one can access, store, manage and share all of their health and wellness information from their mobile device and make those available during an emergency in a secured manner.

In this project, Android application development interfaces and Eclipse IDE development environment, designs were used to develop a mobile application on an Android platform using JAVA. The application’s user interface was designed keeping in view of the user interface guidelines [1] as mentioned by the Android UI team, which shall be simple and easy to use from an Android device. The user has to provide the security pin correctly to access the secured information in the application. Authorization with ClientLogin API [2] is used using user’s login name and password to grant access to their Google service data from inside the application, allowing the user to view, add, edit or remove the service data as required using their application interface. After the authentication, Google issues a token that can be referenced in all subsequent requests.

Index Terms—Google Health, Android Application, Medical Records, Smart Phone Application.

I. INTRODUCTION

With the initiation of mobile devices and usage of electronic health record [3] systems, we will be able to reduce the usage of paper-based health records of the patients by simplifying and improving the quality, work process and cost. Mobile devices offer new ways for users to access health care data and services in a safe and easy to use environment. These new applications must be simple to build, install, experiment and retain, and they should be easily integrated on a reliable infrastructure [4]. Every user might have had more than one doctor, prescribed medication, injuries, allergies, vaccinations etc and it’s also hard to remember what happened when, what the treatment was or even which doctor handled the situation. Centralizing this information is the advantage that Google Health provides and being able to access this information from any location and time is the core aim of this mobile application. Furthermore the user should be able to share his personal information with others. The user can access his health information even without having access to the internet. In Google Medical Records (GMR) application we used Google Health API [5] to communicate with Google’s servers to retrieve a view of the PHR [Personal Health Record] in the form of a Continuity of Care Record (CCR) [6]. The XML-based CCR is then parsed and displayed on their smart phone. This mobile application can be easily migrated to a different health service provider.

II. LITERATURE REVIEW

A. Android Architecture

The architecture enables the developers to reuse, share activities, services and data with other applications.

![Android System Architecture](image)
The above figure [Figure 1] shows the architecture of the Android system and all the services are provided by the layers below it. The runtime and application services are managed through the application framework which exposes the libraries and the kernel.

Even though Linux kernel [8] is used by the Android to manage memory, networking and operating system services, the user and the Android program will never be able to view/make Linux calls directly. The uppermost layer provides a set of core applications like browser, contacts, widgets, calendar, etc and it’s possible to run more than one application at the same time. The Application Framework can be used to reassemble the functions used by other applications with the help of managers and content providers. The layer below the Application Framework consist of two parts, a set of Libraries which are written in C/C++ and an Android run time which consists of a set of core Libraries. The Libraries such as SQLite, WebKit, browser engine, etc are called using the Java interface and the core libraries provide the functionalities available in the main java programming language.

B. Android Programming Framework

To develop an application, one has to install the Android SDK, the Eclipse IDE and the Java Development Kit (JDK).

Java programming is used to develop the application and it is compiled along with the data files using the Android SDK tools into an Android package with .apk suffix which is used to install the application on the Android device.

The Android SDK consists of a set of tools like libraries, emulator, Android Asset Packaging Tool (AAPT), Android Debug Bridge (ADB), etc which are executed using Eclipse integrated development environment (IDE). It also consists of an emulator which is used to build and test the Android application. Using emulator, Android platform and hardware, the options on which the application is built can also be specified.

C. Google Health Authentication Process

To access the content present in the Google server, the user provides a one-time authorization to transfer the information securely by providing their Google login details. The user then receives a ClientLogin session token from the Google authorization service which is stored in the applications database using SQLite for subsequent requests and ongoing updates. Once the Google service recognizes the token, it sends the requested data.

The Health feeds can be accessed by using AuthSub/OAuth authentication which is user/profile specific or by using ClientLogin [6] authentication which is not bounded to any specific profile/user.

<table>
<thead>
<tr>
<th>Feed URL Suffix</th>
<th>AuthSub/OAuth</th>
<th>ClientLogin</th>
</tr>
</thead>
<tbody>
<tr>
<td>register/default</td>
<td>POST</td>
<td>GET POST PUT DELETE</td>
</tr>
<tr>
<td>register/ui/profileID</td>
<td>POST</td>
<td>GET POST PUT DELETE</td>
</tr>
<tr>
<td>profile/default</td>
<td>GET</td>
<td>GET POST PUT DELETE</td>
</tr>
<tr>
<td>profile/ui/profileID</td>
<td>GET</td>
<td>GET POST PUT DELETE</td>
</tr>
<tr>
<td>profile/list</td>
<td>GET</td>
<td>GET POST PUT DELETE</td>
</tr>
</tbody>
</table>

*Operations supported by Authentication Type [6]*

While sending a HTTPS POST request to the server using Client Login, the default Content-type should be: “application/x-www-form-urlencoded” and the parameters; accountType, Email, Passwd, service,
source, logintoken should be embedded in the body of the post.

In the GMR application, a Hashtable<Integer, String> is used to store all the possible responses from the Google server and compared to the response obtained which contains the authorization token labelled as “Auth” in the body of the response.

**Sample Code [9]:**

```plaintext
POST /accounts/ClientLogin HTTP/1.0
Content-type: application/x-www-form-urlencoded
accountType=HOSTED_OR_GOOGLE&Email=jon doe@gmail.com&Passwd=north23AZ&service=cl&source=Gulp-CalGulp-1.05
```

### D. Using SQLite to Store the Health Records

SQLite is a single-tier, open source relational database management system which is implemented as a C library and is included in the Android software stack. Every application is integrated to a SQLite database individually reducing the external dependencies, latency and synchronization.

In SQLite, every column need not be conformed to a single type and values are entered for each row individually resulting in the elimination of type checking while assigning or extracting values for each row from the database. Using SQLiteDatabase class, we can perform delete, insert and update operations when required.

In the GMR application, manual synchronization is used to upload or download the data from the server asynchronously. When the application is installed, the DBAdapter.java file will create all the required tables using the execSQL command and checks wether the application pin has been set or not, if not the security pin is created and stored in the database. All the queries are then stored as a string and are called when required. When the user gets authenticated after providing his username and password, all the user profiles are stored in the database using a double array and depending on the profile selected, the information is retrieved from the database using an ArrayList, Cursor and displayed on the screen.

The SQLite database in the GMR application stores only the latest data related to the authenticated user as available on the server. Every time the user performs synchronization, the data available in the applications database is updated with the new data obtained from the server.

![Figure 5: Client Architecture with SQLite Database [10]](image)

The above figure [Figure 5] shows the SQLite Mobile client architecture [10] which consists of a SQLite database which is installed independently, a Mobile client which consists of a sync engine to upload and download the data from the server and the GMR Mobile application which interacts with both the SQLite database to fetch and display the data on the mobile screen and the Sync engine to get/post the data from the server.

![Figure 6: Database Schema of GMR Application](image)

The above figure [Figure 6] shows the database schema of the GMR application, the login table is independent of all the other tables in the database and the ProfileTable consists of ProfileId as its primary key which is used as
a foreign key in every other table. Using the ProfileId and its respective primary key, the information is retrieved from the database and is displayed on the screen.

E. Parsing the CCR Records

The Continuity of Care Record (CCR) [11] is a health record standard developed jointly by ASTM International, the Massachusetts Medical Society (MMS), the Healthcare Information and Management Systems Society (HIMSS), the American Academy of Family Physicians (AAFP), the American Academy of Pediatrics (AAP), and other health informatics vendors, which are used to create flexible documents consisting of patient’s health information and are used to send health records electronically between the two users using XML format.

The XML data model is optimized for efficient memory usage to minimize the time required for parsing and serialization, and android has an efficient built-in XML pull parser. All the objects within the CCR have ObjectIDs [12] which are generated by the originating system to uniquely identify every explicit instance.

In the below figure [Figure 7], the elements to the right on a particular branch are extensions of the elements to their left. The only standard being implemented by Google Health from the health care industry is CCR (Continuity of Care Record). The HTTP GET request is sent to the Google health server to retrieve all the profiles associated with the ClientLogin token as a single CCR document. In the GMR application, the XMLParser.java is used to import and export all the retrieved CCR data to enable automated healthcare information transmission, making it easy for the users. Using CCR, user health details such as allergies, test results, medications, etc can be obtained. The CCR consists of three components: a ‘Header’, ‘Footer’ and a component consisting of <Body> element. In XMLParser.java, two-dimensional arrays were used to parse through the CCR document using indexOf() method to return the position of the specified value in the string and store it using the CCR XML schema which defines the XML representation of the CCR data elements as reference. A set of CharAt() methods are used to parse the CCR document and store it in the respective two-dimensional array.

While building the GMR application, parsing the CCR document and saving it in the database based on the user profile was the toughest task. Firstly, creating a sample application with which one could successfully connect to the Google server and then save the retrieved CCR document as a XML file; then parse the XML document by looking at the XML schema of the CCR document which was obtained based on the user profile, then directly parse the XML data which was obtained electronically without saving it as XML file but saving it directly in the database using SQLite.

III. GOOGLE MEDICAL RECORDS APPLICATION

A. User Interface

The applications user interface was designed keeping in view of android’s user interface guidelines [1], thus maintaining consistency between the application and that of android platform making it user friendly, simple and predictable.

When the user uses this application for the first time, they have to sign up once for a one-time upfront authorization to transfer the information securely. Most of the screens have a unified look and feel throughout the application and consist of spinners, expanding menus etc when required and displaying only the most
frequently used operations on every screen. Most of the screens were designed using nested layouts.

In the below figure [Figure 8], the user has to enter a six digit pin number to access the login page of the application. Then the user has to enter his Google login details to access their online health records. Once the user gets authenticated, they can view, edit and add their health information using the mobile application. The user can have more than one profile in the same ID and he can change the profile by clicking on the image on top of the home page. Every profile will have its own specific set of data. The user can change the current Google health account within the application and the user can synchronize the data with the server only when he wants to. Once the user is logged into his health account, he will be able to access his health records from all his profiles.

![Figure 8: Usecase Diagram of the GMR Application](image)

Irrespective of the user’s instance, the home button on the phone is configured to display the exit option to quit the GMR application. The option menu changes depending on the user’s instance inside the application, if the user is viewing a health record which is already present then the user gets an “exit” option along with “update” and “delete”. If the user accessed an empty record, he would get an “exit” and an “add” option. The return button on the android mobile device is configured to take the user to the previous screen.

In the above figure [Figure 9], the user needs to login using their Gmail username and password which will then give them an option whether they want to sync the data present in the android mobile with the data present in the Google web service, then it redirects the user to the home page.

Irrespective of the user’s profile, their personal information containing Age, Sex, Race and Blood Type will always be constant. The user can also change their Google account by accessing the settings menu. While entering your health records, the CCR health record standard provides additional health information for the users to choose from; for example when the user is suffering from Fever and starts entering his current problem using the mobile application in the conditions or symptom field with the letter “F”, it provides a list of all the conditions/symptoms available to the user, to select an option starting with the letter “F” and the user can scroll down the given options and select “Fever”. An in-built calendar is also used in the user interface to select the conditions start and stop date for any given problem.

The below figure [Figure 10] shows the home page of GMR’s application where the user can see their current/past medications along with their profile name for which the data is shown. The user can change their profile id by clicking on the image which is being displayed on the top right of the screen.

![Figure 9: Login Page](image)

![Figure 10: Home Page of GMR Application](image)
The less frequent menu options are displayed if required by clicking on the *more* menu. The user can directly access his current/past medication, profiles, conditions and his test results directly from the home page.

The art and iconography were taken from Glyphish [12] which provides icons for mobile applications. Every menu option selected by the user opens a new screen and every screen is represented by a java class file. The “intent” function is used to jump from one screen to another within the application and the “bundle” function is used to maintain and pass the program state from one instance of the application to the other.

B. Application Security

The user’s personal health information is always kept confidential, secured using Secure Socket Layer (SSL) encryption, firewalls and other technologies and it should not be accessible to anyone other than the user unless he/she decides to share it with others. Once the user provides his authentication information, a session token will be provided which will be saved and the user can access their medical records through their Android device. To prevent someone from accessing this information other than the user, a six digit application pin is requested every time somebody starts using this application, thus keeping the health information secured. The first time the user installs the application, a PinTable is created in the database which requests the user to enter the security pin for the health application. The user can later change the security pin by selecting the ‘ChangePin’ option in the ‘Settings’ menu. Even when the user changes their Google health account and re-enters his login details, the security pin will not alter but the rest of the tables are recreated. While developing this application, we built a sample program to test it by searching the text on the screen, and then tested it by comparing the actual result, entered pin and accordingly print the message to check the result.

If the user enters the pin successfully, they are given an option to perform sync between the data present in the Android applications database and the Google Health Service. If “yes”, the data which is already present in the database is deleted and updated with the new data obtained from the server. If “no”, the data which is already present in the database will be displayed.
C. Profile Selection

Every user can maintain more than one profile to personalize his health data depending on the type of treatment. The user can now share a set of his personalized health record with a doctor/well wisher rather than sharing all his records which might not be required. He can also choose to view a user’s profile which is being shared with him.

In the GMR application, every profile has a separate set of tables in the database and depending on the profile selected; the data is retrieved and displayed to the user.

In this application, the user can change the profile that he is currently viewing by clicking on the image on the top right corner of the home page or by selecting the option from the more screen.

D. Add and Edit Health Information

The information between the Mobile device and the Google Health Service is sent, received in an XML format as a CCR document using ClientLogin API which is later parsed and stored in the local database using SQLite and then displayed on the screen individually. The user is asked if he wants to synchronize the local data with the server as soon as he starts the application and enters the PIN correctly or when the user adds/edits the current data.

Every time the user adds/edits the data using the device, the data is sent back to the server as a CCR document and synchronization is performed. The device doesn’t store any past records. Once changed, updated or synchronized then all the tables in the current database are dropped and recreated with the new data, avoiding data overlapping which results in saving a lot of space on the mobile phone.

IV. ANALYSIS AND DISCUSSION

GMR Application: The Google Medical Records application for Android uses ClientLogin for Authentication, Continuity of Care Records (CCR) to exchange information with a web-based platform and android’s SQLite to store the data in the mobile device making it possible to integrate with any personal health record solutions.

To integrate the current GMR application with a different server, the URL of the web-based service provider in the authentication process should be changed and since every service has different names for the given data types, the XMLParser.java should be modified to parse the CCR file and save it accordingly in the database using SQLite.

Google Health: Unable to successfully divert their consumer-centered approach by using their other product domains towards healthcare which might have revolutionized the health experiences of millions of people, Google has decided to discontinue their online service from next year.

With Google discontinuing their health service, the users can use other health web-based platforms like Microsoft HealthVault [14] to store their health and wellness information.

Since Google Health is being discontinued, the Google users can download the current health records in the form of CCR and import them into other online personal health tools such as Microsoft HealthVault.
**Microsoft HealthVault:** Unlike Google Health, Microsoft HealthVault services are open to both users and healthcare professionals. The user is requested to authorize a set of health tools and devices which will collect their information and provides valuable feedback, taking healthcare to the next level.

Microsoft HealthVault provides the users with many Health Solution options like weight reduction, managing a chronic condition, tracking their prescription etc. The users can also upload Documents, Medical Image Study, Continuity of Care Records (CCR) and Continuity of Care Documents (CCD) online.

Since both Google Health and Microsoft HealthVault use the industry standard which is Continuity of Care Records (CCR), the GMR android health application can be modified to obtain the user’s health information from Microsoft HealthVault.

**Security:** Health information of the users being very sensitive and with the risk of losing the mobile device and exposing their personal health records is a major issue. Protecting the user information present in the application is really important.

To make this application secured, a six-digit security pin has been introduced. The user has to enter the security pin before using the application, hence making this application secured.

Currently the GMR application is protected only by the transport level protocols and in the future we would implement self-protecting electronic medical records using Attribute-based encryption (ABE) [13] which would enable the users to have their own control policies by restricting who can read their health records on their mobile device.

**Multiple Profiles:** Every user can have more than one profile, providing the users to access, manage multiple health records very easily. The user can share only a part of his personal information with which he is comfortable with. For example, if the user has two different doctors with which he wants to share his health records with, he can create separate profiles for them, the user would want to share limited information with one of the doctors while sharing everything with the other. Most of the information is being repeated twice resulting in repetition of data. So the data stored in the SQLite database is not unique and wastes a lot of space.

**Access to Internet:** The application stores the health records in a built-in database, which can be accessed even without the internet. The user can sync the application with the server only if required, resulting in reducing the usage of the internet. At the same time, the user cannot update or add any of his medications or other health records without accessing the internet.

**Type of Health Records:** Google health is a new product that offers users to access, collect, store, share and manage their medical records and health information but as of now it does not support nutrition and exercise information as well as images and x-rays.

**Notifications:** The GMR application as of now cannot interact with any other application installed in the mobile device like a calendar or alarm clock which can be used to send notifications to the user regarding his appointments with the doctor or medication.

**XMLParser:** The CCR document consisting of user’s medical records use XML format to send the data electronically and to successfully parse the data obtained and store it in the application database using SQLite. In the initial stages while building the GMR application, parsing the CCR document and storing it in the database was the most challenging part of the project. After successfully connecting to the server, the CCR document was saved as a XML document which was then parsed and broken down into individual categories; like Medication, Allergies, etc and saved in the database.

**V. CONCLUSION**

After presenting the GMR mobile application for Android, a user-friendly model that allows the users to access their health information in a secured and easy to use environment at any location and time; while the online health record standards are still evolving, adding more functionality to enhance the user experience in under consideration. It is believed that this application would help a large segment of the users by providing a secured access to their health records. In the future this application might allow the users to upload images, x-rays, etc and effectively use this application even without having any access to the internet and increase in the number of health data providers and third party services along with a web user interface which provides effective feedback to the users. The GMR application can also be migrated to a different health service provider easily.
After launching the application, a reliable user feedback and statistics regarding the performance of the applications could be gathered and the important changes required to improve the application can be implemented.

References


