Med-Lab Report Viewer

Helani Madurasinghe
Department of Computer Science & Engineering
University of Moratuwa
Sri Lanka
helani.madurasinghe.11@cse.mrt.ac.lk

***Abstract*—In this paper, the project on developing a web based Android application named 'Med-Lab Report Viewer' is discussed in terms of the need for such application, the nature of the application, which technologies and tools were being used, along with how the overall development process was carried out. Med-Lab Report Viewer application lets the medical laboratory to store details of patients, store medical reports, send medical reports to patients and notify them the availability of medical reports. The patients can collect the medical report without visiting the laboratory via their mobile devices. This software application gives the ability for patients to keep their medical reports in their mobile devices and maintain a proper medical history. The project was done according to the RUP software development process with two iterations for three months and finally an initial version was obtained as a successful outcome.**

Keywords—Medical Reports; Android; Web services; GCM; Spring; Mavan

I. INTRODUCTION

There are many small scale medical labs in Sri Lanka which are not part of private hospitals, but provide service to the patients who consult doctors in individual dispensaries. These patients have to visit the medical lab at least twice to give the blood sample and to collect the report. The easiest solution to address above problems is to introduce a "**Medical Lab Report Viewer**" application to the patients. They can use their mobile device to easily access the medical lab services remotely and can show the report to the doctor using the device.

Android has become the most popular mobile device OS in Asian counties. In Sri Lanka many people hold Android devices and it is useful to develop a system that addresses the problem as an Android app.

The design and development of the project is done using the Rational Unified Process as the development methodology. Object oriented approach is used for the implementation. During the system development life cycle of the project we go through four phases following the unified process: inception, elaboration, construction, and transition. The life cycle of the system is composed through the iteration of these phases. Practices under software engineering discipline are highly adhered.

II. BACKGROUND STUDY

A. Cloud Computing

Cloud computing [15] is the delivery of services of the computing rather than a product, whereby shared resources, software, and information are provided to computers and other devices as a utility over a network. Cloud computing also focuses on maximizing the effectiveness of the shared resources. Cloud resources are usually not only shared by multiple users but are also dynamically reallocated per demand.

B. Mobile devices and Applications

Mobile devices have become common devices that many people use today. Even in a country like Sri Lanka many people use smartphones. Almost all of them use various kinds of mobile applications. Apps are usually available through application distribution platforms; some apps are free, while others must be purchased. Usually, they are downloaded from the platform to a target device such as an Android device.

Mobile apps were originally offered for general productivity and information retrieval, including email, calendar, contacts, and stock market and weather information. However, public demand and the availability of developer tools drove rapid expansion into other categories, such as mobile games, factory automation, GPS and location-based services, banking, order-tracking, ticket purchases and recently mobile medical apps.

C. Mobile Computing

Mobile computing [13] is human–computer interaction by which a computer is expected to be transported during normal usage. Mobile computing involves mobile communication, mobile hardware, and mobile software. Communication issues include ad hoc and infrastructure networks as well as communication properties, protocols, data formats and concrete technologies. Mobile software deals with the characteristics and requirements of mobile applications. The user of a mobile computing environment should be able to access data, information or other logical objects from any device in any network while on the move.

D. Mobile Application Development

Mobile application development [14] is the process by which application software is developed for low-power handheld devices, such as PDAs, enterprise digital assistants or mobile phones. These applications can be pre-installed on phones during manufacturing, downloaded by customers from various mobile software distribution platforms, or delivered as web applications using server-side or client-side processing (e.g. JavaScript) to provide an "application-like" experience within a web browser.

Application software developers also have to consider a lengthy array of screen sizes, hardware specifications and configurations because of intense competition in mobile software and changes within each of the platforms. Mobile app development has been steadily growing, both in terms of revenues and jobs created.

III. Problem Statement

There are many small scale medical labs in Sri Lanka which are not part of private hospitals, but provide service to the patients who consult doctors in individual dispensaries. These patients have to visit the medical lab at least twice to give the blood sample and to collect the report. And the patients have to maintain their own folder to collect their medical reports in case they have taken multiple medical reports within a particular period of time.

IV. Proposed Solusion

The easiest solution to address above problems is to introduce a "**Medical Lab Report Viewer**" application to the patients. They can use their Android device to easily access the medical lab services remotely and can show the report to the doctor using the device. Then they have to visit the lab only once to give the blood sample. Customer's medical report history will be stored in medical lab's database and the data will be sent to the customers' mobile device on requests.

Since today Android devices are popular in the society, the medical lab can provide a better service to their customers by introducing this application while increasing the efficiency and elevating the profit.

A. Targeted Users

The product is recommended for the owners/ managers of small scale medical laboratories. They can provide this Android app to the customers who need to view their medical reports remotely. The users are of two types: the customers of the lab, who use the Med-App Report Viewer and the workers of the lab who use the Med-App Assistant Interface.

B. Operating Environment

The Med-App API is hosted on a web server, thus operating on the cloud. The Med-App Assistant Interface is operating as a standalone application on a PC at a small scale medical laboratory. The Med-App Report Viewer is running on the Android devices of the registered customers of the medical laboratory.

C. Scope and Constraints

This project targets only small scale medical laboratories. The outcome helps the patients only when the doctor's dispensary is away from the medical lab. The mobile application usage is restricted to Android devices users. The system facilitates the services only for blood report management; lab management or payment management is not supported. The system can be integrated with an existing medical lab management system.

D. Assumptions

It is assumed that the patients/customers can do their payment manually at the lab because they have to visit the medical laboratory at least once to give the blood sample. Thus a payment method via the app is not embedded to this system.

V. Design

This system uses a layered architecture along with the client-server concept. The system has three major sub components which can be listed as:

- Med-Lab Assistant Interface (Desktop application)
- Med-Lab Report viewer (Android App)
- Med-Lab Assistant API (Web service)

Each was implemented to have layered architecture inside. Apart from that the architectural concept that interconnects each of the components is client-server model. The system design overview is shown in Fig. 1.

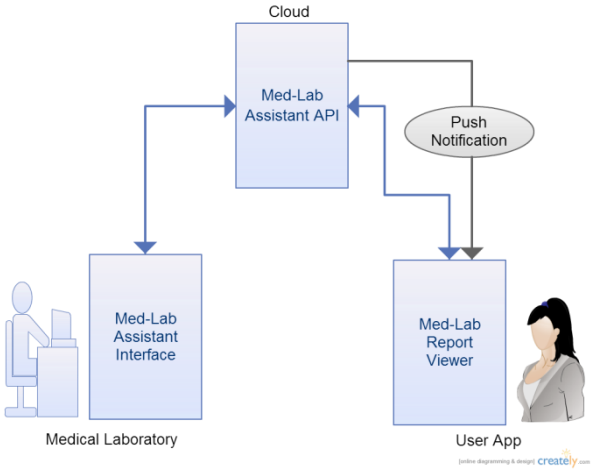

Fig. 1. System design overview.

A. Med-Lab Assistant Interface (Desktop Application)

The users of Med-Lab Assistant Interface are the system operators at the medical laboratory. He/she is able to log into the system, add new patients (customers), enter, store, and view customer details and medical report data. After finish entering medical report data he is able to send a message/notification to the patients to indicate the completion of processes. This component of the system is a standalone application running on the PC at the medical laboratory.

B. Med-Lab Report Viewer (Android App)

The users of Med-Lab Report Viewer are the patients who need to get blood/medical reports from the lab. Once they are registered as a customer at the lab, they are provided with the Android application. They can register their device for the service of receiving blood report to the phone. They receive a message/notification once the report is released. Then they are able to view the report using a mobile app when he visits the doctor next time.

C. *Med-Lab Assistant API (Web Service)*

The API is the heart of the project which provides services to both the lab users and report viewer app users. The API is web hosted along with the database such a way that it can provide services to both other components via internet.

D. *Developer Requirements vs. Design Decisions*

When designing the architecture of the system, not only the user requirements, but also the developer requirements should be considered.

- Maintainability: The software should be easy to maintain. It should have ability of repair or replace faulty or worn-out components without having to replace still-working parts, and capability of cope with the changing requirement or environment
- Reusability: The components of the system should be reusable
- High Cohesiveness: The small components of the system should focus on specific small tasks
- Low Coupling: The function of one component should be independent of others. So the changing of one component will not affect others
- Extensibility: the implementation takes future growth into consideration. New capabilities can be added to the software without major changes to the underlying architecture.
- Compatibility: The software is able to operate with other products that are designed for interoperability with another product.
- Modularity: the resulting software comprises well defined, independent components.

The layered architectural approach that has been selected provides a way to achieve these requirements.

VI. Implementation

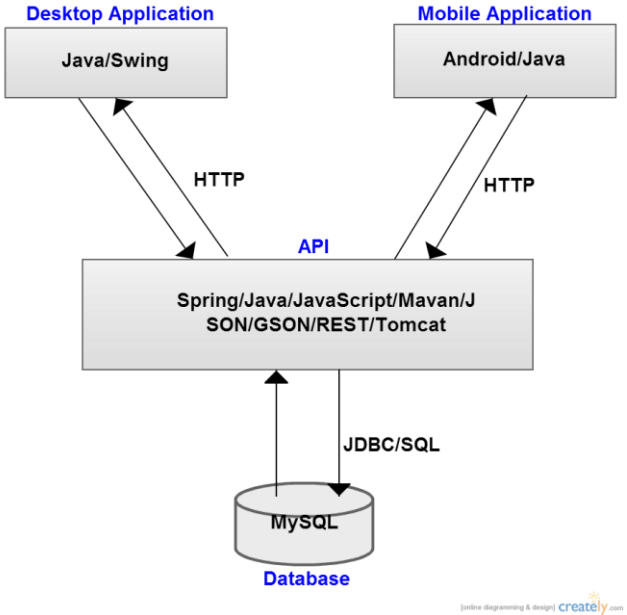

Fig. 2. Implementation architecture.

The implementation architecture of the system shows how the designed software architecture is implemented using tools and technologies. Fig. 2 shows the use of tools and technologies for the development of each component as well as the connections between the components.

A. *Dependency Injection*

Dependency injection is a software design pattern in which an injection of a service is present. An injection is the passing of dependency (service) to a dependent object (a client). The service is made part of the client's state. Passing the service to the client, rather than allowing a client to build or find the service, is the fundamental idea of the pattern. This provides loose coupling and the result is a set of more independent objects that are easy to maintain and test. Use of spring framework provided the advantage of dependency injection to my implementation.

B. *Technologies and Tools*

1) Android: Android is a mobile OS based on the Linux kernel and currently developed by Google. With a UI based on direct manipulation, Android is designed primarily for touchscreen mobile devices such as smartphones and tablet computers. Android is the most popular mobile OS. Android was selected as the platform of this project as it is a popular OS among mobile users today. Android has a growing selection of third party applications, which can be acquired by users either through an app store such as Google Play or the Amazon Appstore, or by downloading and installing the application's APK file from a third-party site. Google Play Store allows users to browse, download and update applications published by Google and third-party developers, and the Play Store client application is pre-installed on devices that comply with Google's compatibility requirements and license the Google Mobile Services software [1] Availability of means to reach users is another reason for selecting Android.

2) Spring Framework: The Spring Framework is an open source application framework and inversion of control container for the Java platform. The framework's core features can be used by any Java application, but there are extensions for building web applications on top of the Java EE platform. Spring framework was selected for the development of the project as it supports JavaSpring has MVC architecture (Model-View-Controller). The Model-view-controller design pattern helps in separating the business logic, presentation logic and navigation logic [9][10][11]. Support for MVC architecture is another reason for selecting Spring allowing the application to fit to MVC architecture.

3) Google Cloud Messaging Service (GCM): Google Cloud Messaging (GCM) is a service that helps developers to send data from servers to their Android applications on Android devices, or from servers to their Chrome apps and extensions [2][3][4]. It has been used in the development of

the project as it provides free means to send notification to Android devices.

4) Java: Java is an object-oriented programming language developed by Sun Microsystems. Java is a platform-independent, multi-threaded programming environment designed for creating programs and applications for the Internet and Intranets. One characteristic of Java is portability, which means that computer programs written in the Java language must run similarly on any hardware/operating-system platform. Java is selected for the development of mobile application as it is the most preferable language for Android development. Most of the Android libraries are available in Java. Further it has been used for the API development and desktop application development in order to maintain the consistency.

5) MySQL: MySQL is an open source relational database management system. Information in a MySQL database is stored in the form of related tables. MySQL databases are typically used for web application development (often accessed using PHP). MySQL databases are queried using a subset of the standard Structured Query Language (SQL) commands [5]. For the development of database MySQL is used as the database schema was clearly known at the beginning of the project.

6) JSON: JavaScript Object Notation is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many others. These properties make JSON an ideal data-interchange language [6][7]. The familiarity with Java lead to select JSON format as the data interchange language in this project.

7) GSON: GSON is a Java library that can be used to convert Java Objects into their JSON representation. It can also be used to convert a JSON string to an equivalent Java object. GSON can work with arbitrary Java objects including pre-existing objects that you do not have source-code of [8].

8) REST: Representational state transfer (REST) is an abstraction of the architecture of the World Wide Web; more precisely, REST is an architectural style consisting of a coordinated set of architectural constraints applied to components, connectors, and data elements, within a distributed hypermedia system. REST ignores the details of component implementation and protocol syntax in order to focus on the roles of components, the constraints upon their interaction with other components, and their interpretation of significant data elements.

9) Tomcat Server: Apache Tomcat is an open source software implementation of the Java Servlet and Java Server Pages technologies. The Java Servlet and Java Server Pages specifications are developed under the Java Community Process.

C. GUI implementation

Graphical User Interface (GUI) is one of the key components in a software application that interact with human users. User friendly GUI is the one of the major nonfunctional requirement of this system. The main design considerations related with GUIs are listed below where these factors are considered in the system GUI designs.

- Attractive user interfaces (pleasant appearance)
- User friendly interfaces that are easy to use and easily learnable
- Easy Navigation and keeping the process flow of the actions
- Give helpful error messages and error messages
- Provide feedback of the actions whether succeeded or not

D. Implementation of Med-Lab Assistant API

The API is the heart of the project which provides services to both the lab users and report viewer app users. The API is web hosted along with the database such a way that it can provide services to both other components via internet.

The use of Spring gave the flexibility to work with many technologies. And also it gave the use of Spring MVC architecture, as well as use of dependency injection to the API implementation. Maven [12] was used as the project management/build tool and Eclipse was the IDE use for the development process.

First, a web application was created as a Maven project. The advantage of this is that the created web application gets the desired structure of a web application. Spring library was put into the library folder and then stated the required dependencies inside the pom.xml file. Finally the project was built using Maven and all the required dependencies were downloaded automatically from Maven repository. Tomcat server was used as the server to run the project. Java, JSON, GSON, REST, HTTP are the technologies used for the API development.

VII. TESTING

Testing of the system has been planned to be done at two phases, parallel testing and system and acceptance testing at the end of the development.

A. Development Testing

1) Unit testing: Individual program units/object classes are tested to verify the functionality of objects and the methods. Unit testing [16] was performed on each object focusing on three major aspects:

- Test all the methods associated with an object
- Set and check values of all attributes of an object
- Test an object for all possible states

JUnit [17] Testing was used for the development testing. Each class of all three components has been tested using JUnit

tests. Test classes were generated for each class. The constructor and all the method of each class has been tested by automating the unit testing process. Android JUnit testing was done for the classes in Android application.

2) Component testing: The component interfaces through which the components talk to each other are tested to ensure that the intercommunication between the components happen correctly. All kind of interfaces such as parameter interfaces, message passing interfaces, and shared memory interfaces had to be tested separately.

Part of this process is done by implementing test methods in unit test class. To test whether each of the components provides the intended functionality the other portion of component testing has been done manually.
Additionally testing on the synchronization of communication between components was performed.

3) System testing: Component interactions were tested. This was done ensure that system works erroneously when all the components work together. All the functionalities of the entire Med-App system were tested manually several times.

4) Development Testing [18]: Normally users test the system in their own environment. But in this project since there is no real user only user acceptance testing was performed. A feedback about the functionality as well as the user friendliness (learnability, appearance of GUIs, etc.) was taken.

VIII. Conclusion

Today, software industry has become an outstanding field in the world. There are various kinds of software which are capable to do simple tasks as well as complex tasks. In Sri Lanka the software industry is not much popular compared to developed countries. But, today many of the large and medium business companies are looking forward to move their manual systems to automated systems.
In this scenario the beginners for software developing can get lot of opportunities to enter the field by starting with smaller projects.

A. Further Enhancements

- Extend the application such that several types of medical reports are provided.
- Extend the application to view medical history of the patient
- Integrate the entire system to an existing medical lab management system
- Develop the mobile application for cross platform

References

[1] Google Inc., "Android Development" [Online]. Available: http://developer.android.com/develop/index.html

[2] Google Inc., "Send Messages from the Cloud" [Online]. Available: https://developers.google.com/cloud-messaging/

[3] Google Inc., "Cloud to device messaging" [Online]. Available: https://developers.google.com/android/c2dm/?csw=1

[4] Wikipedia contributors, "Google Cloud Messaging" [Online]. Available: https://en.wikipedia.org/wiki/Google_Cloud_Messaging. [Accessed: 24 Jun 2015].

[5] Wikipedia contributors, "MySQL" [Online]. Available: https://en.wikipedia.org/wiki/MySQL. [Accessed: 26 Jun 2015].

[6] "Introducing Json" [Online]. Available: http://json.org/

[7] Wikipedia contributors, "JSON" [Online]. Available: https://en.wikipedia.org/wiki/JSON. [Accessed: 16 Jul 2015].

[8] Google Inc., "Class GSON" [Online]. Available: https://google-gson.googlecode.com/svn/trunk/gson/docs/javadocs/com/google/gson/Gson.html

[9] Pivotal Software Inc., "Spring Framework" [Online]. Available: http://projects.spring.io/spring-framework/

[10] Pivotal Software Inc., "Spring Framework" [Online]. Available: https://spring.io/

[11] Wikipedia contributors, "Spring Framework" [Online]. Available: https://en.wikipedia.org/wiki/Spring_Framework. [Accessed: 06 Jul 2015].

[12] The Apache Software Foundation, "Apache Maven Project" [Online]. Available: https://maven.apache.org/

[13] Wikipedia contributors, "Mobile Computing" [Online]. Available: https://en.wikipedia.org/wiki/Mobile_Computing. [Accessed: 16 Jul 2015].

[14] .Wikipedia contributors, "Mobile Application Development" [Online]. Available: https://en.wikipedia.org/wiki/Mobile_application_development. [Accessed: 07 Jul 2015].

[15] Wikipedia contributors, "Cloud Computing" [Online]. Available: https://en.wikipedia.org/wiki/Cloud_computing. [Accessed: 14 Sep 2015].

[16] Wikipedia contributors, "Unit testing" [Online]. Available: https://en.wikipedia.org/wiki/Unit_testing. [Accessed: 17 Aug 2015].

[17] Lars Vogel, "Unit Testing with JUnit Tutorial" [Online]. Available: http://www.vogella.com/tutorials/JUnit/article.html

[18] Wikipedia contributors, "Development Testing" [Online]. Available: https://en.wikipedia.org/wiki/Development_testing. [Accessed: 7 May 2014].