Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X
IMPACT FACTOR: 6.017

IJCSMC, Vol. 7, Issue. 2, February 2018, pg.56 – 72

CROSS PLATFORM MOBILE PHONE APPLICATION USING J-QUERY MOBILE

(A Case Study of a Simple Mobile Chat Application)

[1]Onukem Chijioke, [2]Dr. Oye, N. D.

[1]Department of Computer Science College of Physical and Applied Science,
Micheal Okpara University of Agriculture Umudike, Nigeria, +2348063567797
[2]Department of Computer Science MAUTECH- Yola Nigeria, oyenath@yahoo.co.uk, +2347037460352

Abstract: With today's ubiquity of smart phones, tablets and other mobile devices, more and more businesses develop and use mobile applications. Multiple platforms including Android and iOS form the market of mobile devices, so it can be important to be able to deliver software for more than one platform. Vendor-supported SDKs are feature-rich but incompatible with other platforms. We compared a number of cross-platform frameworks for mostly platform-independent development on mobile platforms, by evaluating them in several categories and weighing them against native SDKs. We found out that cross-platform solutions can be recommended in general, but they are still limited if high requirements apply regarding performance, usability or native user experience. The increasing demand for mobile applications in many areas presents developers and companies with several problems: implementing an application for multiple mobile Platforms, like Android and iOS, may require high effort in terms of development time, resources, maintenance and possibly licenses, tools and deployment. One influencing factor is the fragmentation of the market of mobile platforms, i.e. smart phones and tablets. Android and iOS lead the sales with a high market share, while other platforms are less prevalent but might become more popular in the future.

Keywords: Cross Platform; Mobile Phone; Android platform

Introduction

With today's ubiquity of smart phones, tablets and other mobile devices, more and more businesses develop and use mobile applications. Multiple platforms including Android and iOS form the market of mobile devices, so it can be important to be able to deliver software for more than one platform. Vendor-supported SDKs are feature-rich but incompatible with other platforms. We compared a number of cross-platform frameworks for mostly platform-independent development on mobile platforms, by evaluating them in several categories and weighing them against native SDKs. We found out that cross-platform solutions can be recommended in general, but they are still limited if

high requirements apply regarding performance, usability or native user experience. The increasing demand for mobile applications in many areas presents developers and companies with several problems: implementing an application for multiple mobile Platforms, like Android and iOS, may require high effort in terms of development time, resources, maintenance and possibly licenses, tools and deployment (Cory, 2013). One influencing factor is the fragmentation of the market of mobile platforms, i.e. smart phones and tablets. Android and iOS lead the sales with a high market share, while other platforms are less prevalent but might become more popular in the future (example, BlackBerry 10).

Statement of the Problem

Mobile learning management systems are very important for training purpose. But considering the present scenario, the learners are equipped with a number of mobile devices that run by different operating systems with diverse device features. Therefore, developing a mobile application for different platforms is a cumbersome task if appropriate tools and techniques are not practiced. Personal interviews with people, student, workers, and so on, together with observations and careful examinations revealed that native applications pose some problems which include.

No Portability

Since each native application only runs on one platform, businesses building native apps must make a choice--build for one platform or build for multiple platforms? Unfortunately, there's no easy answer. The mobile platform landscape includes 4 major smartphone platforms and 4 major tablet platforms. Building an app for just one platform excludes 7 platforms, yet building for all platforms requires significant time and resources.

Platform Instability

The mobile platform landscape is notoriously unstable. A popular platform today may disappear in just a few years. For example, both Blackberry and Palm dominated the mobile industry just 5 short years ago. Today, Blackberry is struggling and Palm doesn't exist. The fact is, nobody knows what the mobile platform landscape will look like in another 5 years. Companies that choose the native approach always run the risk of wasting time and money building for a platform that might not last.

Development Cost

While native app development cost varies depending on the app's complexity, it's easily the most expensive and time-consuming approach. For example, Forrester Research estimates that most native apps require at least 6 months of full-time work, and cost between $20,000 and $150,000, depending on complexity. It's important to note that those estimates apply to single-platform native app development. The cost rises exponentially when developing cross-platform native applications, as every platform requires a separate application built with a different programming language.

Development Time

As mentioned above, Forrester Research estimates that a single native app requires 6 months of development time. If building native apps for more than one platform, the time requirements rise depending on the number of developers needed and application complexity. For example, using just one developer for cross-platform smartphone app development brings the development time up to 2 years (4 apps x 6 months each). However, development time estimates become increasingly complex when using multiple developers. For instance, if a business uses four different developers for cross-platform smartphone app development, they will receive four different app designs. As any project manager knows, ensuring that multiple apps created by multiple developers look and function identically is a very time-consuming task.

Maintenance Cost

While all apps require regular updates and maintenance, native apps require the most future maintenance when compared with the other two mobile app options. Beyond regular app maintenance, native apps must also be updated with every new platform release. Additionally, businesses that build native apps for multiple platforms must maintain multiple applications, duplicating every change or update across all applications. For instance, a minor change to a cross platform smartphone and tablet application requires changes to eight separate applications.

Limited Control

When placed in an app store, a native application is completely controlled by the app store's owner (like Apple or Google). For instance, if Apple rejects or bans a company's app from their app store, the company has no recourse. If Apple decides an app doesn't meet their terms of service, the app is removed. If another company claims copyright over an element in the app, the app is removed. Or, if Apple decides the app isn't right for their store, the app is removed. The app store model puts companies at the mercy of a third party. All of the resources put into their application are wasted if that app store's owner decides the app isn't right for their store.

Objectives of the Study

This research work is aimed at developing a single mobile application that can be comfortably deployed to several mobile platforms like Windows, Android, Java, and IOS, with little or no modification of the original application, using J Query Mobile, HTML, CSS, JavaScript, Ajax, PHP, and MySQL Database. Its major objectives include:

a) Ensuring the use of a single application on several mobile operating systems. i.e. Android, Windows, Java, etc.

b) Enabling deployment of an application to different mobile platforms with little or no modification

c) Providing an efficient information sharing platform through text messaging

d) Eliminating Stress encountered by programmers in developing mobile applications for several platforms. Now the programmer needs not to write separate code for different mobile operating system because a single program can be deployed to several mobile platforms without the need to re-write the entire code

e) Provides uniformity in mobile applications of different platforms. Since the program needs not to be re-written for deployment to another operating system, it maintains the same GUI and functionality in all platforms in which it is been deployed to. This means that a single program can exist in windows and android platform without any discrepancy in their GUI or functions / modules.

f) Promote the Hybrid Approach for mobile application development

Major Research Questions

a) Who are the target users of the new system?
b) What technology is behind cross-platform application development?
c) How can users exchange information in form of text via the mobile application?

Significance of the Study

This project is geared towards developing mobile application which can go some extreme providing solutions to the above aforementioned problems and to create awareness on the usefulness of mobile devices via internet browsing and its advantages over its contemporary desktop browsing. Taking into consideration the mobile, less cost security of mobile web application. You conclude that mobile devices have an edge over its contemporary. It can be used at any time and in any place. This explains why businesses are explaining the methods of keeping their employees mobile while increasing profitability. While customers may want to use mobile devices to buy tickets to sporting events, theaters and so on.

Scope and Limitation

In as much as there are many mobile devices that exist today, this research is limited to only smart phones with android, iOS and Windows operating system as a case study. The intension of the author was to investigate and research more than what he has done, but some factors which the author could not conquer were indeed contrary to the author's wishers and expectations.

Literature Review

The mobile development community is at a tipping point. Mobile users demand more choice, more opportunities to customize their phones, and more functionality. Mobile operators want to provide value-added content to their subscribers in a manageable and lucrative way. Mobile developers want the freedom to develop the powerful mobile applications users demand with minimal roadblocks to success. Finally, handset manufacturers want a stable, secure, and affordable platform to power their devices. Up until now single mobile platform has adequately addressed the needs of all the parties.

Benjamin (2008) claims that there are three different types of cell phone users:

- The normal user who uses only the basic applications provided by the cell phone.
- The advanced user who uses a large part of the provided applications.
- The expert user who tries to get deeper into the cell phone environment develops applications and uses the total band of functionalities provided by the cell phone.

A Brief History of Mobile Software Development

Remember way back when a phone was just a phone? When we relied on fixed landlines? When we ran for the phone instead of pulling it out of our pocket? When we lost our friends at a crowded ballgame and waited around for hours hoping to reunite? When we forgot the grocery list and had to find a payphone or drive back home again? Those days are long gone. Today, commonplace problems like these are easily solved with a one-button speed dial or a simple text message like "WRU?" or "20?" or "Milk and?" (Darcey & Conder, 2012)

Darcey & Conder (2012) claims that if you go back to the history of the mobile applications, then you can clearly figure out that a few Java games, a calculator or monthly calendar were all that came under the category of mobile apps. However, the first smart phone was announced for the general use by IBM in 1993 that was equipped with the features like calculator, world clock, calendar and contact book. The BlackBerry Smartphone released in 2002 was the next major achievement in the field of mobile application development and it was marked by BlackBerry Limited, formerly known as Research in Motion Limited (RIM) and integrated with the innovative concept of wireless email.

"The Initial Initiative"

The Motorola DynaTAC 8000X was the first commercially available cell phone. First marketed in 1983, it was 13 x 1.75 x 3.5 inches in dimension, weighed about 2.5 pounds, and allowed you to talk for a little more than half an hour. It retailed for $3,995, plus hefty monthly service fees and per-minute charges.

According to Darcey & Conder (2012), early mobile phones were not particularly full featured. (Although, even the Motorola DynaTAC, shown in Figure 2.1, had many of the buttons we've come to know well, such as the SEND, END, and CLR buttons.) These early phones did little more than make and receive calls and, if you were lucky, there was a simple contacts application that wasn't impossible to use.

Fig. 1. The first mobile phone

What is a mobile system?

A mobile system is a computer system which isn't linked to a certain place. It is possible to move it or carry it around like e.g. a cell phone, a handheld or a special computer system in a car. Although there are many similarities between a stationary and a mobile operating system, there are also clear distinctions concerning mobility (Benjamin, 2008). An example for an application where a normal operating system is not able to be used is ABS control in a car. An operating system like Windows XP which is not stable enough to guarantee the running of the ABS system over a long time, cannot be used. This example points out which attributes are important for a mobile system of any device: The system must be stable and fail-proof.

What Is A Mobile Application?

A mobile application or mobile app is a software application. Basically, it is a computer generated program designed and developed to run on iPhone, Smartphones, tablets and many other mobile devices. Without doubt, the world has accepted the fact that apps are the way to get the most of the smartphones. So, it will be riveting to trace the history markups for the mobile apps development. According to Mercy Rop (2017), when developing a mobile app, it is important to consider the features and constraints of mobile devices like a tablet or smartphone. Mobile devices run on a battery, and their processors are less powerful than the personal computers as well as more features like cameras and location detection.

Mobile Platforms and operating system

To be able to develop mobile applications, we need to first understand the underlying principles of mobile platforms and how they are related to mobile applications. Mobile platforms are basically those that allow software and services to be run on devices (Fling 2009). Examples of mobile platforms include Palm, BlackBerry, iPhone, Android and Windows Mobile. Mobile operating systems provide tools that allow application to share data and services. Examples of mobiles OS includes Palm OS, Symbian, Windows Mobile, Mac OS X and Android. (Note: For simplicity, 'mobile platforms' and 'mobile operating system' will be used interchangeably in the rest of this chapter). For the scope of this study, we shall be considering three major mobile platforms, which are:

- Android Platform
- iOS Platform
- Windows Phone Platform

Android Platform

Android is a mobile operating system developed by Google, based on a modified version of the Linux kernel and other open source software and designed primarily for touchscreen mobile devices such as smartphones and tablets. In addition, Google has further developed Android TV for televisions, Android Auto for cars and Android Wear for wrist watches, each with a specialized user interface. Variants of Android are also used on game consoles, digital cameras, PCs and other electronics. Initially developed by Android Inc., which Google bought in 2005, Android was unveiled in 2007, with the first commercial Android device launched in September 2008. The operating system has since gone through multiple major releases, with the current version being 8.1 "Oreo", released in December 2017 Android Inc. was founded in Palo Alto, California in October 2003 by Andy Rubin, Rich Miner, Nick Sears, and

Chris White (*PhoneArena, 2011).* In July 2005, Google acquired Android Inc. for at least $50 million *(*Manjoo & Farhad, 2015). Its key employees, including Rubin, Miner and White, joined Google as part of the acquisition. *"Google Buys Android for Its Mobile Arsenal". Bloomberg Businessweek. Bloomberg L.P. Archived from the original on February 5, 2011. Retrieved March 12, 2017.)*

General features of Android Operating System

Messaging

SMS and MMS are available forms of messaging, including threaded text messaging and Android Cloud To Device Messaging (C2DM) and now enhanced version of C2DM, Android Google Cloud Messaging (GCM) is also a part of Android Push Messaging services.

Auto Correction and Dictionary

Android Operating System has an interesting feature called Auto Correction. When any word is misspelled, then Android recommends the meaningful and correct words matching the words that are available in Dictionary. Users can add, edit and remove words from Dictionary as per their wish (Tech Mirages, 2016).

Web browser

The web browser available in Android is based on the open-source Blink (previously WebKit) layout engine, coupled with Chrome's V8 JavaScript engine. Then the WebKit-using Android Browser scored 100/100 on the Acid3 test on Android 4.0 ICS; the Blink-based browser currently has better standards support. The browser is variably known as 'Android Browser', 'AOSP browser', 'stock browser', 'native browser', and 'default browser'. Starting with Android 4.4 KitKat, Google has mandated that the default browser for Android proper be Google Chrome. Since Android 5.0 Lollipop, the WebView browser that apps can use to display web content without leaving the app has been separated from the rest of the Android firmware in order to facilitate separate security updates by Google.

Voice-based features

Google search through voice has been available since initial release. Voice actions for calling, texting, navigation, etc. are supported on Android 2.2 onwards. As of Android 4.1, Google has expanded Voice Actions with ability to talk back and read answers from Google's Knowledge Graph when queried with specific commands. The ability to control hardware has not yet been implemented.

Multi-touch

Android has native support for multi-touch which was initially made available in handsets such as the HTC Hero. The feature was originally disabled at the kernel level (possibly to avoid infringing Apple's patents on touch-screen technology at the time). Google has since released an update for the Nexus One and the Motorola Droid which enables multi-touch natively.

Multitasking

Multitasking of applications, with unique handling of memory allocation, is available.

Screen capture

Android supports capturing a screenshot by pressing the power and home-screen buttons at the same time. Prior to Android 4.0, the only methods of capturing a screenshot were through manufacturer and third-party customizations (apps), or otherwise by using a PC connection (DDMS developer's tool). These alternative methods are still available with the latest Android.

TV recording

Android TV supports capturing video and replaying it (Google, 2016).

Video calling

Android does not support native video calling, but some handsets have a customized version of the operating system that supports it, either via the UMTS network (like the Samsung Galaxy S) or over IP. Video calling through Google Talk is available in Android 2.3.4 (Gingerbread) and later. Gingerbread allows Nexus S to place Internet calls with a SIP account. This allows for enhanced VoIP dialing to other SIP accounts and even phone numbers. Skype 2.1

offers video calling in Android 2.3, including front camera support. Users with the Google+ Android app can perform video chat with other Google+ users through Hangouts.

Multiple language support
Android supports multiple languages.

Accessibility
Built-in text-to-speech is provided by *TalkBack* for people with low or no vision. Enhancements for people with hearing difficulties are available, as are other aids.

iOS Platform
iOS (formerly iPhone OS) is a mobile operating system created and developed by Apple Inc. exclusively for its hardware. It is the operating system that presently powers many of the company's mobile devices, including the iPhone, iPad, and iPod Touch. It is the second most popular mobile operating system globally after Android.
Originally unveiled in 2007 for the iPhone, iOS has been extended to support other Apple devices such as the iPod Touch (September 2007) and the iPad (January 2010). As of January 2017, Apple's App Store contains more than 2.2 million iOS applications, 1 million of which are native for iPads. These mobile apps have collectively been downloaded more than 130 billion times.
The iOS user interface is based upon direct manipulation, using multi-touch gestures. Interface control elements consist of sliders, switches, and buttons. Interaction with the OS includes gestures such as swipe, tap, pinch, and reverse pinch, all of which have specific definitions within the context of the iOS operating system and its multi-touch interface. Internal accelerometers are used by some applications to respond to shaking the device (one common result is the undo command) or rotating it in three dimensions (one common result is switching between portrait and landscape mode). Apple has been significantly praised for incorporating thorough accessibility functions into iOS, enabling users with vision and hearing disabilities to properly use its products.
Major versions of iOS are released annually. The current version, iOS 11, was released on September 19, 2017. It is available for all iOS devices with 64-bit processors; the iPhone 5S and later iPhone models, the iPad (2017), the iPad Air and later iPad Air models, all iPad Pro models, the iPad Mini 2 and later iPad Mini models, and the sixth-generation iPod Touch.

Windows Phone Platform
According to Wikipedia, Windows Phone (WP) is a family of mobile operating systems developed by Microsoft for smartphones as the replacement successor to Windows Mobile and Zune. Windows Phone features a new user interface derived from Metro design language. Unlike Windows Mobile, it is primarily aimed at the consumer market rather than the enterprise market. It was first launched in October 2010 with Windows Phone 7. Windows Phone 8.1 is the latest public release of the operating system, released to manufacturing on April 14, 2014.

Windows Phone was replaced by Windows 10 Mobile in 2015; it emphasizes a larger amount of integration and unification with its PC counterpart—including a new, unified application ecosystem, along with an expansion of its scope to include small-screened tablets. On October 8, 2017, Joe Belfiore announced that work on Windows 10 Mobile was drawing to a close due to lack of market penetration and resultant lack of interest from app developers *(*Reilly & Claire, 2017).

Types of Mobile Applications
There are three major types of mobile applications and they include:

- Native Mobile Applications
- HTML/Web Mobile Applications
- Hybrid/Cross Platform Mobile Applications

Native Mobile Applications

In a nutshell, native apps provide the best usability, the best features, and the best overall mobile experience. There are some things you only get with native apps:

- **Multi touch** - double taps, pinch-spread, and other compound UI gestures
- **Fast graphics API** - the native platform gives you the fastest graphics, which may not be a big deal if you're showing a static screen with only a few elements, or a very big deal if you're using a lot of data and require a fast refresh.
- **Fluid animation** - related to the fast graphics API is the ability to have fluid animation. This is especially important in gaming, highly interactive reporting, or intensely computational algorithms for transforming photos and sounds.
- **Built-in components** - The camera, address book, geolocation, and other features native to the device can be seamlessly integrated into mobile apps. Another important built-in components is encrypted storage, but more about that later.
- **Ease of use** - The native platform is what people are accustomed to, and so when you add that familiarity with all of the native features they expect, you have an app that's just plain easier to use.
- **Documentation** - There are over 2500 books alone for iOS and Android development, with many more articles, blog posts, and detailed technical threads on sites like StackOverflow.

According to Mario Korf & Eugene Oksman **(**2016), Native apps are usually developed using an integrated development environment (IDE). IDEs provide tools for building debugging, project management, version control, and other tools professional developers need. While iOS and Android apps are developed using different IDEs and languages, there's a lot of parity in the development environments, and there's not much reason to delve into the differences. Simply put, you use the tools required by the device. You need these tools because native apps are more difficult to develop. Likewise, the level of experience required is higher than other development scenarios, you don't just cut and paste Objective-C and expect it to work. Indeed, the technological know-how of your development team is an important consideration. If you're a professional developer, you don't have to be sold on proven APIs and frameworks, painless special effects through established components, or the benefits of having your code all in one place. Let's face it, today a skilled native iOS or Android developer is a rock star, and can make rock star demands. While we've touched on native apps from a development perspective, there's also the more important perspective: the end user. When you're looking for an app, you'll find it in the store. When you start the app, it fires up immediately. When you use the app, you get fast performance, consistent platform look and feel. When your app needs an update, it tells you so. Native apps give you everything you'd expect from the company that built your device, as if it were simply meant to be.

HTML5 / Web Mobile Application

An HTML5 mobile app is basically a web page, or series of web pages, that are designed to work on a tiny screen. As such, HTML5 apps are device agnostic and can be opened with any modern mobile browser. And because your content is on the web, it's searchable, which can be a huge benefit depending on the app. According to Mario Korf & Eugene Oksman **(**2016), an important part of the "write-once-run-anywhere" HTML5 methodology is that distribution and support is much easier than for native apps. Need to make a bug fix or add features? Done and deployed for all users. For a native app, there are longer development and testing cycles, after which the consumer typically must log into a store and download a new version to get the latest fix.

In the last year, HTML5 has emerged as a very popular way for building mobile applications. Multiple UI frameworks are available for solving some of the most complex problems that no developer wants to reinvent. iScroll does a phenomenal job of emulating momentum style scrolling. JQuery Mobile and Sencha Touch provide elegant mobile components, with hundreds if not thousands of plugins that offer everything from carousels to super

elaborate controls. So if HTML5 apps are easier to develop, easier to support, and can reach the widest range of devices, where do these apps lose out? We already reviewed the major benefits of native development, so we'll just reiterate that you can't access native features on the device. Users won't have the familiarity of the native look and feel, or be able to use compound gestures they are familiar with. But strides are being made on all fronts, and more and more functionality is supported by browsers all the time.

Hybrid Mobile Application

According to the technical library at Salesforce developers, Hybrid development combines the best (or worst) of both the native and HTML5 worlds. According to Pietro Saccomani (2017), if a native app and a web app got married and had a kid; it would be a hybrid app. You install it like a native app, but it's actually a web app on the inside. Hybrid apps, like web apps, are built with JavaScript, HTML, and CSS and run in something called WebView, a simplified browser within your app. Hybrid application development can essentially do everything HTML5 does, and it also incorporates native app features. This is possible when you deploy a wrapper to act as a bridge between platforms to access the native features. A hybrid app consists of two parts. The first is the back-end code built using languages such as HTML, CSS, and Javascript. The second is a native shell that is downloadable and loads the code using WebView. John Bristowe (2015) claims that Hybrid mobile applications are built in a similar manner as websites. Both use a combination of technologies like HTML, CSS, and JavaScript. However, instead of targeting a mobile browser, hybrid applications target a WebView hosted inside a native container. This enables them to do things like access hardware capabilities of the mobile device.

Hybrid development combines the best of both the native and HTML5 worlds. We define hybrid as a web app, primarily built using HTML5 and JavaScript that is then wrapped inside a thin native container that provides access to native platform features. PhoneGap is an example of the most popular container for creating hybrid mobile apps. For the most part, hybrid apps provide the best of both worlds. Existing web developers that have become gurus at optimizing JavaScript, pushing CSS to create beautiful layouts, and writing compliant HTML code that works on any platform can now create sophisticated mobile applications that don't sacrifice the cool native capabilities. PhoneGap is an example of the most popular container for creating hybrid mobile apps (Stepin Solutions, 2016).

On iOS, the embedded web browser or the UI-WebView is not identical to the Safari browser. While the differences are minor, they can cause debugging headaches. That's why it pays off to invest in popular frameworks that have addressed all of the limitations. The most common hybrid application development framework is Apache Cordova, that enables applications to execute across different platforms by relying on standards-compliant API bindings that provides access to the different native device capabilities such as, camera, GPS, accessibility, and so on (Pumex Computing, 2017). Native applications are installed on the device, while HTML5 apps reside on a Web server, so you might be wondering if hybrid apps store their files on the device or on a server. The answer is Yes. In fact, according to Mario Korf and Eugene Oksman (2016) there are two ways to implement a hybrid app.

- **Local** - You can package HTML and JavaScript code inside the mobile application binary, in a manner similar to the structure of a native application. In this scenario you use REST APIs to move data back and forth between the device and the cloud.
- **Server** - Alternatively you can implement the full web application from the server (with optional caching for better performance), simply using the container as a thin shell over the UI-WebView.

Stages of Program Development

According to Rhodes (2012) and PhoneGap: Supported features (2012), there are five stages of software development which include clarifying programming needs, designing of the program coding of the program and documenting and maintaining the program. According to them, clarifying the programming need consist of clarifying the objective and users of the program clarifying the desired output, clarifying the desired input, clarifying

the desired processing, double checking the flexibility of implementation of the program and documenting the analysis. Designing stage of the program is done in three (3) main stages; the first stage determines the program logic through a top-down approach and anodularization using a hierarchy chart. According to them, it is designed in detail either in narrative from using pseudo code or graphic from using flowchart. They also identify the three (3) mini steps of program design as, determination of the program logic using top-down approach, designing details using pseudo code and flowchart and detailed program design. According to them, programming stage involves coding of instructions based on the design made. Once designs have developed the actual writing of the program begins. It implies translating of the logic environment from pseudo code or flowchart into programming languages. The forth step implied testing of the program according to Pilgrim, M.: Dive into HTML5: Local storage (2011), program testing involves running various tests and then running real-word data to make sure the program works.

Anya O. (2008), identify three (3) stages of program testing which include performing test checking, debugging the program and running real-world data. The fifth and final step is documenting, and maintaining the program. According to C. Best (IBID) documentation consists of written description of what a program is and how to use it. According to Anya O. (IBID), documentation should be prepared for different kind of reader, users and programmers. He went further to identify stages of documentation to include preparation of user documentation preparation operator documentation writing of programmer documentation and maintaining the program.

What are the motivations to go Cross Platform/Hybrid?

According to Stepin Solutions (2016), a hybrid application makes it possible to embed HTML5 apps inside a thin native container, combining the elements of native and HTML5 apps. This write-once-run-anywhere approach to mobile development creates cross-platform mobile applications that work on multiple devices. Hybrid mobile applications provide a way for developers to re-use their existing skills in web development. Developers don't like the prospect of getting locked into proprietary platforms. This includes the programming languages and SDKs provided by platform vendors (more on this later).

Hybrid mobile application development looks appealing to an organization's bottom line. Why hire a developer for each platform when you can hire one developer and target all of them through HTML, CSS, and JavaScript? Well, the reality is a bit more complicated. Yes, it's true that hybrid mobile application development enables developers to target more than one platform. However, each platform comes with a set of caveats when it comes to its web runtime or WebView. This is especially true with Android, which is inconsistent between OS versions. Moreover, there might be unique capabilities of platforms to which a developer may wish to target. In those instances, a combination of plugins and platform-specific code must be utilized in order to take advantages of those capabilities. Alternatively, developers can take advantage of 3rd party web runtimes like Crosswalk that can be embedded in your hybrid app.

System Design Methodologies

There are many methodologies used for system analysis and design like the Structured System Analysis and Design Methodology (SSADM), the Unified Modeling Language (UML). Domain Model, Prototyping, Hierarchy Input Processing Output (HIPO), etc. The design strategy used is Object Oriented Software Design (OOD). The design tool used is the Unified Modeling Language (UML).

Object-Oriented Design (OOD)

The object-oriented approach combines data and processes (called methods) into single entities called objects. Objects usually correspond to the real things a system deals with, such as customers, suppliers, contracts, and invoices. Object-oriented models are able to thoroughly represent complex relationships and to represent data and data processing with a reliable notation, which allows an easier mix of analysis and design in a growth process. The aim of the Object-Oriented approach is to make system elements more modular, thus improving system quality and the efficiency of systems analysis and design. In the Object-Oriented approach we tend to focus more on the behavior of the system. The main feature we document is the Object or Class. Objects may be distributed and executed either sequentially or in parallel. Decisions on parallelism need not be taken at an early stage of the process.

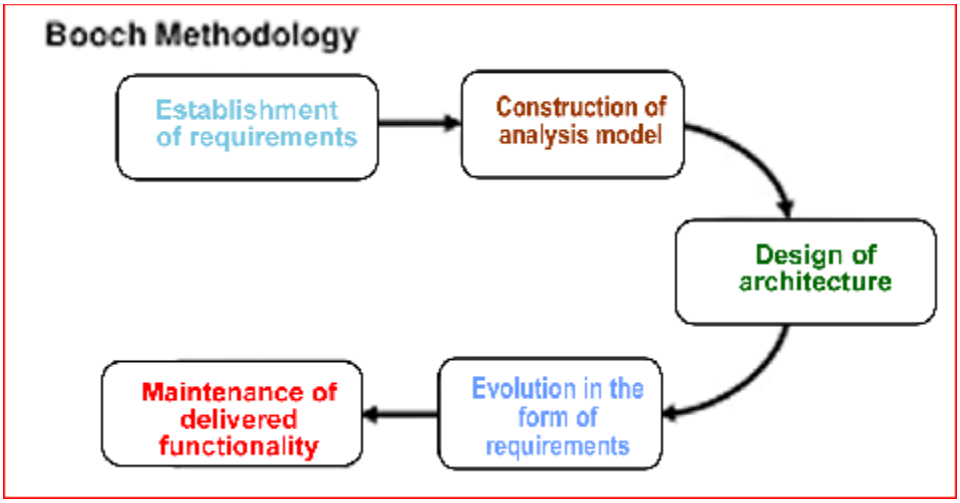

Fig 2. Object Oriented Design (OOD)

Class Diagram of the Proposed System

In UML, a class diagram is a type of static structure diagram that describes the structure of a system showing the system's classes, their attributes, and the relationships between the classes. The primary purpose of the class diagram is to create a vocabulary that is used both by the analyst and users. Class diagrams typically represent the things, ideas or concepts that are contained in the application. Classes are composed of three things: a name, attributes, and operations. Class diagrams also display relationships such as containment, inheritance, associations and others. A class icon is simply a rectangle divided into three compartments. The topmost compartment contains the name of the class. The middle compartment contains a list of attributes (member variables), and the bottom compartment contains a list of operations (member functions). In many diagrams, the bottom two compartments are omitted. Even when they are present, they typically do not show every attribute and operations. The goal is to show only those attributes and operations that are useful for the particular diagram. The diagram below, Fig 3.1, shows the class diagram of the proposed system

Remarks: below are few things to note about the system's class diagram

1. Rectangles represent classes and arrows represent associations in which an object holds a reference to and invokes functions upon the other.
2. A dash character (-) in front of the variables denotes a private variable.
3. A plus character (+) in front of the functions denotes a public function.

The type of variable argument or function argument is shown after the colon following it.

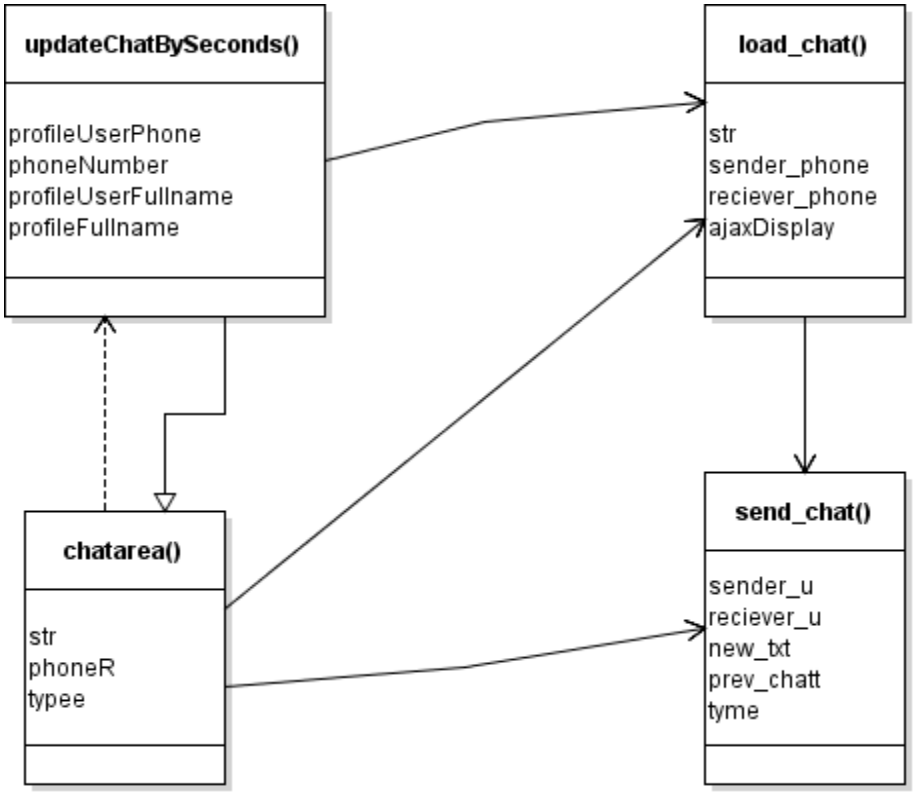

Fig 1.3 chat_table System Class Diagram

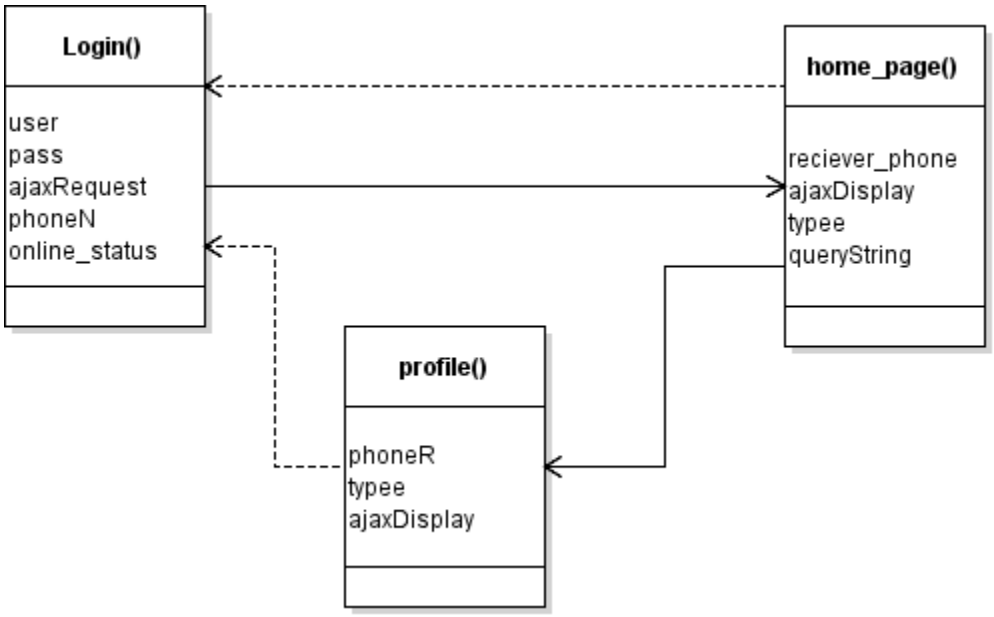

Fig 4. User_profile System Class Diagram

Use Case Diagram of the Proposed System

Use case diagrams are diagrams that portray the basic functions of the system-that is, what the users can do and how the system should respond to the user's actions. The use case communicates at a high level what the system needs to

do. Use cases capture the typical interaction of the system with the system's users (end users and other systems). These interactions represent the external, or functional, view of the system from the perspective of the user. Each use case describes one and only one function in which users interact with the system, although a use case may contain several "paths" that a user can take while interacting with the system (e.g., when searching for a book in Web bookstore, the user might search by subject, by author, or by title). Each path through the use case is referred to as a *scenario.* Fig 1.2 show the use case diagram of my system.

Things to know about the use case diagram include:

- Actor: The labeled stick figures on the diagram represent actors. An *actor* is not a specific user, but a role that a user can play while interacting with the system.
- **Association**: Use cases are connected to actors through *association relationships,* which show with which use cases the actors interact. A line drawn from an actor to a use case depicts an association.
- **Use Case**: A *use case*, depicted by an oval in the UML, is a major process that the system will perform that benefits an actor(s) in some way, and it is labeled using a descriptive verb-noun phrase.
- The large rectangle is the systems boundary and everything inside it is part of the system under development.

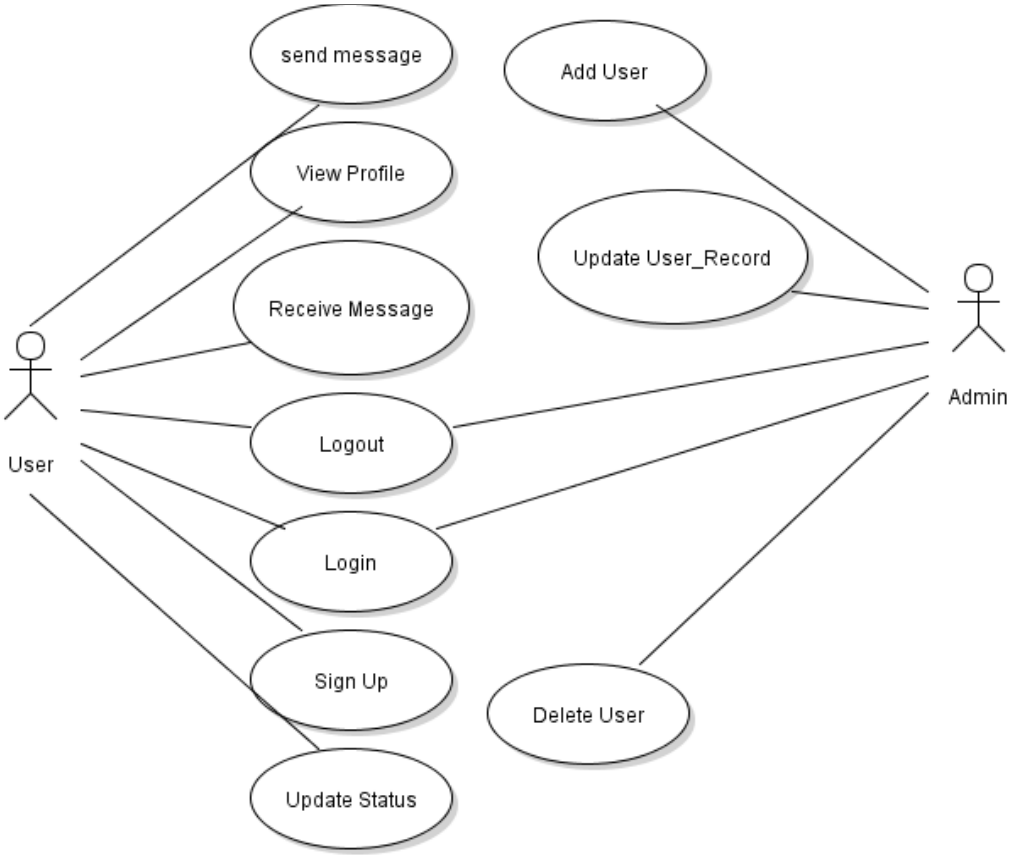

Fig 5. System Use Case Diagram

Existing System

From the investigation and data analysis carried out, it was discovered that the use of native built mobile application is very discouraging. The researcher discovered that many students and staffs using mobile applications on their phones do not know if it is a cross-platform mobile application or a native mobile application. Also, the researcher noticed that nine out of ten students and staffs have candy crush game in their phone, which is a cross-platform mobile application. The problem identified is that for a mobile application developer to build an application for

several mobile operating systems (i.e. Android, Java, or iOS) he must build a different program entirely, for each operating system. This in turn, makes mobile application development lot more stressful and discouraging for programmers who intend to venture into it. The researcher therefore concluded that there is the need for efficient and effective development method, which will enable the developer write a program just once and the deploy that same program to several mobile platforms which saves time and cost.

Design of the Proposed System

The database management system used is MySQL.

Entity Relationship (ER) Diagram

An entity relationship model, also called an entity-relationship (ER) diagram, is a graphical representation of entities and their relationships to each other, typically used in computing in regard to the organization of data within databases or information systems. Entity-relationship modeling is a high-level data modeling technique that helps designers create accurate and useful conceptual models. E-R models are best ex- pressed using graphical E-R diagrams. This technique was originally developed by Professor Peter Chen to serve as a tool for communication between designers and users. Chen recognized the problems that are caused when developers and users fail to understand each other. It is typical for developers and users to think that they each know exactly what the other is thinking. Unfortunately, human communication is not that good. In the presence of misunderstanding, developers build information systems that do not meet user needs. The result is either the total failure of the sys- tem that was developed or a major increase in costs as the system is rewritten.

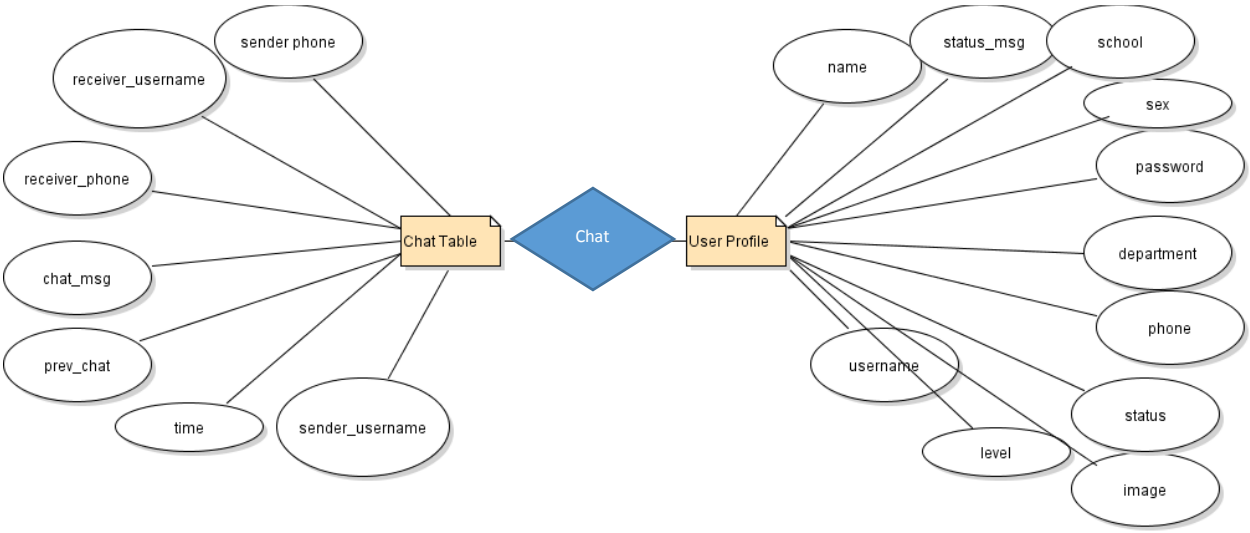

Fig 6. ER Diagram for the Database of the proposed system.

Database Models

A data model can be thought of as a diagram or flowchart that illustrates the relationships between data. Although capturing all the possible relationships in a data model can be very time-intensive, it's an important step and shouldn't be rushed. Well-documented models allow stake-holders to identify errors and make changes before any programming code has been written. An ER model is an abstract way of describing a database. In the case of a relational database, which stores data in tables, some of the data in these tables point to data in other tables - for instance, your entry in the database could point to several entries for each of the phone numbers that are yours. The ER model would say that you are an entity, and each phone number is an entity, and the relationship between you and the phone numbers is 'has a phone number'. Diagrams created to design these entities and relationships are called entity–relationship diagrams or ER diagrams. The DBMS used for the database design of this system is MySQL and the ER modeler used is the MySQL workbench modeler. The following images represent some of the database tables of the proposed system.

The following are the Enhanced Entity Relationship (EER) models of the proposed system.

Conclusion

The problem identified is that for a mobile application developer to build an application for several mobile operating systems (i.e. Android, Java, or iOS) he must build a different program entirely, for each operating system. This in turn, makes mobile application development lot more stressful and discouraging for programmers who intend to venture into it. In conclusion, there is a need for efficient and effective mobile application development method, which will enable the developer write a program just once and the deploy that same program to several mobile platforms which saves time and cost. The debate around which type of app is the best is still very relevant today as the lines between the three approaches are becoming increasingly blurred. While the discussion to differentiate the three mobile apps continues, it's important to remember that you shouldn't choose an approach for the technology, but instead, choose based on what you want your app to do. If you choose an approach that doesn't allow your app to utilize device features, for example, then you'll end up wasting a lot of time and money when you decide to add features. The choice between native, web, or hybrid is dependent on a number of factors, including business needs, app requirements, developer skill, and timelines. The bottom line is that each type of app offers an entirely different experience and it's important to know the strengths and weakness of each before jumping into development.

REFERENCES

Anya O. (2009). There is no WebKit on mobile http://quirksmode.org/blog/archives/2009/10/there_is_no_web.html

Benjamin S. (2008). The Android mobile platform. Retrieved from: https://www.google.com.ng/url?sa=t&rct=j&q=&esrc=s&source=web&cd=2&cad=rja&uact=8&ved=0ahUKEwiJzuaZvNfYAhWIIsAKHck4CDMQFgguMAE&url=http%3A%2F%2Fciteseerx.ist.psu.edu%2Fviewdoc%2Fdownload%3Fdoi%3D10.1.1.175.2961%26rep%3Drep1%26type%3Dpdf&usg=AOvVaw3-lGzIB5w2_oZ1mmWGmNdT

Cory, J. (2013). Software library, Janalta interactive, http://www.techopedia.com/definition/3828/software-library: Accessed 14 May, 2013.

Curtis .B. (2011). PhoneGap and the Enterprise. http://www.slideshare.net/drbac/phonegap-day-ibm-phonegap-andthe-enterprise, Retrieved July, 2011 .

Darcey L. & Conder S. (2012). Android Wireless Application Development. Retrieved from http://www.informit.com/articles/article.aspx?p=1388959: Assessed August 10, 2012.

Edwin, A. (2014). The rise and fall of 2go mobile social chat network. Retrieved from http://www.highteknology.com/2013/08/the-rise-and-fall-of-2go-mobile-social.html

Eran, Z. (2012) Native, HTML5, and Hybrid Mobile App Development: Real-Life Experiences, online video, http://www.youtube.com/watch?v=We0byPckthQ: Accessed on 16 May, 2013.

Gartner (2013). Gartner press release 2013, http://www.gartner.com/newsroom/id/2408515: Accessed 12, April 2013.

Github (2013). mustache.js - Logic-less {{mustache}} templates with JavaScript, Github inc., https://github.com/janl/mustache.js/#readme : Accessed 11 May, 2013.

Goadrich, M.H., Rogers, M.P. (2011): Smart smartphone development: iOS versus Android. In: Proc. SIGCSE '11. pp. 607–612. ACM, New York, NY, USA.

Google. (2012). Android open source project, http://source.android.com/ Retrieved March 9, 2016.

Google. (2016). "TV recording: Android Developers". Retrieved March 9, 2016.

http://blog.comakeit.com/9-hybrid-apps-that-are-changing-the-status-quo

IDC. (2013). International Data Corporation, 2013 Press Release, http://www.idc.com/getdoc.jsp?containerId=prUS23946013: Accessed April 12, 2013.

Ikediashi. (2012). What is MoSync. http://www.mosync.com/content/mosynccross-platform-mobile-development-made-easy, Accessed 2012.

iPhonical. (2010). http://code.google.com/p/iphonical/

John .B. (2015). What is a Hybrid Mobile App? Retrieved from https://www.developer.telerik.com/featured/what-is-a-hybrid-mobile-app/: Assessed March 25, 2015.

jQuery project license (2012). http://jquery.org/license/

jQuery Mobile (2011). http://jquerymobile.com/

jQuery Mobile documentation (2012). http://jquerymobile.com/demos/1.1.0/

jQuery Mobile graded browser support (2012). http://jquerymobile.com/gbs/

Korf, M and Oksman, E (2013). Native, HTML5, or Hybrid: Understanding Your Mobile Application Development Options, http://.wiki.developerforce.com/page/Native,_HTML5,_or_Hybrid:_Understanding_Your_Mobile_Application_Development_Options : Accessed May 18, 2013.

Korf, M and Oksman E. (2016). Native, HTML5, or Hybrid: Understanding Your Mobile Application Development Options. Retrieved from https://developer.salesforce.com/page/Native,_HTML5,_or_Hybrid:_Understanding_Your_Mobile_Application_Development_Options Assessed June 2016.

Manjoo, F. (2015). "A Murky Road Ahead for Android, Despite Market Dominance". The New York Times. Retrieved March 12, 2017.

Mercy Rop (2017). What is a Mobile Application? Retrieved from https://legibra.com/mobile-application/ : Assessed May 15, 2017.

Pietro S. (2017). Native, Web or Hybrid Apps; what's the Difference? Retrieved from https://www.mobiloud.com/blog/native-web-or-hybrid-apps/:Assessed December 13, 2017.

PhoneArena, (2011). "Google's Android OS: Past, Present, and Future". Retrieved March 12, 2017.

PhoneGap, (2012). Support, http://phonegap.com/support#support-packages

PhoneGap, (2012). Supported features, http://phonegap.com/about/features/

PhoneGap, (2012). Build, https://build.phonegap.com

Pilgrim, M. (2011). Dive into HTML5: Local storage, http://diveintohtml5.info/storage.html

Pumex Computing (December, 2017). Native and Hybrid Apps. Retrieved from http://www.pumexcomputing.com/blog/native-and-hybrid-apps/

Reilly, C. (2017). *"Windows 10 Mobile gets its final death sentence"*. *CNET*. Retrieved 10 September, 2017.

Rhodes (2012), http://www.motorola

Stepin Solutions. (2016). Native Mobile App Vs. Hybrid Mobile App. Retrieved from http://www.stepin-solutions.com/blog/native-mobile-app-vs-hybrid-mobile-app/: Assessed April 2, 2016.

Techterms.com. (2013). Framework, http://www.techterms.com/definition/framework: Accessed 14 May, 2013.

Tech Mirages (2016-10-22), How To Add or Remove Words From Android Dictionary, www.youtube.com/watch?v=b4dxlsyb9lg Retrieved 2017-12-25

