The Development of Mobile Application for Manufacturing Operation – JF Engineering Mobile Information System

Mohd Azmi Bin Hussin
Faculty of Electrical Engineering
Universiti Teknologi MARA, Shah Alam (UiTM)
40450 Shah Alam, Selangor, Malaysia.
mohdazmihussin@gmail.com

***Abstract*—Nowadays, the manufacturing industry is facing several challenges to retain their competitiveness, particularly in producing a high quality products at a lower cost of operation. Engineering division as a part of the most important team in manufacturing is responsible for ensuring the production operation not be interrupted by any technical issues which can affects the company's profit margin or may results in losses. Thus, the idea to develop a mobile application that enhances efficiency and productivity to be used by Engineering team, brought to the development of Sony EMCS JF Engineering Mobile System. This mobile application was aimed to help reducing the response time and improve the approach of information sharing for production issues. The application was developed in an Android platform with Android Studio. It was based on a client-server architecture. The application was integrated with JF web portal and using Parse.com as its back-end for cloud database. It has many functionalities which related to JF Engineering's core activities such as Offline Defect Control, Start-Up Check Data, Touch-Up Data and Pallet Management. This mobile application also offers menu which enables users to send and view reports, email, calling, messaging, view profiles, and many others.**

Keywords—Mobile Application, Android, Manufacturing, Client-Server Architecture, Cloud

I. INTRODUCTION

The manufacturing industry is facing several challenges to retain their competitiveness, particularly in producing a high quality products at a lower cost of operation. Industry players in Malaysia are no exception, much less amid global economic uncertainty. The management of the industry and their employees, including engineers inevitably have to make improvements and innovations in their work activities, in order to ensure the survival of their livelihood's industry. The engineering division is one of the most important organization in manufacturing operation. They are responsible for ensuring that the production lines are not be interrupted by any technical failure or engineering problem which can affect production output volume. The inefficiency of the engineering team in handling manufacturing issues can causes losses or reduction of the company's profit margins. Due to this fact, an interactive, comprehensive and user-friendly mobile application was developed which can help the engineering team in manufacturing industry gains operational competitiveness, reduce costs and increase efficiencies.

This mobile application was developed in an Android platform by using Android Studio. Basically, the goal of this project is to facilitate JF Process Engineering Department operation in Sony EMCS (Malaysia) Sdn. Bhd. as they have to struggle to cope with the tough manufacturing environment of a new product by Sony, Jyushi Fuushi (JF)'s pre-E module for the thinnest ever 4K TV in the world.

This mobile application, namely JF Engineering Mobile Information System was designed based on the needs and the requirements by JF Process Engineering Department members. The functionalities which available in this mobile application includes their core activities in handling offline defect, start-up check, touch up and pallet management. Besides, the application also offers mobile tools for communication and easy access material for JF Engineering members' reference. This mobile application was first ever software developed in Android platform at Sony's manufacturing plant in Bangi. It should pave the way for more innovation and value-added activities among local engineers for industrial survival in Malaysia.

A. Android

Android is a wide-ranging Linux kernel-based operating environment. At the beginning of its deployment, Android was targeted for mobile phone arena. It includes smartphones and low-cost flip-phone devices. However, the full range of computing services and numerous functional support which available on Android have the maximum potential to be extended beyond the mobile phone market. Other platforms and applications, such as tablets can use the same technology [1].

Android applications are primarily written in the Java programming language and can include different kinds of resource files such as XML (Extensible Markup Language) in addition to Java. APK which stands for Android application package is a file format that generated after Java program and other related resources are compiled. Android operating system uses this package file format for distribution and installation of software and middleware [2].

II. METHODOLOGY

Agile methodology was selected as the appropriate approach in developing JF Engineering Mobile Information System application. It is the latest software development methodology to come over to the need for continuous change and improvement in a software project. This methodology helps the developers react to unpredictability through incremental, iterative work paces, known as sprints [7].

This methodology appears to have its root in manufacturing domains as the software developer team's capability in these sectors need to efficiently and effectively respond to the changing of user requirement during life cycle's project [8].

As this project implemented Agile model for the development of JF Engineering Mobile Information System application, the following phases are taken place:

- Requirements specification: A comprehensive study of the behaviour and features of the system to be developed. The requirements for the project development is identified.
- Design: The system's architecture is defined and the important components of the system is identified.
- Implementation: The system design is implemented into source code.
- Evaluation and Testing: The system source code is verified whether has met its specification, the success of the previous phases is evaluated and modifications is made, if necessary.

A. Survey

The first phase of the development of this project is requirements and specifications. During this phase, all requirements to the intended mobile application were documented. A sufficient time was spent during this phase in conducting extensive interviews with JF Process Engineering Department management and staffs. All received data is structured and analyzed. The technical limitations that may arise were considered and come out with a ready-to-follow specification to meet the department operation's needs.

All the survey respondents agreed the mobile application which to be developed need to provide functionalities to handle process issues in production lines such as offline defect control, start-up check activity, edge coat touch-up issue and pallet management. They also expected this new mobile application can replaces WhatsApp as a communication tool which extensively used now during operation hour. The results from this survey brought to the specification of output for the mobile application.

Output can be defined as data generated by a system and software application. It includes data produced at a software level such as the calculation result and user interface. The following are the functional requirements that were developed in this project:

- Login and Account Registration Screen
- Offline Defect Screen
- Start Up Check Screen

B. Mobile Application in Manufacturing Industry

According to ABI Research by 2016, manufacturing industry will generate approximately 23% of the nearly US$5 billion in mobile enterprise application worldwide. Mobile enterprise applications, also called mobile B2E (Business to Employee) applications, include dashboard applications, work flow approval applications, and line-of-business applications for both the smartphone and tablet. Based on ABI Research, even though healthcare is one of the most dynamic sectors for mobile technologies, manufacturing is still the largest sector for mobile enterprise applications worldwide. Manufacturing industry is also the second-largest employer worldwide, and workers in many manufacturing roles are expected to use mobile applications, including delivery drivers, management and supervisory personnel, sales, installation and repair workers, and shipping/receiving workers, according to the study [3].

Today, the developers create mobile application in Android platform for work program that allow users to get work done from wherever they are, to increase the productivity including in manufacturing industry. There were also many studies and researches made in applying Android in this industry. Andrei Drumea studied the control of industrial systems using Android-based devices, and also presented some tests and results of controlling of an industrial system – hydraulic test bench. Based from the test results and analysis, the study concluded that Android devices can be used for controlling simple industrial systems, as USB latency for Android devices makes it impossible to use them directly to control analog to digital and digital to analog converters if sample rates smaller than seconds are required [4].

Mircea Murar and Stelian Brad studied the monitoring and controlling of smart equipments using Android compatible devices towards IoT (Internet of Things) applications and services in manufacturing industry. The researchers developed an architecture for IoT for implementation in manufacturing environment. They implemented and tested the selected solution for connectivity between a smart equipment and Android compatible devices. The study concluded that the solution between a manufacturing resource and monitoring device were capable to meet it needs [5].

Checking the attendance of employee is one of the importance issues which related to manufacturing productivity. In order to overcome this issue, Puchong Subpratatsavee and Tanapas Kongtrakul studied and created a system that makes easier to check employee's coming 'automatichecky'. The system is implemented in Thailand Manufacturing near SiRacha province. The system is based on 1D barcode technology and run on mobile application with Android OS. The study showed how a system relying on 1D-barcode technologies can be developed. The developed system is simple and expandable. The barcode that have been employed for this specific system are barcode, and the experiment used has shown stable and reliable results. The algorithm has secured important data that user can keep on this code. These barcode can be implemented in the manufacturing industry and able to replace old employee ID cards [6].

- Touch Up Screen
- Pallet Management Screen

Besides these output specifications, the mobile application will also include other functionalities as follows:

- Staff Directory
- Calendar
- Announcement
- Report
- Download
- Link to related portals

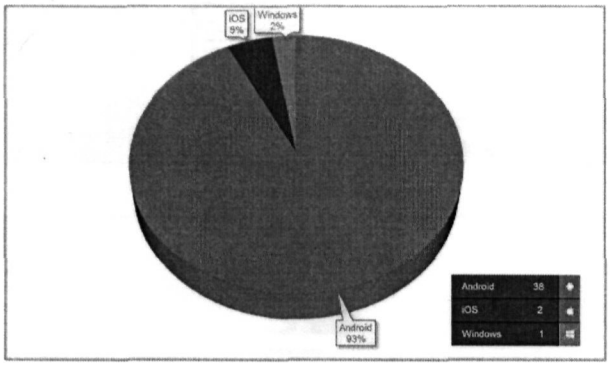

Fig. 1. Number of smartphone OS used in JF Process Engineering Department (Data collected ending on October 1, 2015)

The survey also has been conducted to find usage percentages of Android smartphone and other smartphones among staffs in JF Process Engineering Department. The survey shows that 93% from staffs use Android smartphone, 5% for iOS and 2% for Windows. Figure 1 shows the pie chart of the number of smartphone OS used in JF Process Engineering Department. As there are a number of staffs who use other operating system (OS) smartphones (iOS and Windows Phone), this project need to consider developing a new portal with a mobile view as an integrated system for JF Engineering Mobile Information System. The JF Engineering Mobile Information System basically will be developed in the Android platform which will be integrated with the portal.

From the same survey, it was found that the lowest Android version used by JF Process Engineering Department members was the Android version 4.0.x Ice Cream Sandwich. The clustered bar chart in Figure 2 shows the number of Android versions used in JF Process Engineering Department. Thus, the application then, was developed with the minimum SDK (Software Development Kit) version API (Application Programming Interface) 14 (Ice Cream Sandwich) and the target SDK version API 22 (Lollipop) to ensure the intended app is compatible with all the department members' smartphone.

Minimum SDK API 14 refers to the minimum version of an Android platform on which the application will only run on smartphone with API level 14 (Ice Cream Sandwich) or higher. Target SDK API 17 refers to the version of an Android platform is designed to run for API level 22 (Lollipop).

Android API (Application Programming Interface) libraries is the core of the SDK. It provides developer access to the Android stack, which are the same libraries used at Google to develop native Android applications [9].

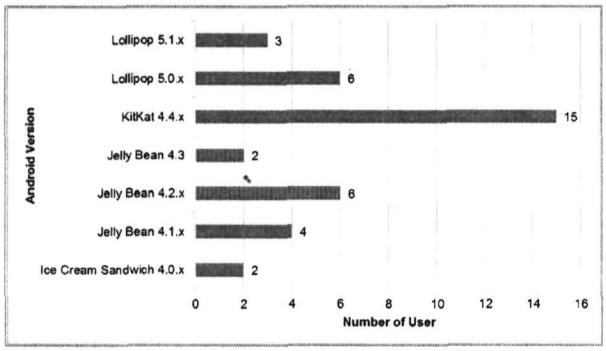

Fig. 2. Number of Android version used in JF Process Engineering Department (Data collected ending on October 1, 2015)

B. Environment Setup

The development of this mobile application were done in Windows 10 Pro operating system. Several softwares and development environments were installed and setup as follows:

- Java Development Kit (JDK) – lays the foundation for the Android SDK.
- Android Studio – an easy to use Integrated Development Environment (IDE) that brings together Java and the Android SDK to make it simple to write Android application.
- Android SDK – included in Android Studio, which provides access to Android libraries and allows development for Android.
- Parse SDK – Some of its important functionalities include communicating over the network, persisting data to disk, and returning data to the application developer so that they can update their user interface.

Before Android Studio can be installed, the system must meet the minimum requirement by having Java Development Kit (JDK) version 7 or newer. As Android SDK and Android applications were developed using Java programming language, JDK should be the first component installed in Android development environment. Therefore, JDK version 7 need to be downloaded and installed. Once the JDK was completely installed and configured, Android Studio can be installed to the system. Android Studio was downloaded from URL: http://developer.android.com/sdk/index.html.

The installation of Android Studio has included the Android Studio IDE and the Android SDK tools in its bundle. By default, the Android SDK does not include all the need to start the application development; hence the packages of SDK separates tools, platforms, and other components need to be download using the Android SDK Manager as shown in Figure 3.

As this mobile application is a client-server architecture's project as shown in Figure 4, its development extensively used the free features offered by Parse.com. Parse.com is Backend as

a Service (BaaS) provider. It offers free plan for web and mobile application developers on any platform.

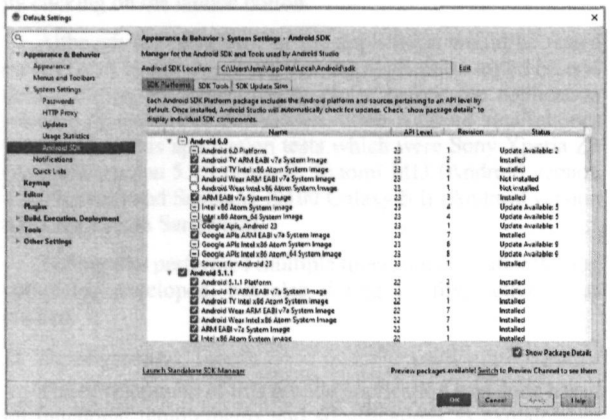

Fig. 3. Android SDK Manager

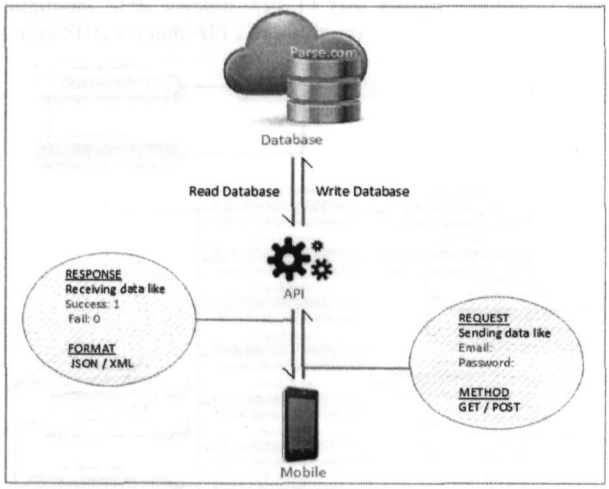

Fig. 4. Android client-server connects to database in Parse.com

Parse.com provides SDKs for devices running iOS, Android, Windows (Phone) 8, OS X and JavaScript. It handles everything needed to store data in the cloud. Basic data types, locations, photos, and many more can be saved and queried using Parse's database and web based data browser [10].

At the beginning, an account on Parse.com was created so that this mobile application can utilize it. The SDK provided by Parse.com was downloaded and installed to the project. Figure 5 shows the location where Parse SDK.jar was added.

The lines as shown in Listing 1 were added to the build.gradle in the app project.

```
dependencies {
    compile 'com.parse.bolts:bolts-android:1.+'
    compile 'com.parse:parse-android:1.+'
}
```

Listing 1. Compiling Parse SDK in build.gradle

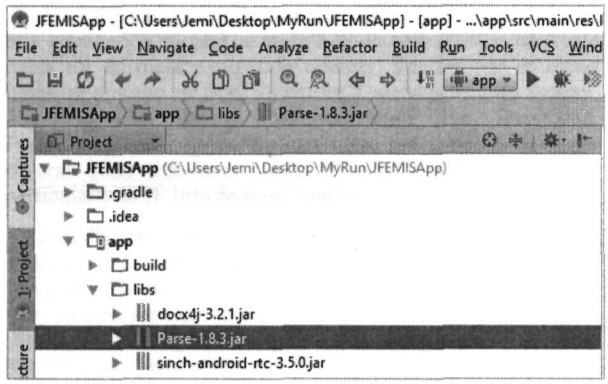

Fig. 5. Parse SDK .jar file was added to libs folder

To connect the app to Parse.com, it must request the INTERNET and ACCESS_NETWORK_STATE permissions. The following lines as shown in Listing 2 were added inside the <manifest> tag in the AndroidManifest.xml:

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission
  android:name="android.permission.ACCESS_NETWORK_STATE" />
```

Listing 2. Connecting Parse.com in AndroidManifest.xml

The lines as shown in Listing 3 were written to a Java class, Application.java to enable local datastore.

```
Parse.enableLocalDatastore(this);
Parse.initialize(this,"APP_ID","CLIENT_KEY");
```

Listing 3. Enable Local Datastore in Application.java

Besides Parse SDK, there were several other libraries added to this project such as sinch-android-rtc, docx4j and mpandroidchartlibrary.

C. Virtual and Real Devices for Testing Setup

In developing an Android app, the developer need to compile and run the app many times. An Android app can be tested by installing and running it either on a real device or in an Android Virtual Device (AVD) emulator environment. For this app project, Android virtual devices of Nexus 6 API 22 (Android version 5.1 Lollipop) and Nexus One API 14 (Android version 4.0 Ice Cream Sandwich) were used. These two virtual devices were selected to match with Android version specification which discussed earlier. Nexus One API 14 was selected as the minimum version which compatible with this app, while Nexus 6 API 22 as the target version. The Android version 5.1.1 Lollipop was the latest Android version during the initial development of this app. The CPU/API (Central Processing Unit/Application Binary Interface) for these virtual devices were configured to ARM system image. The ABI defines how an app's machine code is supposed to interact with the system at runtime. The ARM system image was chosen because the majority of Android real devices in the market currently run on ARM architecture.

The application was tested on the AVD by selecting emulator from the Android Virtual Device Manager, followed by clicking on the launch button.

Although the behaviour of the application would be tested on AVDs, it is still important that the application tested on real device during the development phase before the application released to users. For this reason, three Android smartphones were used in this application tests which were Sony Xperia Z3 (Android version 5.1 Lollipop), Xiaomi MI3 (Android version 4.4.4 KitKat) and Samsung I9100 Galaxy S II (Android version 4.0.1 Ice Cream Sandwich).

Testing was performed multiple times, not only after the app completed developed, but also during its first activity was created.

D. Development

The development of this mobile application was done based on functional requirements and specifications as explained in Section A. Hence, the application was developed with the minimum SDK version API 14 (Ice Cream Sandwich) and target SDK version API 22 (Lollipop).

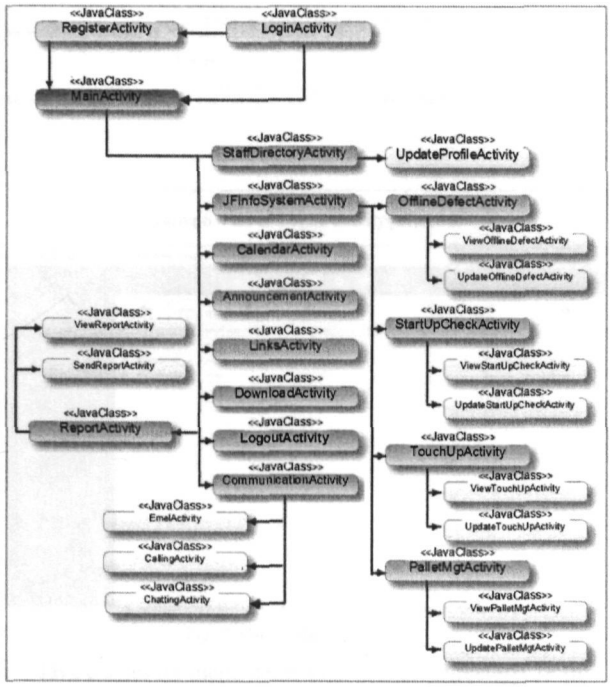

Fig. 6. Simplified class diagram for JF Engineering Mobile System's classes

Simplified class diagram in Figure 6 shows the main classes which developed for this application.

The navigation drawer was implemented as a backbone design for this application. By using the navigation drawer, the user for this application may see all the menu items which are available with just one swipe. When a user selects the navigation drawer icon in the action bar, a menu list of options appears to slide out from the left edge of the screen. However, this menu list is hidden most of the time. [11]

After the navigation drawer was added for MainActivity in this application project, a source code was written to create items for its main navigation options. The options or menu items for this app are Staff Directory, Calendar, JF Info System, Announcement, Report, Communication Hub, Download, Portal Link and Logout. Besides these menu items, there are several other sub-items were created in this application, particularly in JF Info System's menu item.

Initially, Android Studio automatically created two Java classes which were MainActivity.java and NavigationDrawerFragment.java. Thus, new Java classes were then created for each menu items and sub-items. The lines as shown in Listing 4 are part of a source code for this MainActivity.java:

```
@Override
public void onNavigationDrawerItemSelected(int position) {
        FragmentManager fragmentManager = getSupportFragmentManager();

    if (position==0){
        fragmentManager.beginTransaction()
                .replace(com.soemkl.azmi.jfemisapp.R.id.container,
fragment1.newInstance())
                .commit();

    }else if (position==1){
        fragmentManager.beginTransaction()
                .replace(com.soemkl.azmi.jfemisapp.R.id.container,
fragment2.newInstance())
                .commit();
    }else if (position==2){
        fragmentManager.beginTransaction()
                .replace(com.soemkl.azmi.jfemisapp.R.id.container,
fragment3.newInstance())
                .commit();
    }else if (position==3){
        fragmentManager.beginTransaction()
                .replace(com.soemkl.azmi.jfemisapp.R.id.container,
JFInfo.newInstance())
                .commit();
    }else if (position==4) {
        fragmentManager.beginTransaction()
                .replace(com.soemkl.azmi.jfemisapp.R.id.container,
```

Listing 4. Part of MainActivitiy.java code

LoginActivity class was created as a launch screen for this application by implement coding in AndroidManifest.xml file as shown in Listing 5.

The implementation of CATEGORY_LAUNCHER for this class means the LoginActivity appears in the Launcher as a top-level application.

```
<activity
    android:name="com.soemkl.azmi.jfemisapp.LoginActivity"
    android:label="@string/app_name" >
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

Listing 5. Implementation of launcher in AndroidManifest.xml

The functionality of Login for this application used free cloud service provided by Parse.com and developed with Parse SDK. Listing 6 shows the implementation of the Java code in LoginActivity class. Username, password and password for the login and registration activities were saved in a cloud database in Parse.com account which already registered by the developer earlier. Figure 7 shows user table's database in Parse.com which were manipulated for this application's login functionality.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(com.soemkl.azmi.jfemisapp.R.layout.loginactivity_layout);

    DeclareVariable();


    loginbutton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {

                String userinput = username.getText().toString();
                String passwordinput = password.getText().toString();

                ParseUser.logInInBackground(userinput, passwordinput, new
LogInCallback() {
                    @Override
                    public void done(ParseUser user, ParseException e) {

                        if (user != null) {

                            ParseInstallation installation =
ParseInstallation.getCurrentInstallation();

installation.put("user",ParseUser.getCurrentUser().getUsername().toString());

installation.put("userObjectId",ParseUser.getCurrentUser().getObjectId().toString()
);
                            installation.saveInBackground();

                            Intent intent = new Intent(LoginActivity.this,
MainActivity.class);
                            startActivity(intent);



                            final ProgressDialog dlg = new
ProgressDialog(LoginActivity.this);
                            dlg.setTitle("Please wait");
                            dlg.setMessage("Log in. Please wait");
                            dlg.show();

                            Toast.makeText(getApplicationContext(), "You have
successfully logged in.", Toast.LENGTH_LONG).show();

                        } else {

                            Toast.makeText(getApplicationContext(), "Login error:
If you have not registered yet, please register.", Toast.LENGTH_LONG).show();
                        }
                    }
                });
        }
    });
```

Listing 6. Part of LoginActivity.java code

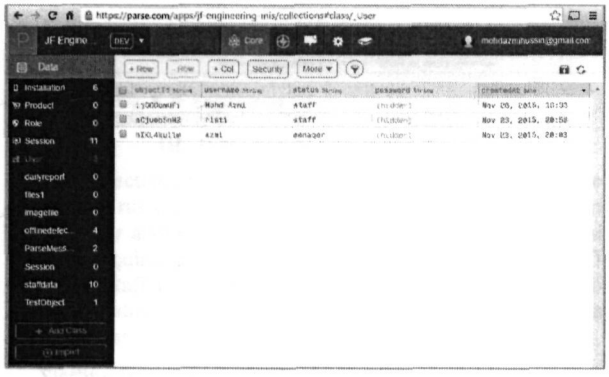

Fig. 7. User table in Parse.com

Java classes for menu items such as Staff Directory, JF Info System, Calendar, Communication Hub, etc. were implemented, followed by activities in sub-menu items.

The example of a Java class implementation of a sub-menu item for UpdateOfflineDefect activity is shown in Listing 7. Similar to many activities in this application, the functionality of update offline defect also stores a database in Parse.com. It same goes to ViewOffineDefect activity which retrieves data stored by users in Parse.com.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(com.soemkl.azmi.jfemisapp.R.layout.updateofflinedefect_layout);

    declareedittext();

    save.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {

            line = lineid.getText().toString();
            date = dateid.getText().toString();
            shift = shiftid.getText().toString();
            defect = defectid.getText().toString();
            action = actionid.getText().toString();
            pic = picid.getText().toString();
            remark = remarkid.getText().toString();


            ParseObject offlinedefectdata = new ParseObject("offlinedefectdata");
            offlinedefectdata.put("LINE", line);
            offlinedefectdata.put("DATE", date);
            offlinedefectdata.put("SHIFT", shift);
            offlinedefectdata.put("DEFECT", defect);
            offlinedefectdata.put("ACTION", action);
            offlinedefectdata.put("PIC", pic);
            offlinedefectdata.put("REMARK", remark);
            offlinedefectdata.put("IMAGE", fp);
            offlinedefectdata.saveInBackground();

            Toast.makeText(getApplicationContext() ,"Successfully Save Your Data"
,Toast.LENGTH_LONG).show();

            Intent k = new Intent(UpdateOfflineDefect.this, OfflineDefect.class);
            startActivity(k);
        }
    });
```

Listing 7. Part of UpdateOfflineDefectActivity.java code

E. User Interface Design

One of the objectives of this project was to create friendly-user Graphic User Interface (GUI) that can be used easier and faster by users. Basically, a hierarchy of View and ViewGroup objects are used to build a graphic user interface for an Android application. Buttons or text fields are examples of UI widgets in View objects. ViewGroup objects are invisible view containers which define the way the child views are laid out, such as in a grid or a vertical list. XML vocabulary which provided by Android that corresponds to the subclasses of View and ViewGroup can define user interface in XML using a hierarchy of user interface elements [12].

Layouts are subclasses of the ViewGroup. This application mostly implemented LinearLayout in its activities' layouts. The lines as shown in Listing 8 are the extract of the layout codes which were implemented to design the screen.

```
<?xml version="1.0" encoding="utf-8"?>

<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
android:layout_height="match_parent"
    android:orientation="vertical"
    android:background="#ffffffff">

    <Button
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="ENTER STAFF DIRECTORY"
        android:layout_marginTop="150dp"
        android:textColor="#ffffffff"
        android:id="@+id/staffdirectory"
        android:background="#0d47a1" />

</LinearLayout>
```

Listing 8. Extract of Layout .xml file

User interfaces in this project were developed to meet the requirements specifications by users. The components provided by Android eased the design process as many of them need only simple defining of content.

F. JF Engineering Information Portal

As an integrated web to this application, JF Engineering Information portal was developed. It is a simple portal as a mirror to the mobile application. This portal was developed with Drupal and linked to Parse.com, and using PHP SDK. Although the portal allowed to access to the internet by company system administrator, for the security reason, it is still an intranet portal. The portal as shown in Figure 8, can only be accessed using company network. The functionalities in this portal almost similar with JF Info System menu items in the mobile application.

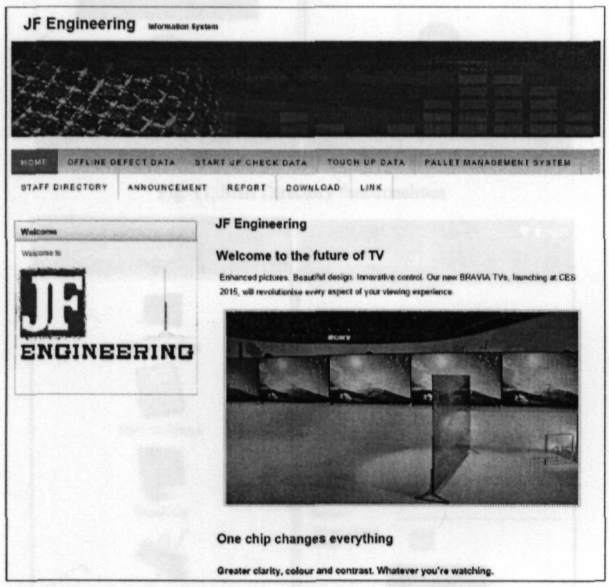

Fig. 8. JF Engineering Information System portal

III. DISCUSSION AND RESULT

The objectives in developing this mobile application were done. It can run on a minimum Android version 4.0.x Ice Cream Sandwich or above. The functionalities which were developed meet the requirements specifications such as it can view a staff profile in Staff Directory, update the offline defect in JF Info System, sending messages among members and downloading a document for technical reference.

Starting from the application's first launch, the login screen as shown in Figure 9 appears. The registered users then, enter the username and password, and the application brings it to a 'Home' screen as shown in Figure 10. Otherwise, if the user is not register yet, the registration can be made by pressing the 'register' button.

In a 'Home' screen, users can view the main navigation option by one time swiping to menu item on the left edge of the screen. From this menu item, users can select the functionalities which they want to use.

For example, if the users want to view Staff Directory in searching for other member's telephone number, they can go to Staff Directory and the screen will appear as shown in Figure 11.

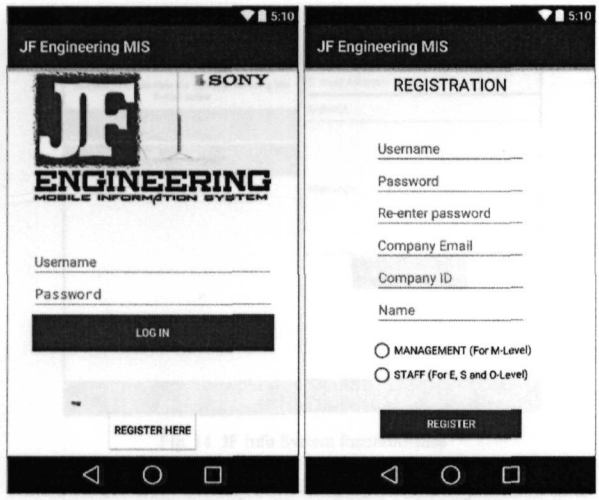

Fig. 9. Login and Registration functionalities

Fig. 10. Home screen and main menu item

In using the functionalities for monitoring and updating processes in production lines, the user just need to press on JF Info System and the application will show the screen as shown in Figure 12. They can now update the Offline Defect in production line by clicking Offline Defect Update in sub-menu Offline Defect. The Start-Up Check and Touch-Up also can be updated using this application as shown in Figure 13.

With another functionality in Report item, the management can view Defect Report and Daily Report which uploaded by other members through their smartphone as shown in Figure 14. This application also provides communication hub, and e-mail is one of its functionalities, besides calling and chatting.

All the functionalities of this application were tested and evaluated using real devices and emulator as explained in Section II (C). The results showed that they passed the expected criteria, although there are lots of improvement can be made.

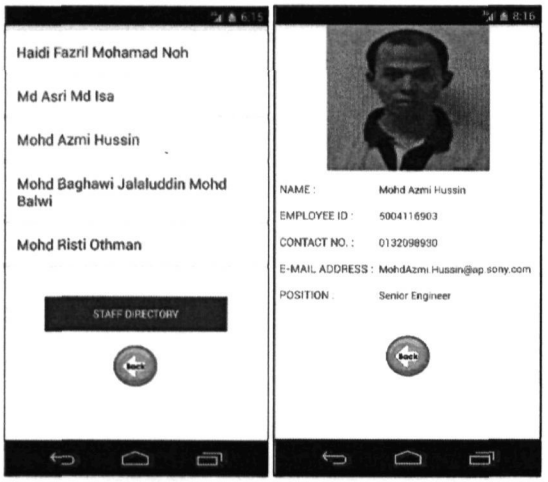

Fig. 11. Staff Directory functionalities

Fig. 12. JF Info System functionalities

Fig. 13. JF Info System functionalities

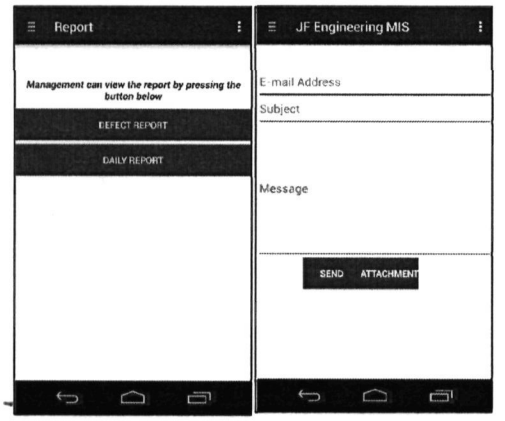

Fig. 14. JF Info System functionalities

IV. CONCLUSION

JF Engineering Information System which running on an Android platform is capable of helping and supporting JF Process Engineering's operations with multiple functionalities in monitoring production lines' processes. It also can be an alternative communication tool for information sharing among the department's members. For the future enhancement, there are many aspects of improvement need to be done, which related to functional and non-functional requirements. Among the improvement are real-time webcam process monitoring, real-time auto-generate graph for start-up check activities, and staff attendance monitoring. Android platform can be used for manufacturing tools and should be utilized to improve the productivity and efficiency in the industry.

REFERENCES

[1] Cohen, Ryan, and Wang, Tao, Android Application Development for the Intel® Platform, Apress Open, 2014. (*references*)

[2] Burton, Michael, Android™ Application Development For Dummies®, 3rd Edition, John Wiley & Sons, Inc., New Jersey, 2015.

[3] Enterprise Mobility Market Data [Online]. Available: https://www.abiresearch.com/market-research/product/1001623-business-mobility-market-data

[4] Drumea, A., "Control of Industrial Systems Using Android-based Devices," in Electronics Technology (ISSE), 2013 36th International Spring Seminar, pp.405-408, 8-12 May 2013.

[5] Murar, M.; Brad, S., "Monitoring and Controlling of smart equipments using Android compatible devices towards IoT applications and services in manufacturing industry," in Automation, Quality and Testing, Robotics, 2014 IEEE International Conference, pp.1-5, 22-24 May 2014.

[6] P. Subpratatsavee, T. Kongtrakul, "Employee Attendance Checker in Manufacturing Using Barcode and Mobile with Embedded Camera", in Applied Mechanics and Materials, Vol. 752-753, Apr. 2015.

[7] Buragga, K. A., & Zaman, N., Software Development Techniques for Constructive Information Systems Design, Hershey, PA, 2013, pp.114.

[8] Dingsøyr T., "A Decade of Agile Methodologies: Towards Explaining Agile Software Development," in The Journal of Systems and Software. 2012-06-01;85:1213-1221.

[9] Meier, Reto, Professional Android™ Application Development, Wiley Publishing, Inc., Indiana, 2009.

[10] C. Delessio, L. Darcey, and S. Conder, Sams Teach Yourself Android™ Application Development in 24 Hours, Sams, Indiana, 2015.

[11] Building a Simple User Interface [Online]. Available: http://developer.android.com/training/basics/firstapp/building-ui.html

[12] Parse Tutorials [Online]. Available: https://parse.com/tutorial