

# Review of Suggesting Exercise Based On Mood Through Mobile Application

Shubham Kumar<sup>1</sup>, Prof. Sarwesh Site<sup>2</sup>

<sup>1</sup>M.Tech Scholar, <sup>2</sup>Assistant Professor, Department of Computer Science and Engineering All Saints' College of Technology, Bhopal, India

**Abstract:**— Android is an open-source software stack that contains operating system, Middleware, User Interface that is built on top of Linux operating system. The android application has been rapidly evolving towards making a greater impact on everyday life to large population. There is lots of type of app like Productivity, lifestyle, shopping, Food & beverages, Music and audio, Business, tool, Education, Book and References, Entertainment.

As per my research, this has attracted the attention of mood who suggest an Android app that target various activity like Exercise, Yoga, Jumba, Meditation and game like paint by number of any image to release your stress. This paper presents a review of forest Counselling android application.

**Keywords:** – Internet, Android, Life Style, Mood

## I. INTRODUCTION

**Android Application:** It is Smart phone mobile application that provides many features in mobile like smart phone and tablets. It is basically designed by Google. Google provides download features in play Store app. So user can download any application like social media application like Facebook, Instagram, twitter any many more and shopping related application like flipcart, amazon, mессo like that, also many more gaming application and news application. Here my application is mood related response to user get stress free in daily life to use it. So I have developed a mobile application related mood where the process of application to use the user can simply select any mood that is mention there and then go to the activities related screen where user can select single /multiple activities and get the mood and activities average graph and count in the stats page and also overall ratio get in the home page. This app also give suggestions about exercise and yoga which is mood fit and healthy to do in your daily/lifetime.



Figure 1: Android Emulator/Virtual Device

In Android emulators make it quick and easy to run Android apps and games on your laptop PC or Mac.

Maybe you're working on an app for Android and want to test it on a computer instead of a mobile device. You can not only do that with one of these Android emulators but you can simulate various screen sizes and phone models as well, letting you see how it does in different sizes of screen.

Or, maybe you want to run an app that's only available on Android. Take Facebook, for example. Unless you're on the mobile app, you're limited in what you can do with the social media platform. But, with an emulator, you can edit and upload photos from your laptop or Mac desktop – something you just can't do otherwise.

## II. BACKGROUND

Java is a high-level, class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible.



**International Journal of Recent Development in Engineering and Technology**  
**Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 11, Issue 09, September 2022)**

It is a general-purpose programming language intended to let programmers write once, run anywhere (WORA),[17] meaning that compiled Java code can run on all platforms that support Java without the need to recompile.[18] Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of the underlying computer architecture. The syntax of Java is similar to C and C++, but has fewer low-level facilities than either of them. The Java runtime provides dynamic capabilities (such as reflection and runtime code modification) that are typically not available in traditional compiled languages. As of 2019, Java was one of the most popular programming languages in use according to GitHub,[19][20] particularly for client-server web applications, with a reported 9 million developers.[21]

Java was originally developed by James Gosling at Sun Microsystems. It was released in May 1995 as a core component of Sun Microsystems' Java platform. The original and reference implementation Java compilers, virtual machines, and class libraries were originally released by Sun under proprietary licenses. As of May 2007, in compliance with the specifications of the Java Community Process, Sun had relicensed most of its Java technologies under the GPL-2.0-only license. Oracle offers its own HotSpot Java Virtual Machine, however the official reference implementation is the OpenJDK JVM which is free open-source software and used by most developers and is the default JVM for almost all Linux distributions.

R. The first (XML 1.0) was initially defined in 1998. It has undergone minor revisions since then, without being given a new version number, and is currently in its fifth edition, as published on November 26, 2008. It is widely implemented and still recommended for general use.

S. The second (XML 1.1) was initially published on February 4, 2004, the same day as XML 1.0 Third Edition,[35] and is currently in its second edition, as published on August 16, 2006. It contains features (some contentious) that are intended to make XML easier to use in certain cases.[36] The main changes are to enable the use of line-ending characters used on EBCDIC platforms, and the use of scripts and characters absent from Unicode 3.2. XML 1.1 is not very widely implemented and is recommended for use only by those who need its particular features.[37]

T. Prior to its fifth edition release, XML 1.0 differed from XML 1.1 in having stricter requirements for characters available for use in element and attribute names and unique identifiers: in the first four editions of XML 1.0 the characters were exclusively enumerated using a specific version of the Unicode standard (Unicode 2.0 to Unicode 3.2.) The fifth edition substitutes the mechanism of XML 1.1, which is more future-proof but reduces redundancy.

The approach taken in the fifth edition of XML 1.0 and in all editions of XML 1.1 is that only certain characters are forbidden in names, and everything else is allowed to accommodate suitable name characters in future Unicode versions. In the fifth edition, XML names may contain characters in the Balinese, Cham, or Phoenician scripts among many others added to Unicode since Unicode 3.2.[36]

U. Almost any Unicode code point can be used in the character data and attribute values of an XML 1.0 or 1.1 document, even if the character corresponding to the code point is not defined in the current version of Unicode. In character data and attribute values, XML 1.1 allows the use of more control characters than XML 1.0, but, for "robustness", most of the control characters introduced in XML 1.1 must be expressed as numeric character references (and #x7F through #x9F, which had been allowed in XML 1.0, are in XML 1.1 even required to be expressed as numeric character references[38]). Among the supported control characters in XML 1.1 are two line break codes that must be treated as whitespace characters, which are the only control codes that can be written directly.

V. Development lead Andrey Breslav has said that Kotlin is designed to be an industrial-strength object-oriented language, and a "better language" than Java, but still be fully interoperable with Java code, allowing companies to make a gradual migration from Java to Kotlin.[25]

Semicolons are optional as a statement terminator; in most cases a newline is sufficient for the compiler to deduce that the statement has ended.[26]

Kotlin variable declarations and parameter lists have the data type come after the variable name (and with a colon separator), similar to Ada, BASIC, Pascal, TypeScript and Rust. This, according to an article from Roman Elizarov, current project lead, results in alignment of variable names and is more pleasing to eyes especially when there are a few variable declarations in succession and one or more of the types is too complex for type inference or needs to be declared explicitly for human readers to understand.[27][28] Variables in Kotlin can be read-only, declared with the `val` keyword, or mutable, declared with the `var` keyword.[29] Class members are public by default, and classes themselves are final by default, meaning that creating a derived class is disabled unless the base class is declared with the open keyword.



**International Journal of Recent Development in Engineering and Technology**  
**Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 11, Issue 09, September 2022)**

In addition to the classes and member functions (which are equivalent to methods) of object-oriented programming, Kotlin also supports procedural programming with the use of functions.[30] Kotlin functions and constructors support default arguments, variable-length argument lists, named arguments and overloading by unique signature. Class member functions are virtual, i.e. dispatched based on the runtime type of the object they are called on.

Kotlin 1.3 added support for contracts,[31] which are stable for the standard library declarations, but still experimental for user-defined declarations. Contracts are inspired by Eiffel's design by contract[32] programming paradigm.

Kotlin code may be compiled to JavaScript, allowing for interoperability between code written in the two languages. This can be used either to write full web applications in Kotlin, or to share code between a Kotlin backend and a JavaScript frontend.[33]

### III. ANDROID DEVELOPMENT TECHNIQUES

#### A. Jetpack

Google's Android / Jetpack libraries are a collection of utilities, aimed at simplifying common app needs. For example, Room for an on-device database, or Live Data to update displays when the underlying data changes.

Instead of reinventing the wheel with each project, or relying on another developer open sourcing their implementation, the Jetpack libraries allow everyone to have access to the essentials. These libraries are updated very frequently, with new features as well as timely bug fixes. As the libraries are built to work together, implementing AndroidX libraries where possible helps minimise the amount of unexpected behaviour in the app.

#### B. Modular design

Historically, apps were built as one giant "app" module, that contained everything the entire app needed. Whilst this did make sharing resources easy, it did mean parts of the app couldn't be reused for other apps / open source, and more importantly the entire codebase had to be recompiled when making changes.

If, instead, an app consists of many smaller modules, only the changed code needs to be recompiled, leading to much faster build times. Additionally, it opens the door to using advanced Android features such as Instant Apps (users can use part of your app without installing anything) and Dynamic Features (installing additional parts of the app when they are needed).

Splitting an existing app into modules can be painful, as previously hidden isolation issues are revealed ("What is Date Utility, and why does every class need it!?"), but once completed the codebase will be in a much healthier state.

Plus, if a new app ever needs similar functionality, being able to instantly reuse existing modules will save time!

#### C. App Bundles

When distributing an app to a user's device using a traditional APK, all resources for every possible device have to be installed. This means 5 copies of every bitmap image (for different screen densities), multiple copies of libraries for different device architectures, and even multiple sets of values for margins and paddings.

When distributing using an app bundle, the APK the user downloads instead just contains the resources they actually need. This results in average app size reductions of 20%, with unoptimised apps receiving much more impressive reductions. Some earlier exploration has analyzed the utilization of various strategies to fabricate AIDs. Analyzed the exhibition of two element choice App bundles have only been around for 18 months, but already over 25% of app installs are of apps using them! They are recommended by Google, and most apps will work with them without any work besides that required for app bundle signing on the Play Store.

At Candy space we are in the process of migrating to app bundles, whilst being careful to avoid breaking existing workflows (Slack, QAing builds, non-Google Play installs). Alistair Sykes' article is an excellent resource for migration, taking into account CI servers, Slack, and Google Play Internal App Sharing.

Yep, testing. Sure, it's not a shiny new feature, or anything the user will actually see, but thoroughly testing your app to ensure reliability is absolutely essential on an app with an established user base. As crash rates can directly influence your Play Store rating (and definitely affect your ratings!), keeping it low is essential.

### IV. CONCLUSION

This research has demonstrated the development of an Android Web Application that operates on a network/gps which connection can be installed on Android devices with minimum sdk version 2.2 or API Level 8. The basic requirement of this application is to have Wi-Fi connection availability. Once the application is installed, it can be moved to SD - Card. To provide users with a vast range of capability to analyze chronological change, negotiation has been made on scale over the number of maps available.



**International Journal of Recent Development in Engineering and Technology**  
**Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 11, Issue 09, September 2022)**

Therefore, one inch to one mile scale Ordnance Survey maps have been used.

An online questionnaire was prepared along with hands on testing of the application to judge the success of the application. No major issues were found with the results of users' feedback. This feedback showed that everyone had found the application to be very much useful in analyzing historical change. Every participant agreed that the application can be easily navigated and was quite easy to handle and operate.

Overlay technique was much preferred by the users over the Series to analyze the historical change. However, some of the participants were not satisfied with the scale of the maps used. A few user interface issues were also faced and raised by the users which are expected to be delivered in the future release. On the whole, an appreciative attitude was observed towards using the application in spite of some above mentioned critical comments.

Learning Android App Development is still a great thing to do because there is so much going on in this area. It's a profit-making profession on high demand.

There are many innovative ideas in the mobile applications market, and the growth of new technologies that will improve the mobile environment assure us a great outcome for the investment in these type of careers.

Smart home apps, voice-activated apps, artificial intelligence and Chabot's in e-commerce, are only some of the technologies looking to be part of the mobile world. The main goal of all them is to make our lives easier.

We all use our mobile devices and applications as part of our lives, whether we choose to use it for entertainment, for being more productive in our jobs or for shopping. So, it's not an exaggeration to say that we depend on the Application Developers to do many things in our daily lives today.

Investing in education is investing in your future, therefore choosing a career that will give you a good job perspective is a good call to make.

Above careers like dentistry, physician assistant and nurse practitioner, the best job in 2018 is Software Developer. These professionals are in huge demand and are expected to earn more than \$100,000 annually.

Aside from job opportunities in the software developer environment, areas related to Engineering and Computer Science are also in the top of the best lucrative careers in 2018.

Software developers are seen as a self-taught professionals, who live mostly connected to their computer and are always looking for new ways to increase their knowledge.

This profession is so wanted that even companies are forced to create flexible and learning-driven workplaces to hire and retain people.

Companies are looking for talented developers with experience in creating projects and not because of the number of degrees they own. Also, employers evaluate problem-solving skills rather than programming language fluency.

#### *The growth of m-commerce*

All businesses need an app, this allows them to obtain benefits like Brand awareness, increase revenues, become mobile-friendly (understanding the consumption habits of today) and generate engagement.

The mobile application market has a good projection: by 2021 it is expected that the number of mobile app downloads worldwide reach the 352 billion.

Also, the mobile commerce is expected to grow. By the end of 2018, China – the world's largest e-commerce market – is expected to have a 75% of e-commerce sales via a mobile device since m-commerce is starting to take a place in the market, as we've seen before.

Users spend more time using their mobile devices than using their desktop computer. The difference of the time spent on each one is significant and more than likely to increase in the future: 59% of their time spent on a mobile device vs 41% on a desktop computer.

Learning Android App Development is still a great thing to do because it opens the door to a lot of opportunities. There is a huge demand of software developers and the payment is quite good.

Regarding this, if you are interested on this particular area you may consider to enroll into the Professional Android Developer MicroMaster Program and learn with Google Developer Experts. All courses starting on February 11, 2018.

#### REFERENCES

- [1] Introduction to Android: <http://developer.android.com/guide/index.html>.
- [2] Android API: <http://developer.android.com/reference/packages.html>
- [3] Java 6 API: <http://docs.oracle.com/javase/6/docs/api/>
- [4] Android Fundamentals: <http://developer.android.com/guide/components/fundamentals.html>
- [5] The Java Tutorials: <http://docs.oracle.com/javase/tutorial/>
- [6] Android User Interfaces: <http://developer.android.com/guide/topics/ui/index.html>
- [7] Layout: <http://developer.android.com/guide/topics/ui/declaring-layout.html>
- [8] Common Tasks: <http://developer.android.com/guide/appendix/faq/commontasks.html>



**International Journal of Recent Development in Engineering and Technology**  
**Website: [www.ijrdet.com](http://www.ijrdet.com) (ISSN 2347-6435(Online) Volume 11, Issue 09, September 2022)**

- [9] Google Maps: <http://code.google.com/android/add-ons/google-apis/maps-overview.html>
- [10] Iconography: [http://developer.android.com/guide/practices/ui\\_guidelines/icon\\_design.html](http://developer.android.com/guide/practices/ui_guidelines/icon_design.html)
- [11] Sample Source Code: <http://developer.android.com/resources/samples/get.html>
- [12] Android Training: <http://developer.android.com/training/index.html>
- [13] Android Developer's Blog: <http://android-developers.blogspot.com/>
- [14] Developer FAQ: <http://developer.android.com/resources/faq/>
- [15] Developer Forums: <http://developer.android.com/resources/community-groups.html>
- [16] Android Developer's Group: <http://groups.google.com/group/android-developers?lnk=>
- [17] XDA-Developers Forums: <http://forum.xda-developers.com/>