Android Project

Business Card: Ecard

Prepared by

A.K.M. Nazmul Islam

2016-1-96-001

Under the guidance of

Khan Mohammad Habibullah

Lecturer

East West University

Statement:

We hereby proclaim that this thesis is based on the results we found by our hard work. Contents of the work found by other researcher(s) are motioned by references. This thesis has never been previously submitted for any degree neither in whole nor in part.

Signature of Supervisor:

Khan Mohammad Habibullah

Lecturer

Department of Computer Science and Engineering

East West University, Dhaka, Bangladesh.

Signature of Chairperson:

Dr. Ahmed Wasif Reza

Associate Professor and Chairperson

Department of Computer Science and Engineering

East West University, Dhaka, Bangladesh.

Acknowledgment:

I would like to thank almighty ALLAH for giving us the opportunity and aptitude to complete a valuable amount of knowledge in this life span and with His blessings; I shall continue to pursue all the good things in our life ahead.

I also thank our supervisor, Khan Mohammad Habibullah heartily for giving us the opportunity to work under him and also for his kind support in making this project successful.

I thank East West University for giving me the opportunity to be a part of it and all the faculty members who have taught and motivated me to think more deeply.

Finally, I would like to thank all the seniors and juniors who have helped in minute tasks. I may miss names so I thank you all heartily.

Signature of student:

A.K.M. Nazmul Islam

2016-1-96-001

Department of Computer Science and Engineering

East West University, Dhaka, Bangladesh.

Abstract

This project is about a professional business card named “Ecard” in the academic purpose. Android is the most popular platform for mobile operating system. The power of the mobile operating system is increasing rapidly. In this report, I proposed the design and implementation of a professional business card using this platform. Designing and develop useful android app with user interfaces by using extending and creating own layouts, activities and views of the project. This project is developed in Java Programming Language by using Android Studio. I have used the Android software development applications on the Android platform. The most important of these are android software testing Emulator and the tools itself provided by the Google.

Table of Contents

List of figures

Chapter 1: Introductions

1.1 Introduction

Android is a mobile operating system (OS) currently developed by Google based on the Linux kernel and designed primarily for touch screen mobile devices such as smart phones and tablets. Android's user interface is mainly based on direct manipulation, using touch gestures that loosely correspond to real-world actions such as swiping, tapping and pinching, to manipulate on-screen objects, along with a virtual keyboard for text input. In addition to touch screen devices, 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 notebooks, game consoles, digital cameras, and other electronics.

Initially developed by Android Inc., which Google bought in 2005, Android was unveiled in 2007, along with the founding of the Open Handset Alliance – a consortium of hardware, software, and telecommunication companies devoted to advancing open standards for mobile devices. Beginning with the first commercial Android device in September 2008, the operating system has gone through multiple major releases, with the current version being 7.0 "Nougat", released in August 2016. Android applications ("apps") can be downloaded from the Google Play store, which features over 2.7 million apps as of February 2017. Android has been the best-selling OS on tablets since 2013 and runs on the vast majority around 70% of smart phones. As of May 2017, Android has two billion monthly active users and it has the largest installed base of any operating system.

Android's source code is released by Google under an open source license although most Android devices ultimately ship with a combination of free and open source and proprietary software including proprietary software required for accessing Google services. Android is popular with technology companies that require a ready-made, low-cost and customizable operating system for high-tech devices. Its open nature has encouraged a large community of developers and enthusiasts to use the open-source code as a foundation for community-driven projects, which deliver updates to older devices, add new features for advanced users or bring Android to devices originally shipped with other operating systems. The extensive variation of hardware in Android devices causes significant delays for software upgrades, with new versions of the operating system and security patches typically taking months before reaching consumers, or sometimes not at all. The success of Android has made it a target for patent and copyright litigation between technology companies. [1]

My initiatives as a Computer Science and Engineering student are to make an android based app which will access by the registered user. I choose android to build my app because this app requires java programming knowledge and I am capable of doing java programming which is invoke me to take the opportunity to make an app for android users. I implemented a mobile version app with portrait screen resolutions. So in this report I would like to talk about this app itself, the motivation of the app, the design and implementation of the app. Finally I will conclude by presenting my app on real device and describing my future works about this app.

1.2 . Background

This is the time of information and communication technology. So, I am inspired by the current trend of the popular platform. I can build up a website services but I think that developing an app is more interesting than developing the web services. The market value of an app development is also very high. So, android app is my only focus point of view when I intending to make something for my project work.

1.3 Objectives

- Build an app myself on android platform.
- Explain the differences between android and other mobile development environment.
- Understand how Android application work, life cycle, manifest, intents and using external resources.
- Design and develop useful android applications with compelling user friendly interfaces by using, extending and creating my own layouts and views.
- Discuss the process how to create an android operating system supporting app.
- Comparison with existing business card
- Develop a user friendly business card app which will help the professional user.

1.4 Motivation

This project is based on to use a card to avoid carrying able card. After a while a carrying able card might be damaged. So, we can use an electronic card to avoid this problem. Basically I motivated to create a business card using android platform for this reason. Having been exposed and have had a good and long experienced hand IT expertise in my professional career line up, I have been also motivated towards the development of an android app of my own and learn how to develop user friendly app using the next generation technology. I am searching my scope on the job market as an android developer and contribute to the developer community. That is why I made this app for android user.

1.5 Contributions

To develop the app all resources are collected. Software's are from Google, hardware from mobile device producer which is described in this part to initialize an android development. I use some software to develop this app such as android studio, netbeans and xampp.

1.6 Project Report Organization

Chapter 2

In this part, I will discuss about the basic information of android, structure, history and foundation of the android platform. It also describes all the related literature, article and previous work, I used in my application to develop my app.

Chapter 3

In this part, I will describe the implantation procedure for the setting up the android environment, installing android development tools (ADT). There is no framework used for this app. So, all the code is written in pure java language. I discuss all the method I used in the app. I also discuss how they look and how they implemented step by step.

Chapter 4

In this part, I added the conclusion and my future plan of the app. After that there is reference section.

Chapter 2: Overview of Android Platform

2.1 What is Android?

Android is a mobile operating system (OS) currently developed by Google based on the Linux kernel and designed primarily for touch screen mobile devices such as smart phones and tablets. Android's user interface is mainly based on direct manipulation, using touch gestures that loosely correspond to real-world actions such as swiping, tapping and pinching, to manipulate on-screen objects, along with a virtual keyboard for text input. In addition to touch screen devices, 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 notebooks, game consoles, digital cameras, and other electronics.

Initially developed by Android Inc., which Google bought in 2005, Android was unveiled in 2007, along with the founding of the Open Handset Alliance – a consortium of hardware, software, and telecommunication companies devoted to advancing open standards for mobile devices. Beginning with the first commercial Android device in September 2008, the operating system has gone through multiple major releases, with the current version being 7.0 "Nougat", released in August 2016. Android applications ("apps") can be downloaded from the Google Play store, which features over 2.7 million apps as of February 2017. Android has been the best-selling OS on tablets since 2013 and runs on the vast majority around 70% of smart phones. As of May 2017, Android has two billion monthly active users and it has the largest installed base of any operating system.

Android's source code is released by Google under an open source license although most Android devices ultimately ship with a combination of free and open source and proprietary software including proprietary software required for accessing Google services. Android is popular with technology companies that require a ready-made, low-cost and customizable operating system for high-tech devices. Its open nature has encouraged a large community of developers and enthusiasts to use the open-source code as a foundation for community-driven projects, which deliver updates to older devices, add new features for advanced users or bring Android to devices originally shipped with other operating systems. The extensive variation of hardware in Android devices causes significant delays for software upgrades, with new versions of the operating system and security patches typically taking months before reaching consumers, or sometimes not at all. The success of Android has made it a target for patent and copyright litigation between technology companies. [1]

2.2 History of Android

Android Inc. was founded in Palo Alto, California in October 2003 by Andy Rubin, Rich Miner, Nick Sears, and Chris White. Rubin described the Android project as "tremendous potential in developing smarter mobile devices that are more aware of its owner's location and preferences". The early intentions of the company were to develop an advanced operating system for digital cameras, and this was the basis of its pitch to investors in April 2004. The company then decided that the market for cameras was not large enough for its goals, and by five months later it had diverted its efforts and was pitching Android as a handset operating system that would rival Symbian and Microsoft Windows Mobile.

Rubin had difficulty attracting investors early on, and Android was facing eviction from its office space. Steve Perlman, a close friend of Rubin, brought him $10,000 in cash in an envelope, and shortly thereafter wired an undisclosed amount as seed funding. Perlman refused a stake in the company, and has stated "I did it because I believed in the thing, and I wanted to help Andy."

In July 2005, Google acquired Android Inc. for at least $50 million. Its key employees, including Rubin, Miner and White, joined Google as part of the acquisition. Not much was known about the secretive Android at the time, with the company having provided few details other than that it was making software for mobile phones. At Google, the team led by Rubin developed a mobile device platform powered by the Linux kernel. Google marketed the platform to handset makers and carriers on the promise of providing a flexible, upgradeable system. Google had "lined up a series of hardware components and software partners and signaled to carriers that it was open to various degrees of cooperation".

Speculation about Google's intention to enter the mobile communications market continued to build through December 2006. An early prototype had a close resemblance to a BlackBerry phone, with no touch screen and a physical QWERTY keyboard, but the arrival of 2007's Apple iPhone meant that Android "had to go back to the drawing board". Google later changed its Android specification documents to state that "Touch screens will be supported", although "the Product was designed with the presence of discrete physical buttons as an assumption, therefore a touch screen cannot completely replace physical buttons". In September 2007, *InformationWeek* covered an Evalueserve study reporting that Google had filed several patent applications in the area of mobile telephony.

On November 5, 2007, the Open Handset Alliance, a consortium of technology companies including Google, device manufacturers such as HTC, Motorola and Samsung, wireless carriers such as Sprint and T-Mobile, and chipset makers such as Qualcomm and Texas Instruments, unveiled itself, with a goal to develop "the first truly open and comprehensive platform for mobile devices". The first commercially available smartphone running Android was the HTC Dream, also known as T-Mobile G1, announced on September 23, 2008.

Since 2008, Android has seen numerous updates which have incrementally improved the operating system, adding new features and fixing bugs in previous releases. Each major release is named in alphabetical order after a dessert or sugary treat, with the first few Android versions being called "Cupcake", "Donut", "Eclair", and "Froyo", respectively. During its announcement of Android KitKat in 2013, Google explained that "Since these devices make our lives so sweet, each Android version is named after a dessert", although a Google spokesperson told CNN in an interview that "It's kind of like an internal team thing, and we prefer to be a little bit — how should I say — a bit inscrutable in the matter, I'll say".

In 2010, Google launched its Nexus series of devices, a lineup in which Google partnered with different device manufacturers to produce new devices and introduce new Android versions. The series was described as having "played a pivotal role in Android's history by introducing new software iterations and hardware standards across the board", and became known for its "bloat-free" software with "timely [...] updates". At its developer conference in May 2013, Google announced a special version of the Samsung Galaxy S4, where, instead of using Samsung's own Android customization, the phone ran "stock Android" and was promised to receive new system updates fast. The device would become the start of the Google Play edition program, and was followed by other devices, including the HTC One Google Play edition, and Moto G Google Play edition. In 2015, *Ars Technica* wrote that "Earlier this week, the last of the Google Play edition Android phones in Google's online storefront were listed as "no longer available for sale"" and that "Now they're all gone, and it looks a whole lot like the program has wrapped up".

From 2008 to 2013, Hugo Barra served as product spokesperson, representing Android at press conferences and Google I/O, Google's annual developer-focused conference. He left Google in August 2013 to join Chinese phone maker Xiaomi. Less than six months earlier, Google's then-CEO Larry Page announced in a blog post that Andy Rubin had moved from the Android division to take on new projects at Google, and that Sundar Pichai would become the new Android lead. Pichai himself would eventually switch positions, becoming the new CEO of Google in August 2015 following the company's restructure into the Alphabet conglomerate, making Hiroshi Lockheimer the new head of Android.

In June 2014, Google announced Android One, a set of "hardware reference models" that would "allow [device makers] to easily create high-quality phones at low costs", designed for consumers in developing countries. In September, Google announced the first set of Android One phones for release in India. However, *Recode* reported in June 2015 that the project was "a disappointment", citing "reluctant consumers and manufacturing partners" and "misfires from the search company that has never quite cracked hardware". Plans to relaunch Android One surfaced in August 2015, with Africa announced as the next location for the program a week later. A report from *The Information* in January 2017 stated that Google was "expanding its "Android One" program for low-cost smart phones to the U.S. in coming months".

Google introduced the Pixel and Pixel XL smart phones in October 2016, marketed as being the first phones made by Google, and exclusively featured certain software features, such as the Google Assistant, before wider rollout. The Pixel phones replaced the Nexus series, and Rick Osterloh, Google's senior vice president of hardware, confirmed in March 2017 that a successor to the Pixel is coming later in 2017. [1]

2.3 Features of Android

The few main features of android are given below:

2.3.1 Interface

Android's default user interface is mainly based on direct manipulation, using touch inputs that loosely correspond to real-world actions, like swiping, tapping, pinching, and reverse pinching to manipulate on-screen objects, along with a virtual keyboard. Game controllers and full-size physical keyboards are supported via Bluetooth or USB. The response to user input is designed to be immediate and provides a fluid touch interface, often using the vibration capabilities of the device to provide haptic feedback to the user. Internal hardware, such as accelerometers, gyroscopes and proximity sensors are used by some applications to respond to additional user actions, for example adjusting the screen from portrait to landscape depending on how the device is oriented, or allowing the user to steer a vehicle in a racing game by rotating the device, simulating control of a steering wheel.

Android devices boot to the home screen, the primary navigation and information "hub" on Android devices, analogous to the desktop found on personal computers. Android home screens are typically made up of app icons and widgets; app icons launch the associated app, whereas widgets display live, auto-updating content, such as a weather forecast, the user's email inbox, or a news ticker directly on the home screen. A home screen may be made up of several pages, between which the user can swipe back and forth. Third-party apps available on Google Play and other app stores can extensively re-theme the home screen, and even mimic the look of other operating systems, such as Windows Phone. Most manufacturers customize the look and features of their Android devices to differentiate themselves from their competitors.

An All Apps screen lists all installed applications, with the ability for users to drag an app from the list onto the home screen. A recent screen lets users switch between recently used apps. [1]

2.3.2 Applications

Applications ("apps"), which extend the functionality of devices, are written using the Android software development kit (SDK) and, often, the Java programming language. Java may be combined with C/C++, together with a choice of non-default runtimes that allow better C++ support. The Go programming language is also supported, although with a limited set of application programming interfaces (API). In May 2017, Google announced support for Android app development in the Kotlin programming language.

The SDK includes a comprehensive set of development tools, including a debugger, software libraries, a handset emulator based on QEMU, documentation, sample code, and tutorials. Initially, Google's supported integrated development environment (IDE) was Eclipse using the Android Development Tools (ADT) plug-in; in December 2014, Google released Android Studio, based on IntelliJ IDEA, as its primary IDE for Android application development. Other development tools are available, including a native development kit (NDK) for applications or extensions in C or C++, Google App Inventor, a visual environment for novice programmers, and various cross platform mobile web applications frameworks. In

January 2014, Google unveiled a framework based on Apache Cordova for porting Chrome HTML 5 web applications to Android, wrapped in a native application shell.

Due to the open nature of Android, a number of third-party application marketplaces also exist for Android, either to provide a substitute for devices that are not allowed to ship with Google Play Store, provide applications that cannot be offered on Google Play Store due to policy violations, or for other reasons. Examples of these third-party stores have included the Amazon Appstore, GetJar, and SlideMe. F-Droid, another alternative marketplace, seeks to only provide applications that are distributed under free and open source licenses.[1]

2.3.3 Memory Management

Since Android devices are usually battery-powered, Android is designed to manage processes to keep power consumption at a minimum. When an application is not in use the system suspends its operation so that, while available for immediate use rather than closed, it does not use battery power or CPU resources. Android manages the applications stored in memory automatically: when memory is low, the system will begin invisibly and automatically closing inactive processes, starting with those that have been inactive for longest. Lifehacker reported in 2011 that third-party task killers were doing more harm than good. [1]

2.3.4 Virtual Reality

At Google I/O on May 2016, Google announced Daydream, a virtual reality platform that relies on a smart phone and provides VR capabilities through a virtual reality headset and controller designed by Google itself. The platform is built into android starting with Android Nougat, differentiating from standalone support for VR capabilities. The software is available for developers, and was released in 2016. [1]

2.4 Hardware of Android

The main hardware platform for Android is the ARM (ARMv7 and ARMv8-A architectures), with x86, MIPS and MIPS64, and x86-64architectures also officially supported in later versions of Android. The unofficial Android-x86 project provided support for the x86 architectures ahead of the official support. MIPS architecture was also supported before Google did. Since 2012, Android devices with Intel processors began to appear, including phones and tablets. While gaining support for 64-bit platforms, Android was first made to run on 64-bit x86 and then on ARM64. Since Android 5.0 "Lollipop", 64-bit variants of all platforms are supported in addition to the 32-bit variants.

Requirements for the minimum amount of RAM for devices running Android 5.1 range from 512 MB of RAM for normal-density screens, to about 1.8 GB for high-density screens. The recommendation for Android 4.4 is to have at least 512 MB of RAM, while for "low RAM" devices 340 MB is the required minimum amount that does not include memory dedicated to various hardware components such as the baseband processor. Android 4.4 requires a 32-bit ARMv7, MIPS or x86 architecture processor (latter two through unofficial ports), together with an OpenGL ES 2.0 compatible graphics processing unit (GPU). Android supports OpenGL ES 1.1, 2.0, 3.0, 3.1 and as of latest major version, 3.2 and Vulkan. Some applications may explicitly require a certain version of the OpenGL ES, and suitable GPU hardware is required to run such applications.

In addition to running on smart phones and tablets, several vendors run Android natively on regular PC hardware with a keyboard and mouse. In addition to their availability on commercially available hardware, similar PC hardware-friendly versions of Android are freely available from the Android-x 86 projects, including customized Android 4.4. Using the Android emulator that is part of the Android SDK, or third-party emulators, Android can also run non-natively on x86 architectures. Chinese companies are building a PC and mobile operating system, based on Android, to "compete directly with Microsoft Windows and Google Android". The Chinese Academy of Engineering noted that "more than a dozen" companies were customizing Android following a Chinese ban on the use of Windows 8 on government PCs. [1]

2.5 Development of Android

Android is developed by Google until the latest changes and updates are ready to be released, at which point the source code is made available to the Android Open Source Project. This source code can be found without modification on select devices, mainly the Nexusseries of devices. The source code is, in turn, adapted by original equipment manufacturers (OEMs) to run on their hardware. Android's source code does not contain the often proprietary device drivers that are needed for certain hardware components.

In 2007, the green Android logo was designed for Google by graphic designer Irina Blok. The design team was tasked with a project to create a universally identifiable icon with the specific inclusion of a robot in the final design. After numerous design developments based on science fiction and space movies, the team eventually sought inspiration from the human symbol on restroom doors and modified the figure into a robot shape. As Android is open-source, it was agreed that the logo should be likewise, and since its launch the green logo has been reinterpreted into countless variations on the original design. [1]

2.5.1 Update Schedule

Google announces major incremental upgrades to Android on a yearly basis. The updates can be installed on devices over-the-air. The latest major release is 7.0 "Nougat", announced in March 2016 and released the following August.

In 2012, Google began decoupling certain aspects of the operating system (particularly its core applications) so they could be updated through the Google Play store independently of the OS. One of those components, Google Play Services, is a closed-source system-level process providing APIs for Google services, installed automatically on nearly all devices running Android 2.2 "Froyo" and higher. With these changes, Google can add new system functionality through Play Services and update apps without having to distribute an upgrade to the operating system itself.[144] As a result, Android 4.2 and 4.3 "Jelly Bean" contained relatively fewer user-facing changes, focusing more on minor changes and platform improvements.

In May 2016, *Bloomberg* reported that Google was making efforts to keep Android more up-to-date, including accelerated rates of security updates, rolling out technological workarounds, reducing requirements for phone testing, and ranking phone makers in an attempt to "shame" them into better behavior. As stated by *Bloomberg*: "As smart phones get more capable, complex and hack able, having the latest software work closely with the hardware is increasingly important". Hiroshi Lockheimer, the Android lead, admitted that "It’s not an ideal situation", further commenting that the lack of updates is "the weakest link on security on Android". Wireless carriers were described in the report as the "most challenging discussions", due to carriers' slow approval time due to testing on their networks, despite some carriers, including Verizon and Sprint, having already shortened their respective approval times. HTC's then-executive Jason Mackenzie called monthly security updates "unrealistic" in 2015, and Google was trying to persuade carriers to exclude security patches from the full testing procedures. In a further effort for persuasion, Google shared a list of top phone makers measured by updated devices with its Android partners, and is considering making the list public. Mike Chan, co-founder of phone maker Nextbit and former Android developer, said that "The best way to solve this problem is a massive re-architecture of the operating system", "or Google could invest in training manufacturers and carriers "to be good Android citizens"". [1]

2.5.2 Linux Kernel

Android's kernel is based on one of the Linux kernel's long-term support (LTS) branches. As of 2017, Android devices mainly use versions 3.18 or 4.4 of the Linux kernel. The actual kernel depends on the individual device.

In August 2011, Linus Torvalds said that "eventually Android and Linux would come back to a common kernel, but it will probably not be for four to five years". In December 2011, Greg Kroah-Hartman announced the start of Android Mainlining Project, which aims to put some Android drivers, patches and features back into the Linux kernel, starting in Linux 3.3. Linux included the autosleep and wakelocks capabilities in the 3.5 kernel, after many previous attempts at merger. The interfaces are the same but the upstream Linux implementation allows for two different suspend modes: to memory (the traditional suspend that Android uses), and to disk (hibernate, as it is known on the desktop). Google maintains a public code repository that contains their experimental work to re-base Android off the latest stable Linux versions.

The flash storage on Android devices is split into several partitions, such as /system for the operating system itself, and /data for user data and application installations. In contrast to desktop Linux distributions, Android device owners are not given root access to the operating system and sensitive partitions such as /system are read-only. However, root access can be obtained by exploiting security flaws in Android, which is used frequently by the open-source community to enhance the capabilities of their devices, but also by malicious parties to install viruses and malware. [1]

2.5.3 Software Stack

On top of the Linux kernel, there are the middleware, libraries and APIs written in C, and application software running on an application framework which includes Java-compatible libraries. Development of the Linux kernel continues independently of other Android's source code bases.

Until version 5.0, Android used Dalvik as a process virtual machine with trace-based just-in-time (JIT) compilation to run Dalvik "dex-code" (Dalvik Executable), which is usually translated from the Java bytecode. Following the trace-based JIT principle, in addition to interpreting the majority of application code, Dalvik performs the compilation and native execution of select frequently executed code segments ("traces") each time an application is launched. Android 4.4 introduced Android Runtime (ART) as a new runtime environment, which uses ahead-of-time (AOT) compilation to entirely compile the application bytecode into machine code upon the installation of an application. In Android 4.4, ART was an experimental feature and not enabled by default; it became the only runtime option in the next major version of Android, 5.0.

For its Java library, the Android platform uses a subset of the now discontinued Apache Harmony project. In December 2015, Google announced that the next version of Android would switch to a Java implementation based on OpenJDK.

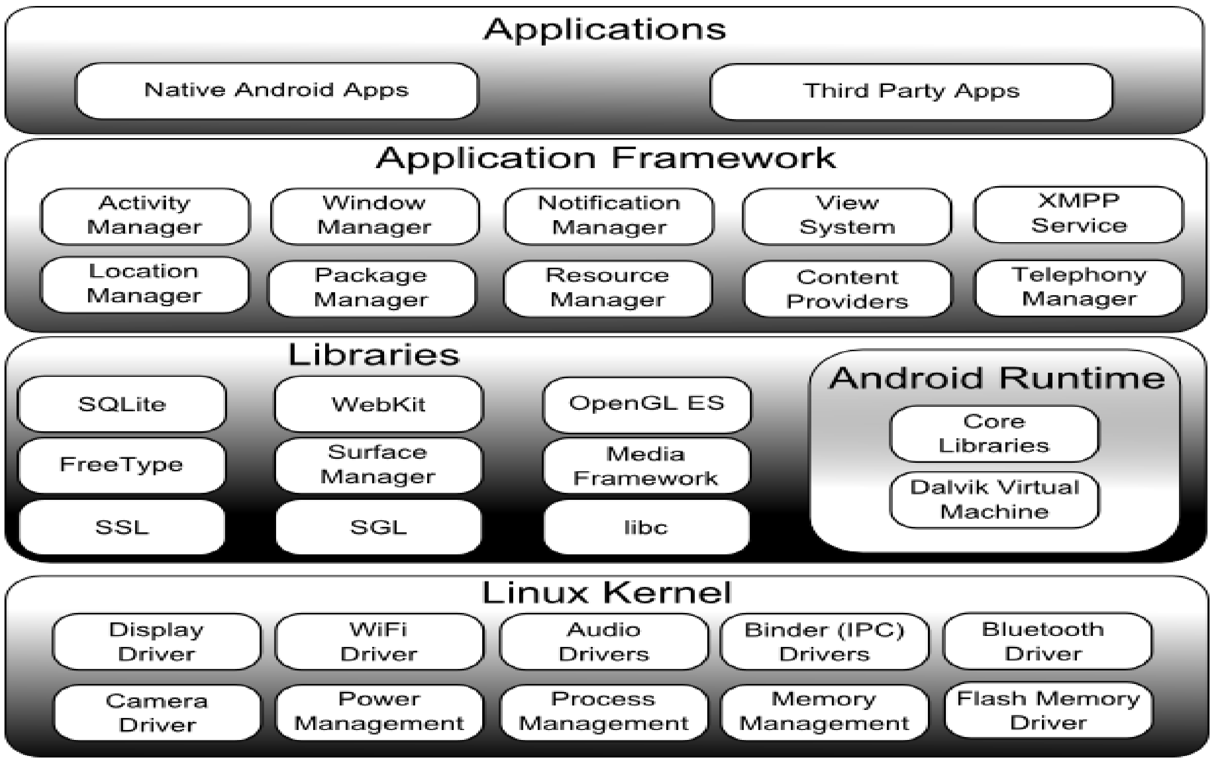


Figure 01: Android Architecture Diagram

Android's standard C library, Bionic, was developed by Google specifically for Android, as a derivation of the BSD's standard C library code. Bionic itself has been designed with several major features specific to the Linux kernel. The main benefits of using Bionic instead of the GNU C Library(glibc) or uClibc are its smaller runtime footprint, and optimization for low-frequency CPUs. At the same time, Bionic is licensed under the terms of the BSD licence, which Google finds more suitable for the Android's overall licensing model.

Android has another operating system, Trusty OS, within it, as a part of "Trusty" "software components supporting a Trusted Execution Environment (TEE) on mobile devices." "Trusty and the Trusty API are subject to change. [..] Applications for the Trusty OS can be written in C/C++ (C++ support is limited), and they have access to a small C library. [..] All Trusty applications are single-threaded; multithreading in Trusty user space currently is unsupported. [..] Third-party application development is not supported in" the current version, and software running on the OS and processor for it, run the "DRM framework for protected content. [..] There are many other uses for a TEE such as mobile payments, secure banking, full-disk encryption, multi-factor authentication, device reset protection, replay-protected persistent storage, wireless display ("cast") of protected content, secure PIN and fingerprint processing, and even malware detection." [1]

2.5.4 Open Source Community

Android has an active community of developers and enthusiasts who use the *Android Open Source Project* (AOSP) source code to develop and distribute their own modified versions of the operating system. These community-developed releases often bring new features and updates to devices faster than through the official manufacturer/carrier channels, with a comparable level of quality; provide continued support for older devices that no longer receive official updates; or bring Android to devices that were officially released running other operating systems, such as the HP Touch Pad. Community releases often come pre-rooted and contain modifications not provided by the original vendor, such as the ability to over clock or over/under volt the device's processor. CyanogenMod was the most widely used community firmware, and CyanogenMod has been discontinued and Lineage OS is the successor of CyanogenMod.

Historically, device manufacturers and mobile carriers have typically been unsupportive of third-party firmware development. Manufacturers express concern about improper functioning of devices running unofficial software and the support costs resulting from this. Moreover, modified firmwares such as CyanogenMod sometimes offer features, such as tethering, for which carriers would otherwise charge a premium. As a result, technical obstacles including locked boot loaders and restricted access to root permissions are common in many devices. However, as community-developed software has grown more popular, and following a statement by the Librarian of Congress in the United States that permits the "jailbreaking" of mobile devices, manufacturers and carriers have softened their position regarding third party development, with some, including HTC, Motorola, Samsung and Sony, providing support and encouraging development. As a result of this, over time the need to circumvent hardware restrictions to install unofficial firmware has lessened as an increasing number of devices are shipped with unlocked or unlock able boot loaders, similar to Nexus series of phones, although usually requiring that users waive their devices' warranties to do so. However, despite manufacturer acceptance, some carriers in the US still require that phones are locked down, frustrating developers and customers. [1]

2.6 Security and Privacy

These are few security and privacy issues are given below:

2.6.1 Scope of Surveillance by Public Institutions

As part of the broader 2013 mass surveillance disclosures it was revealed in September 2013 that the American and British intelligence agencies, the National Security Agency(NSA) and Government Communications Headquarters (GCHQ), respectively, have access to the user data on iPhone, BlackBerry, and Android devices. They are reportedly able to read almost all smart phone information, including SMS, location, emails, and notes. In January 2014, further reports revealed the intelligence agencies' capabilities to intercept the personal information transmitted across the Internet by social networks and other popular applications such as *Angry Birds*, which collect personal information of their users for advertising and other commercial reasons. GCHQ has, according to *The Guardian*, a wiki-style guide of different apps and advertising networks, and the different data that can be siphoned from each. Later that week, the Finnish Angry Birds developer Rovio announced that it was reconsidering its relationships with its advertising platforms in the light of these revelations, and called upon the wider industry to do the same.

The documents revealed a further effort by the intelligence agencies to intercept Google Maps searches and queries submitted from Android and other smart phones to collect location information in bulk. The NSA and GCHQ insist their activities are in compliance with all relevant domestic and international laws, although the Guardian stated "the latest disclosures could also add to mounting public concern about how the technology sector collects and uses information, especially for those outside the US, who enjoy fewer privacy protections than Americans." [1]

2.6.2 Common Security Threats

Research from security company Trend Micro lists premium service abuse as the most common type of Android malware, where text messages are sent from infected phones to premium-rate telephone numbers without the consent or even knowledge of the user. Other malware displays unwanted and intrusive advertisements on the device, or sends personal information to unauthorized third parties. Security threats on Android are reportedly growing exponentially; however, Google engineers have argued that the malware and virus threat on Android is being exaggerated by security companies for commercial reasons, and have accused the security industry of playing on fears to sell virus protection software to users. Google maintains that dangerous malware is actually extremely rare, and a survey conducted by F-Secure showed that only 0.5% of Android malware reported had come from the Google Play store.

In a March 2017 post on Google's Security Blog, Android security leads Adrian Ludwig and Mel Miller wrote that "More than 735 million devices from 200+ manufacturers received a platform security update in 2016" and that "Our carrier and hardware partners helped expand deployment of these updates, releasing updates for over half of the top 50 devices worldwide in the last quarter of 2016". They also wrote that "About half of devices in use at the end of 2016 had not received a platform security update in the previous year", stating that their work would continue to focus on streamlining the security updates program for easier deployment by manufacturers. Furthermore, in a comment to *TechCrunch*, Ludwig stated that the wait time for security updates had been reduced from "six to nine weeks down to just a few days", with 78% of flagship devices in North America being up-to-date on security at the end of 2016.

Android smart phones have the ability to report the location of Wi-Fi access points, encountered as phone users move around, to build databases containing the physical locations of hundreds of millions of such access points. These databases form electronic maps to locate smart phones, allowing them to run apps like Foursquare, Google Latitude, Facebook Places, and to deliver location-based ads. Third party monitoring software such as TaintDroid, an academic research-funded project, can, in some cases, detect when personal information is being sent from applications to remote servers. [1]

2.6.3 Technical Security Features

Android applications run in a sandbox, an isolated area of the system that does not have access to the rest of the system's resources, unless access permissions are explicitly granted by the user when the application is installed.

Since February 2012, Google has used its Google Bouncer malware scanner to watch over and scan apps available in the Google Play store. A "Verify Apps" feature was introduced in November 2012, as part of the Android 4.2 "Jelly Bean" operating system version, to scan all apps, both from Google Play and from third-party sources, for malicious behavior. Originally only doing so during installation, Verify Apps received an update in 2014 to "constantly" scan apps, and in 2017 the feature was made visible to users through a menu in Settings.

In September 2014, Jason Nova of *Android Authority* reported on a study by the German security company Fraunhofer AISEC in antivirus software and malware threats on Android. Nova wrote that "The Android operating system deals with software packages by sandboxing them; this does not allow applications to list the directory contents of other apps to keep the system safe. By not allowing the antivirus to list the directories of other apps after installation, applications that show no inherent suspicious behavior when downloaded are cleared as safe. If then later on parts of the app are activated that turn out to be malicious, the antivirus will have no way to know since it is inside the app and out of the antivirus' jurisdiction". The study by Fraunhofer AISEC, examining antivirus software from Avast, AVG, Bitdefender, ESET, F-Secure, Kaspersky, Lookout, McAfee (formerly Intel Security), Norton, Sophos, and Trend Micro, revealed that "the tested antivirus apps do not provide protection against customized malware or targeted attacks", and that "the tested antivirus apps were also not able to detect malware which is completely unknown to date but does not make any efforts to hide its malignity".

In August 2013, Google announced Android Device Manager (renamed Find My Device in May 2017), a service that allows users to remotely track, locate, and wipe their Android device, with an Android app for the service released in December. In December 2016, Google introduced a Trusted Contacts app, letting users request location-tracking of loved ones during emergencies. [1]

2.7 Licensing

The source code for Android is open-source: it is developed in private by Google, with the source code released publicly when a new version of Android is released. Google publishes most of the code (including network and telephony stacks) under the non-copyleft Apache License version 2.0. which allows modification and redistribution. The license does not grant rights to the "Android" trademark, so device manufacturers and wireless carriers have to license it from Google under individual contracts. Associated Linux kernel changes are released under the copyleft GNU General Public License version 2, developed by the Open Handset Alliance, with the source code publicly available at all times. Typically, Google collaborates with a hardware manufacturer to produce a flagship device (part of the Nexus series) featuring the new version of Android, then makes the source code available after that device has been released. The only Android release which was not immediately made available as source code was the tablet-only 3.0 *Honeycomb*release. The reason, according to Andy Rubin in an official Android blog post, was because *Honeycomb* was rushed for production of the Motorola Xoom, and they did not want third parties creating a "really bad user experience" by attempting to put onto smartphones a version of Android intended for tablets.

Some stock applications in AOSP code that were formerly used by earlier versions of Android, such as Search, Music, and Calendar, have been abandoned by Google in favor of non-free replacements distributed through Play Store (Google Search, Google Play Music, and Google Calendar) that are no longer open-source. Moreover, open-source variants of some applications also exclude functions that are present in their non-free versions, such as Photosphere panoramas in Camera, and a Google Now page on the default home screen (exclusive to the proprietary version "Google Now Launcher", whose code is embedded within that of the main Google application). [1]

2.8 Leverage over Manufacturers

Google licenses their Google Mobile Services software, along with Android trademarks, only to hardware manufacturers for devices that meet Google's compatibility standards specified in the Android Compatibility Program document. Thus, forks of Android that make major changes to the operating system itself do not include any of Google's non-free components, stay incompatible with applications that require them, and must ship with an alternative software marketplace in lieu of Google Play Store. Examples of such Android forks are Amazon's Fire OS (which is used on the Kindle Fire line of tablets, and oriented toward Amazon services), the Nokia X Software Platform (a fork used by the Nokia X family, oriented primarily toward Nokia and Microsoft services), and other forks that exclude Google apps due to the general unavailability of Google services in certain regions (such as China). In 2014, Google also began to require that all Android devices which license the Google Mobile Services software display a prominent "Powered by Android" logo on their boot screens.

Members of the Open Handset Alliance, which include the majority of Android OEMs, are also contractually forbidden from producing Android devices based on forks of the OS; in 2012, Acer Inc. was forced by Google to halt production on a device powered by Alibaba Group's Aliyun OS with threats of removal from the OHA, as Google deemed the platform to be an incompatible version of Android. Alibaba Group defended the allegations, arguing that the OS was a distinct platform from Android (primarily using HTML5apps), but incorporated portions of Android's platform to allow backwards compatibility with third-party Android software. Indeed, the devices did ship with an application store which offered Android apps; however, the majority of them were pirated. [1]

2.9 Market Share

Research company Canalys estimated in the second quarter of 2009, that Android had a 2.8% share of worldwide smart phone shipments. By the fourth quarter of 2010, this had grown to 33% of the market becoming the top-selling smart phone platform, overtaking Symbian. By the third quarter of 2011, Gartner estimated that more than half (52.5%) of the smartphone sales belonged to Android. By the third quarter of 2012 Android had a 75% share of the global smartphone market according to the research firm IDC.

In July 2011, Google said that 550,000 Android devices were being activated every day, up from 400,000 per day in May, and more than 100 million devices had been activated with 4.4% growth per week. In September 2012, 500 million devices had been activated with 1.3 million activations per day. In May 2013, at Google I/O, Sundar Pichai announced that 900 million Android devices had been activated.

Android market share varies by location. In July 2012, "mobile subscribers aged 13+" in the United States using Android were up to 52%, and rose to 90% in China. During the third quarter of 2012, Android's worldwide smart phone shipment market share was 75%, with 750 million devices activated in total. In April 2013 Android had 1.5 million activations per day. As of May 2013, 48 billion applications ("apps") have been installed from the Google Play store, and by September 2013, one billion Android devices have been activated.

As of February 2017, the Google Play store has over 2.7 million Android applications published, and As of May 2016, apps have been downloaded more than 65 billion times. The operating system's success has made it a target for patent litigation as part of the so-called "smart phone wars" between technology companies.

While Android phones in the Western world commonly include Google's proprietary add-ons (such as Google Play) to the otherwise open-source operating system, this is increasingly not the case in emerging markets; "ABI Research claims that 65 million devices shipped globally with open-source Android in the second quarter of [2014], up from 54 million in the first quarter"; depending on country, percent of phones estimated to be based only on Android's source code (AOSP), forgoing the Android trademark: Thailand (44%), Philippines (38%), Indonesia (31%), India (21%), Malaysia (24%), Mexico (18%), Brazil (9%).

By August 2016, the two biggest continents have gone mobile-majority, judged by web use ("desktop" has 46.92%–55.16% use worldwide, depending on day of the week, making some weeks desktop-minority; lowest full month was at 50.05%); because of Android (see usage share of operating systems), that has majority use on smart phones in virtually all countries (all continents have gone Android-majority, including North America[315][316] except for Oceania, because of Australia), with few exceptions (all of which have iOS-majority); in the US, Android is close to iOS, having exchanged majority position a few times, Canada and the following are also exceptions: Japan, Philippines, Australia and the only exceptions in Europe are the UK, Switzerland, Belgium and the Nordic countries Denmark, Iceland, Sweden and Norway.

By 2016, Android was on the majority of smart phones in virtually all countries in the world, excluding United States and Canada (while including North America continent as a whole), Australia and Japan. A few countries, such as the UK, lose Android-majority if tablets are included.

According to an April 2017 StatCounter report, Android overtook Microsoft Windows to become the most popular operating system for total Internet usage.

In September 2015, Google announced that Android had 1.4 billion monthly active users. This changed to 2 billion monthly active users in May 2017. [1]

2.10 Adoption on Table

Despite its success on smart phones, initially Android tablet adoption was slow. One of the main causes was the egg situation where consumers were hesitant to buy an Android tablet due to a lack of high quality tablet applications, but developers were hesitant to spend time and resources developing tablet applications until there was a significant market for them. The content and app "ecosystem" proved more important than hardware specs as the selling point for tablets. Due to the lack of Android tablet-specific applications in 2011, early Android tablets had to make do with existing smart phone applications that were ill-suited to larger screen sizes, whereas the dominance of Apple's iPad was reinforced by the large number of tablet-specific iOS applications.

Figure 02: The first-generation Nexus 7 tablet, running Android 4.1 Jelly Bean

In March 2016, Galen Gruman of *InfoWorld* stated that Android devices could be a "real part of your business [..] there's no longer a reason to keep Android at arm's length. It can now be as integral to your mobile portfolio as Apple's iOS devices are". A year earlier, Gruman had stated that Microsoft's own mobile Office apps were "better on iOS and Android" than on Microsoft's own Windows 10 devices. [1]

2.11 Platform Usage

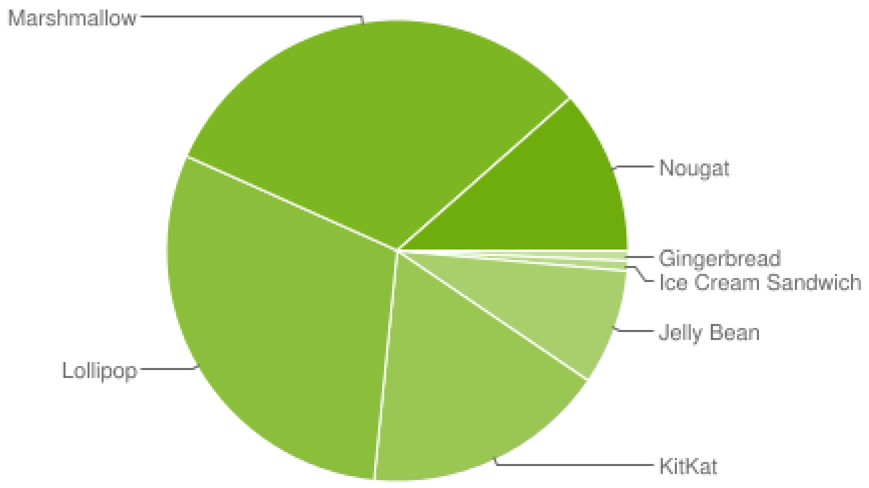


Figure 03: Android Platform Usage

Charts in this section provide breakdowns of Android versions, based on devices accessing the Google Play Store in a seven-day period ending on July 11, 2017.[350][c] Therefore, these statistics exclude devices running various Android forks that do not access the Google Play Store, such as Amazon's Fire tablets. [1]

Version	Code name	Release date	API level	DVM/ART	Distribution	First devices to run version
7.1	Nougat	October 4, 2016	25	ART	0.9%	Pixel, Pixel XL
7.0		August 22, 2016	24	ART	10.6%	Nexus 5X, Nexus 6P

Version	Code name	Release date	API level	DVM/ART	Distribution	First devices to run version
6.0	Marshmallow	October 5, 2015	23	ART	31.8%	
5.1	Lollipop	March 9, 2015	22	ART	22.3%	Android One
5.0		November 3, 2014	21	ART 2.1.0	7.8%	Nexus 6
4.4	KitKat	October 31, 2013	19	DVM (and ART 1.6.0)	17.1%	Nexus 5
4.3	Jelly Bean	July 24, 2013	18	DVM	1.2%	Nexus 7 2013
4.2		November 13, 2012	17	DVM	4.1%	Nexus 4, Nexus 10
4.1		July 9, 2012	16	DVM	2.8%	Nexus 7
4.0	Ice Cream Sandwich	October 19, 2011	15	DVM	0.7%	Galaxy Nexus
2.3	Gingerbread	February 9, 2011	10	DVM 1.4.0	0.7%	Nexus S

As of 2017, more than 60% of devices have OpenGL ES 3.0 or higher. [1]

2.12 Software Development Kit

Android software development is the process by which new applications are created for the Android operating system. Applications are usually developed in Java programming language using the Android software development kit (SDK), but other development environments are also available.

2.13 Android SDK

The Android software development kit (SDK) includes a comprehensive set of development tools. These include a debugger, libraries, a handset emulator based on QEMU, documentation, sample code, and tutorials. Currently supported development platforms include computers running Linux (any modern desktop Linux distribution), Mac OS X 10.5.8 or later, and Windows 7 or later. As of March 2015, the SDK is not available on Android itself, but software development is possible by using specialized Android applications.

Until around the end of 2014, the officially supported integrated development environment (IDE) was Eclipse using the Android Development Tools (ADT) Plugin, though IntelliJ IDEA IDE (all editions) fully supports Android development out of the box, and NetBeans IDE also supports Android development via a plugin. As of 2015, Android Studio, made by Google and powered by IntelliJ, is the official IDE; however, developers are free to use others, but Google made it clear that ADT was officially deprecated since the end of 2015 to focus on Android Studio as the official Android IDE. Additionally, developers may use any text editor to edit Java and XML files, then use command line tools (Java Development Kit and Apache Ant are required) to create, build and debug Android applications as well as control attached Android devices (e.g., triggering a reboot, installing software package(s) remotely).

Enhancements to Android's SDK go hand in hand with the overall Android platform development. The SDK also supports older versions of the Android platform in case developers wish to target their applications at older devices. Development tools are downloadable components, so after one has downloaded the latest version and platform, older platforms and tools can also be downloaded for compatibility testing. [2]

2.14 Android Studio

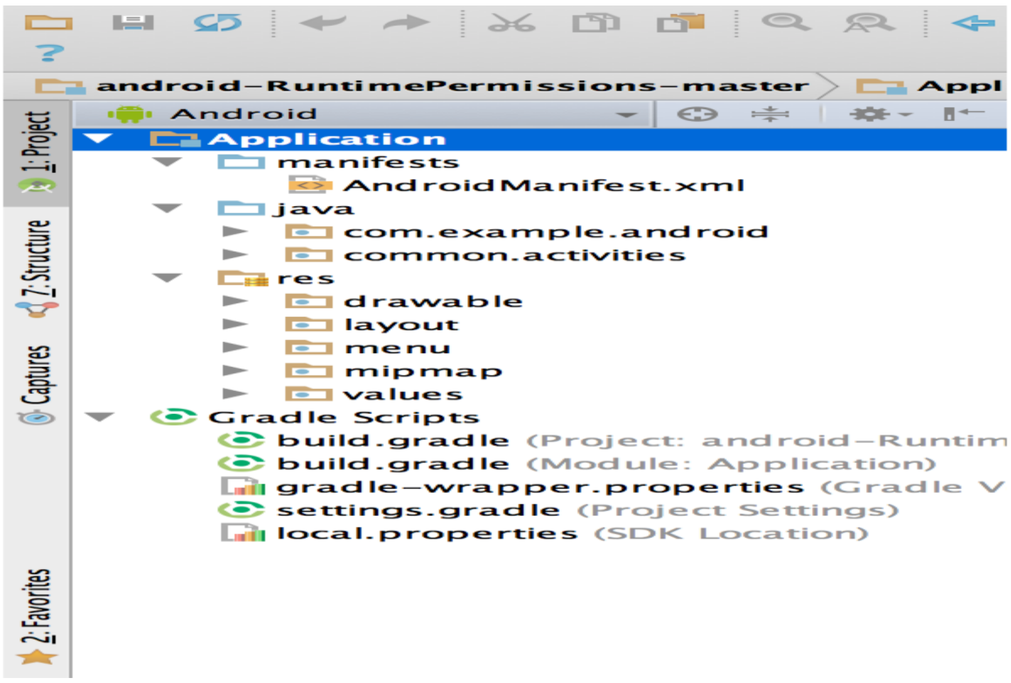
Android Studio is the official Integrated Development Environment (IDE) for Android app development, based on IntelliJ IDEA . On top of IntelliJ's powerful code editor and developer tools, Android Studio offers even more features that enhance your productivity when building Android apps, such as:

- A flexible Gradle-based build system
- A fast and feature-rich emulator
- A unified environment where you can develop for all Android devices
- Instant Run to push changes to your running app without building a new APK
- Code templates and GitHub integration to help you build common app features and import sample code
- Extensive testing tools and frameworks
- Lint tools to catch performance, usability, version compatibility, and other problems
- C++ and NDK support
- Built-in support for Google Cloud Platform, making it easy to integrate Google Cloud Messaging and App Engine [4]

2.15 Project Structure

Each project in Android Studio contains one or more modules with source code files and resource files. Types of modules include:

- Android app modules
- Library modules
- Google App Engine modules



```
Application
    manifests
        AndroidManifest.xml
    java
        com.example.android
        common.activities
    res
        drawable
        layout
        menu
        mipmap
        values
Gradle Scripts
    build.gradle (Project: android-Runtim
    build.gradle (Module: Application)
    gradle-wrapper.properties (Gradle V
    settings.gradle (Project Settings)
    local.properties (SDK Location)
```

Figure 04: The Project Files in Android View

By default, Android Studio displays your project files in the Android project view, as shown in figure 1. This view is organized by modules to provide quick access to your project's key source files.

All the build files are visible at the top level under **Gradle Scripts** and each app module contains the following folders:

- **manifests**: Contains the AndroidManifest.xml file.
- **java**: Contains the Java source code files, including JUnit test code.
- **res**: Contains all non-code resources, such as XML layouts, UI strings, and bitmap images.

The Android project structure on disk differs from this flattened representation. To see the actual file structure of the project, select **Project** from the **Project** dropdown (in figure 1, it's showing as**Android**).

You can also customize the view of the project files to focus on specific aspects of your app development. For example, selecting the **Problems** view of your project displays links to the source files containing any recognized coding and syntax errors, such as a missing XML element closing tag in a layout file. [4]

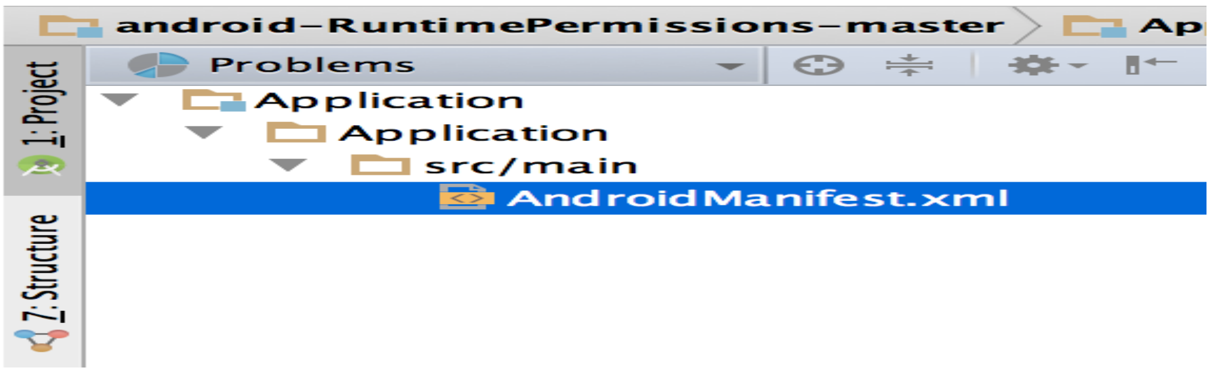
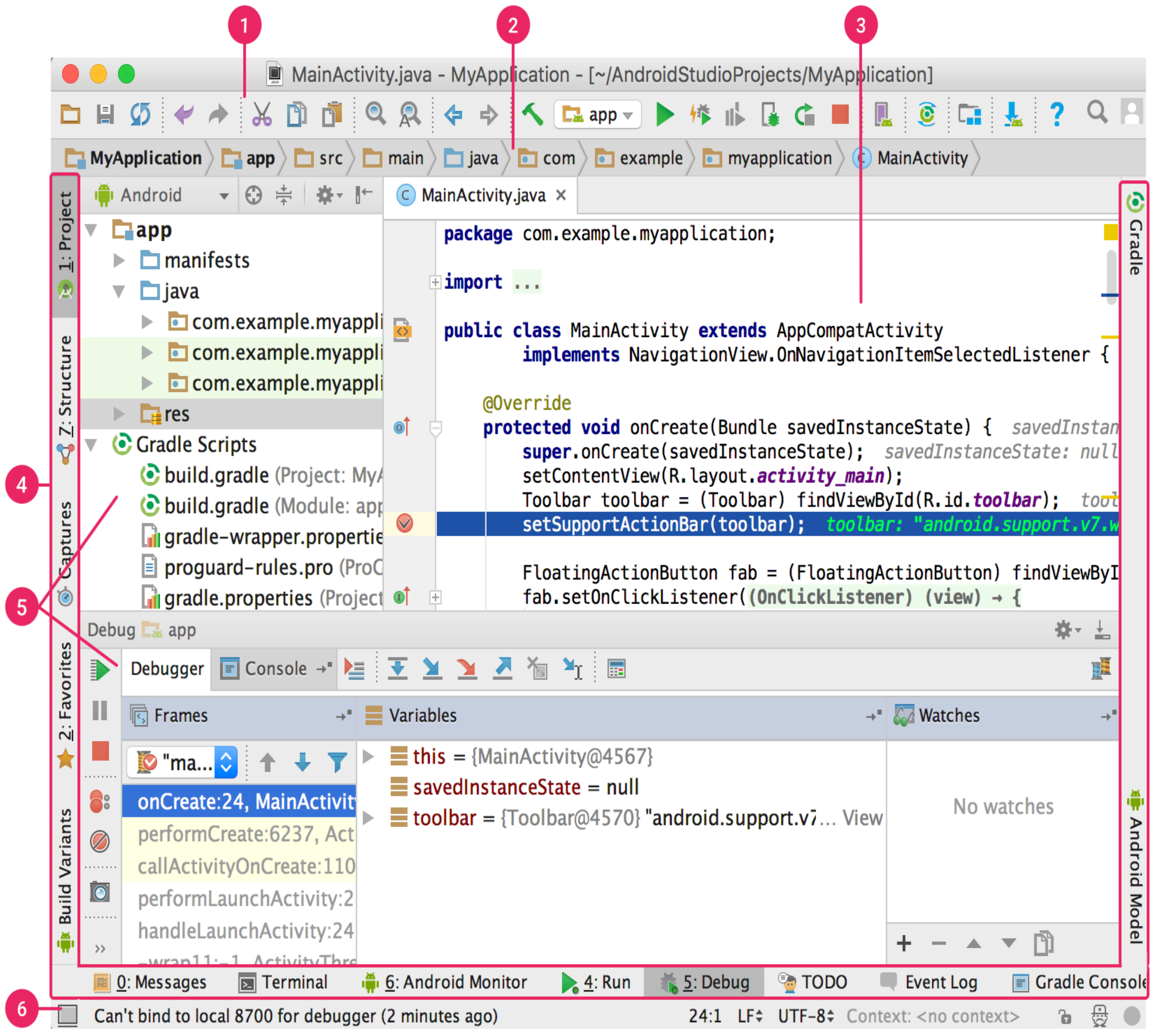


Figure 05: The Project Files in Problems View, Showing a Layout File with a Problem

2.16 The User Interface



```
package com.example.myapplication;

import ...

public class MainActivity extends AppCompatActivity
        implements NavigationView.OnNavigationItemSelectedListener {

    @Override
    protected void onCreate(Bundle savedInstanceState) {  savedInstan
        super.onCreate(savedInstanceState);  savedInstanceState: null
        setContentView(R.layout.activity_main);
        Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);  tool
        setSupportActionBar(toolbar);  toolbar: "android.support.v7.w

        FloatingActionButton fab = (FloatingActionButton) findViewByI
        fab.setOnClickListener((OnClickListener) (view) → {
```

```
this = {MainActivity@4567}
savedInstanceState = null
toolbar = {Toolbar@4570} "android.support.v7... View
```

```
Can't bind to local 8700 for debugger (2 minutes ago)     24:1  LF÷  UTF-8÷  Context: <no context>
```

Figure 06: The Android Studio Main Window

1. The **toolbar** lets you carry out a wide range of actions, including running your app and launching Android tools.
2. The **navigation bar** helps you navigate through your project and open files for editing. It provides a more compact view of the structure visible in the **Project** window.
3. The **editor window** is where you create and modify code. Depending on the current file type, the editor can change. For example, when viewing a layout file, the editor displays the Layout Editor.
4. The **tool window bar** runs around the outside of the IDE window and contains the buttons that allow you to expand or collapse individual tool windows.
5. The **tool windows** give you access to specific tasks like project management, search, version control, and more. You can expand them and collapse them.

6. The **status bar** displays the status of your project and the IDE itself, as well as any warnings or messages. [4]

2.17 Navigation

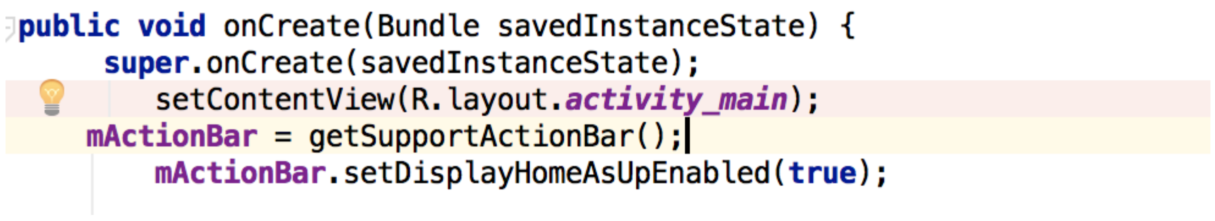
Here are some tips to help you move around Android Studio.

- Switch between your recently accessed files using the *Recent Files* action. Press **Control + E** (**Command + E** on a Mac) to bring up the Recent Files action. By default, the last accessed file is selected. You can also access any tool window through the left column in this action.
- View the structure of the current file using the *File Structure* action. Bring up the File Structure action by pressing **Control + F12** (**Command + F12** on a Mac). Using this action, you can quickly navigate to any part of your current file.
- Search for and navigate to a specific class in your project using the *Navigate to Class* action. Bring up the action by pressing **Control + N** (**Command + O** on a Mac). Navigate to Class supports sophisticated expressions, including camel humps, paths, line navigate to, middle name matching, and many more. If you call it twice in a row, it shows you the results out of the project classes.
- Navigate to a file or folder using the *Navigate to File* action. Bring up the Navigate to File action by pressing **Control + Shift + N** (**Command + Shift + O** on a Mac). To search for folders rather than files, add a / at the end of your expression.
- Navigate to a method or field by name using the *Navigate to Symbol* action. Bring up the Navigate to Symbol action by pressing **Control + Shift + Alt + N** (**Command + Shift + Alt + O** on a Mac).
- Find all the pieces of code referencing the class, method, field, parameter, or statement at the current cursor position by pressing **Alt+F7** [4]

2.18 Style and Formatting

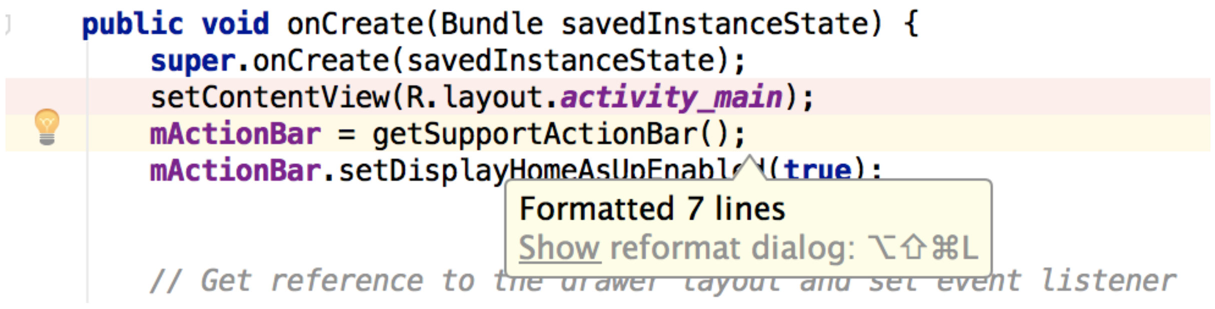
As you edit, Android Studio automatically applies formatting and styles as specified in your code style settings. You can customize the code style settings by programming language, including specifying conventions for tabs and indents, spaces, wrapping and braces, and blank lines. To customize your code style settings, click **File** > **Settings** > **Editor** > **Code Style** (**Android Studio** > **Preferences** > **Editor** > **Code Style** on a Mac.)

Although the IDE automatically applies formatting as you work, you can also explicitly call the *Reformat Code* action by pressing **Control + Alt + L** (**Opt + Command + L** on a Mac), or auto-indent all lines by pressing **Control + Alt + I** (**Alt + Option + I** on a Mac). [4]



```
public void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    mActionBar = getSupportActionBar();
        mActionBar.setDisplayHomeAsUpEnabled(true);
```

Figure 07: Code before Formatting



```
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        mActionBar = getSupportActionBar();
        mActionBar.setDisplayHomeAsUpEnabl

        // Get reference to the drawer layout and set event listener
```

Formatted 7 lines
Show reformat dialog: ⌥⇧⌘L

Figure 08: Code after Formatting

2.19 Version Control Basic

Android Studio supports a variety of version control systems (VCS's), including Git, GitHub, CVS, Mercurial, Subversion, and Google Cloud Source Repositories.

After importing your app into Android Studio, use the Android Studio VCS menu options to enable VCS support for the desired version control system, create a repository, import the new files into version control, and perform other version control operations:

1. From the Android Studio **VCS** menu, click **Enable Version Control Integration**.
2. From the drop-down menu, select a version control system to associate with the project root, and then click **OK**.

The VCS menu now displays a number of version control options based on the system you selected.

Note: You can also use the **File > Settings > Version Control** menu option to set up and modify the version control settings. [4]

2.20 Gradle Build System

Android Studio uses Gradle as the foundation of the build system, with more Android-specific capabilities provided by the Android plugin for Gradle. This build system runs as an integrated tool from the Android Studio menu, and independently from the command line. You can use the features of the build system to do the following:

- Customize, configure, and extend the build process.
- Create multiple APKs for your app, with different features using the same project and modules.
- Reuse code and resources across sourcesets.

By employing the flexibility of Gradle, you can achieve all of this without modifying your app's core source files. Android Studio build files are namedbuild.gradle. They are plain text files that use Groovy syntax to configure the build with elements provided by the Android plugin for Gradle. Each project has one top-level build file for the entire project and separate module-level build files for each module. When you import an existing project, Android Studio automatically generates the necessary build files. [4]

2.21 Start a New Project

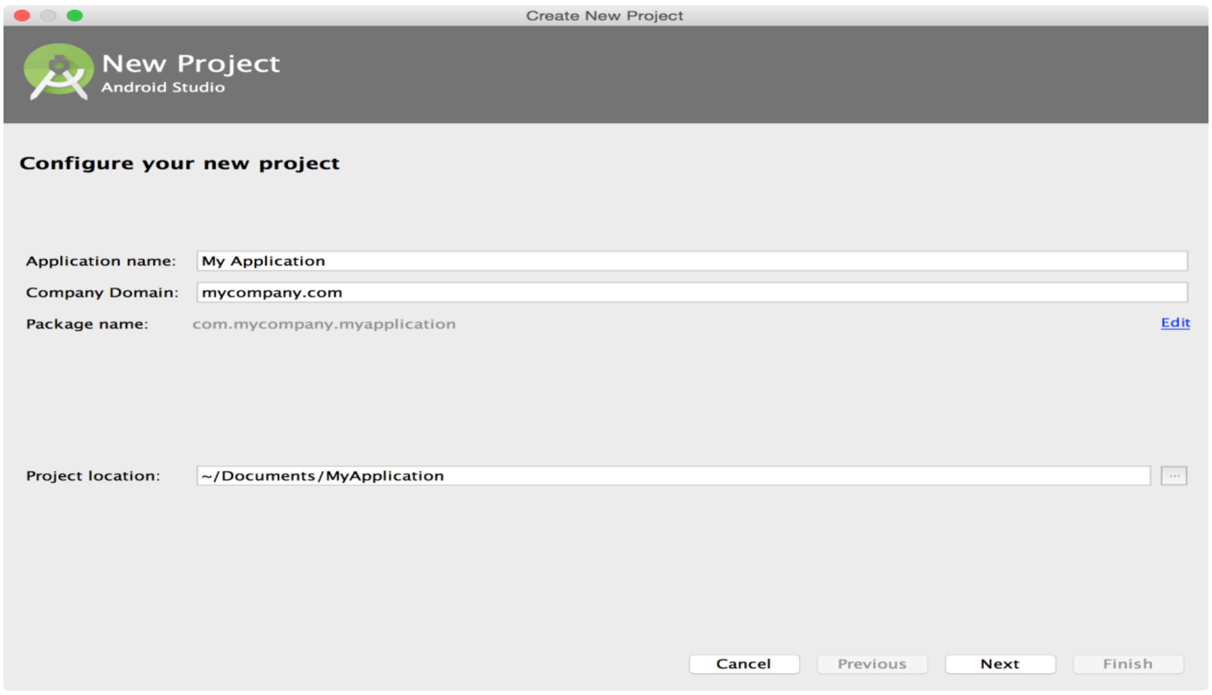
Android Studio makes it easy to create Android apps for various form factors, such as phone, tablet, TV, Wear, and Google Glass. The **New Project** wizard lets you choose the form factors for your app and populates the project structure with everything you need to get started. Use the following steps to create a new project.

2.21.1 Step 1: Start and Configure the Project

If you didn't have a project opened, Android Studio shows the Welcome screen. To create a new project, click **Start a New Android Studio project**.

If you had a project opened, Android Studio shows the development environment. To create a new project, click **File** > **New** > **New Project**.

The next window lets you configure the name of your app, the package name, and the location of your project. [5]


Create New Project

New Project
Android Studio

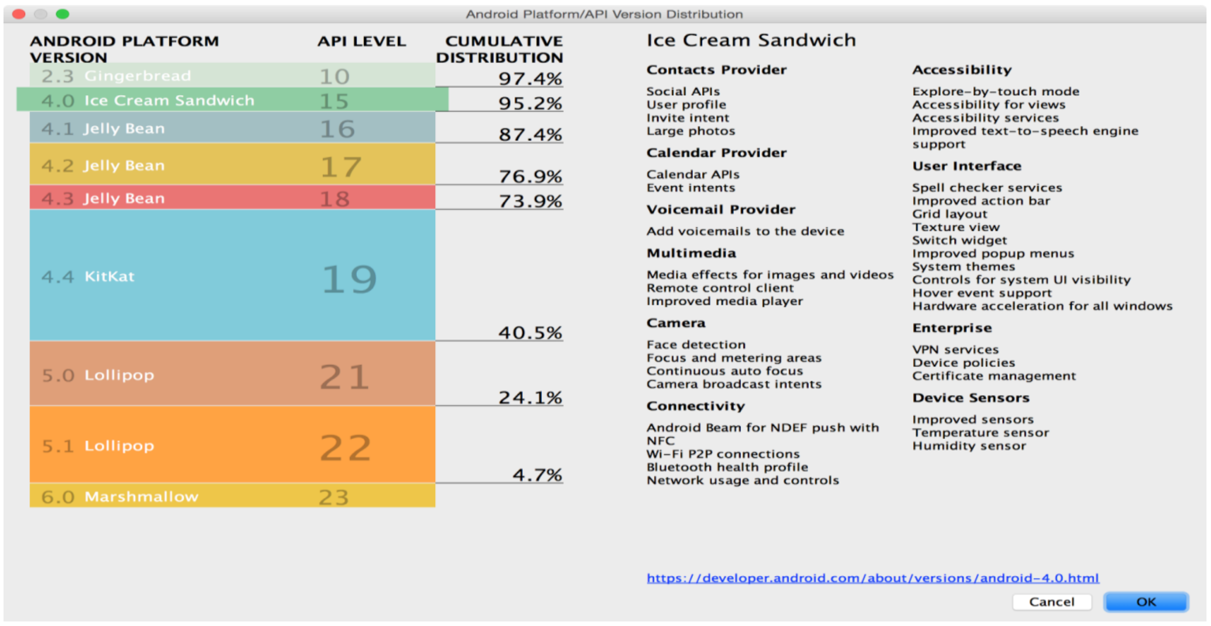
Configure your new project

Application name: My Application
Company Domain: mycompany.com
Package name: com.mycompany.myapplication
Project location: ~/Documents/MyApplication

Figure 09: The Configure Your New Project Screen

2.21.2 Step 2: Select from Factors and API Level

The next window lets you select the form factors supported by your app, such as phone, tablet, TV, Wear, and Google Glass. The selected form factors become the app modules within the project. For each form factor, you can also select the API Level for that app. To get more information, click **Help me choose**. [5]


Android Platform/API Version Distribution

ANDROID PLATFORM VERSION	API LEVEL	CUMULATIVE DISTRIBUTION
2.3 Gingerbread	10	97.4%
4.0 Ice Cream Sandwich	15	95.2%
4.1 Jelly Bean	16	87.4%
4.2 Jelly Bean	17	76.9%
4.3 Jelly Bean	18	73.9%
4.4 KitKat	19	40.5%
5.0 Lollipop	21	24.1%
5.1 Lollipop	22	4.7%
6.0 Marshmallow	23	

Ice Cream Sandwich

Contacts Provider
Social APIs
User profile
Invite intent
Large photos

Calendar Provider
Calendar APIs
Event intents

Voicemail Provider
Add voicemails to the device

Multimedia
Media effects for images and videos
Remote control client
Improved media player

Camera
Face detection
Focus and metering areas
Continuous auto focus
Camera broadcast intents

Connectivity
Android Beam for NDEF push with NFC
Wi-Fi P2P connections
Bluetooth health profile
Network usage and controls

Accessibility
Explore-by-touch mode
Accessibility for views
Accessibility services
Improved text-to-speech engine support

User Interface
Spell checker services
Improved action bar
Grid layout
Texture view
Switch widget
Improved popup menus
System themes
Controls for system UI visibility
Hover event support
Hardware acceleration for all windows

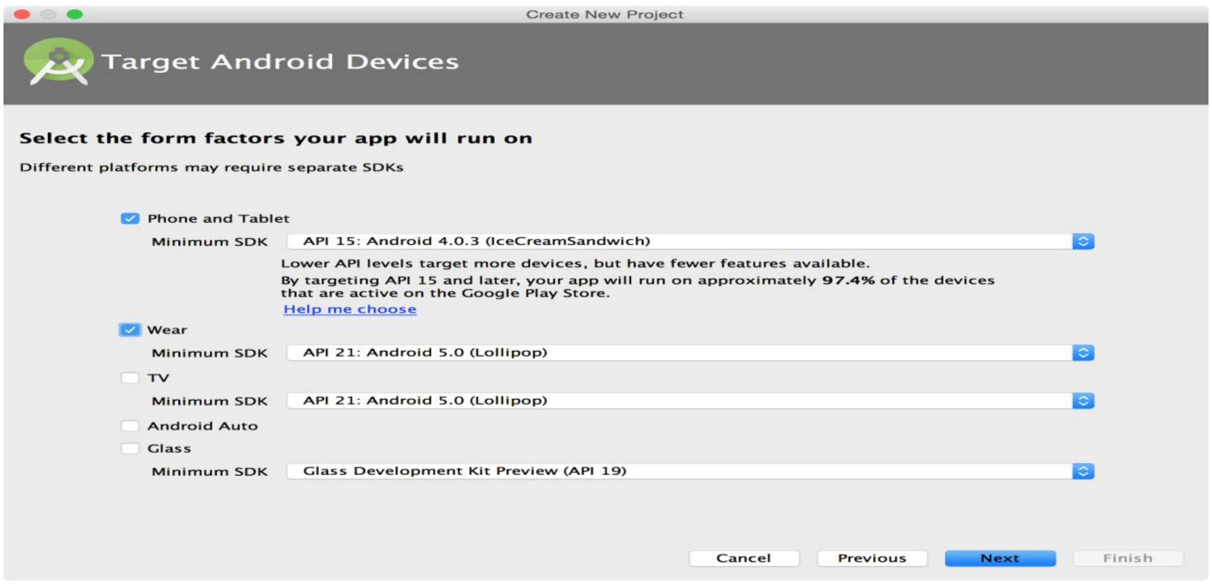
Enterprise
VPN services
Device policies
Certificate management

Device Sensors
Improved sensors
Temperature sensor
Humidity sensor

https://developer.android.com/about/versions/android-4.0.html

Figure 10: Chart of the Current Android Version Distributions

The Android Platform Distribution window shows the distribution of mobile devices running each version of Android, as shown in figure 2. Click on an API level to see a list of features introduced in the corresponding version of Android. This helps you choose the minimum API Level that has all the features that your apps needs, so you can reach as many devices as possible. Then click **OK**. [5]



Create New Project

Target Android Devices

Select the form factors your app will run on

Different platforms may require separate SDKs

☑ Phone and Tablet

Minimum SDK: API 15: Android 4.0.3 (IceCreamSandwich)

Lower API levels target more devices, but have fewer features available.
By targeting API 15 and later, your app will run on approximately **97.4%** of the devices that are active on the Google Play Store.
Help me choose

☑ Wear

Minimum SDK: API 21: Android 5.0 (Lollipop)

☐ TV

Minimum SDK: API 21: Android 5.0 (Lollipop)

☐ Android Auto

☐ Glass

Minimum SDK: Glass Development Kit Preview (API 19)

Figure 11: The Target Android Devices Screen

Then, on the Target Android Devices window, once you've selected your form factors and API versions, click **next**. [5]

2.21.3 Step 3: Add an Activity

The next screen lets you select an activity type to add to your app, as shown in figure 4. This screen displays a different set of activities for each of the form factors you selected earlier.

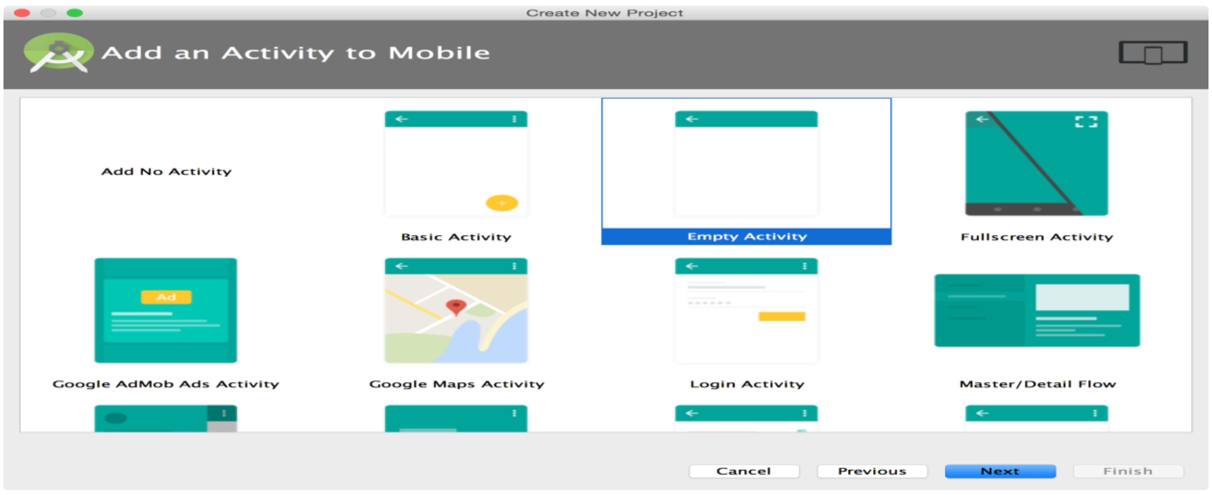
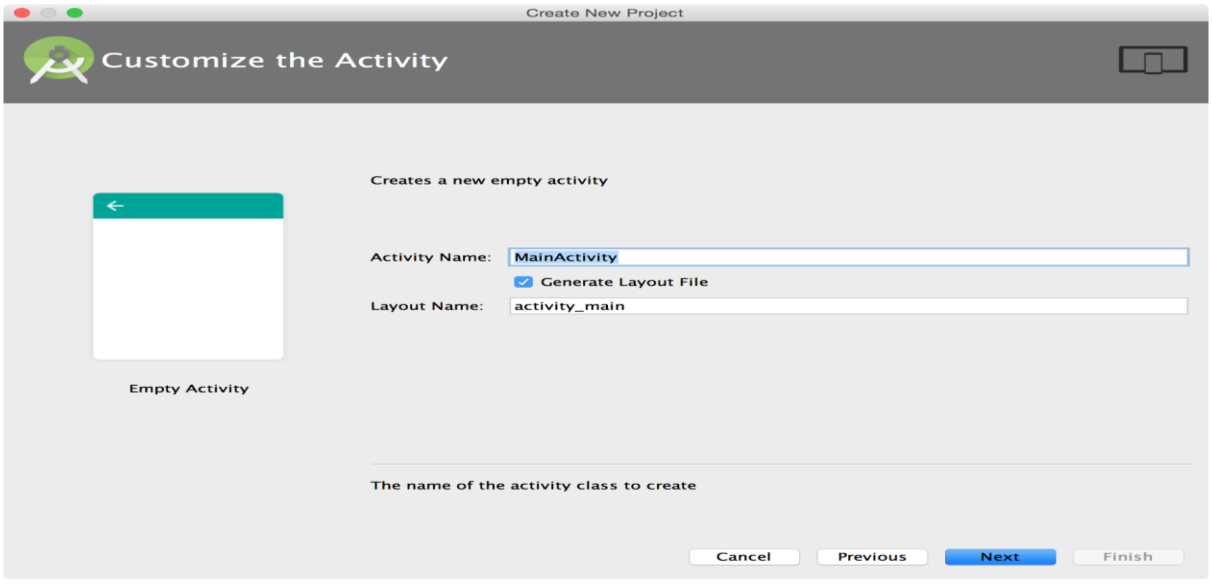


Figure 12: The Add an Activity Screen for a Mobile Form Factor

Choose an activity type then click **next**. [5]

2.21.4 Step 4: Configure Your Activity

The next screen lets you configure the activity to add to your app, as shown in below:



Create New Project

Customize the Activity

Creates a new empty activity

Activity Name: MainActivity

[✓] Generate Layout File

Layout Name: activity_main

Empty Activity

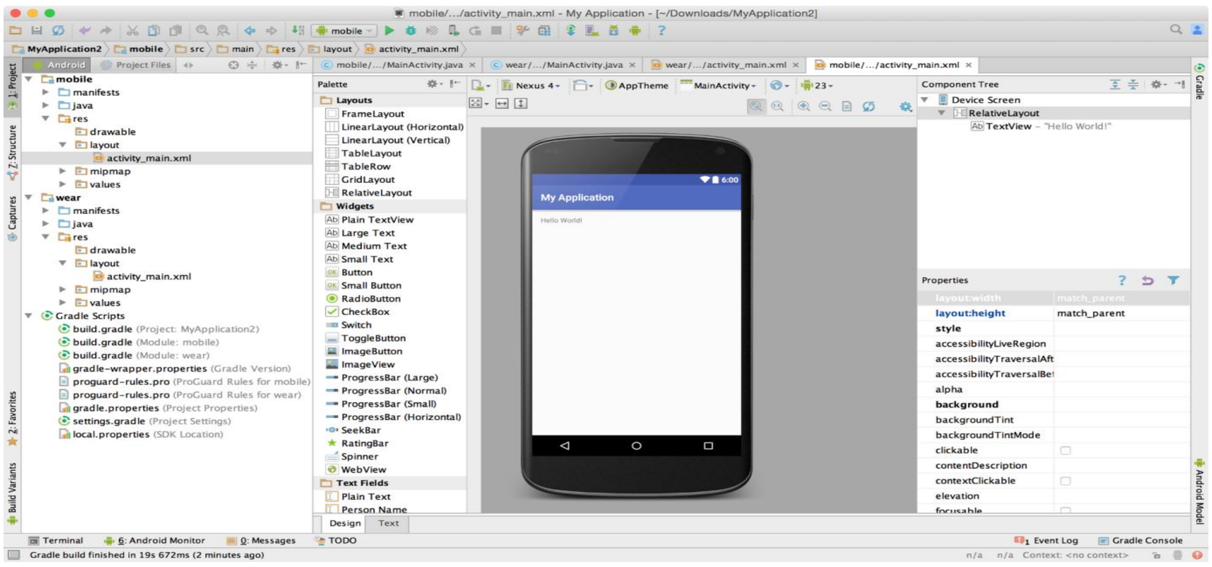
The name of the activity class to create

Figure 13: The Customize the Activity Screen

Enter the activity name, the layout name, and the activity title. Then click **Finish**. [5]

2.21.5 Step 5: Develop Your App

Android Studio creates the default structure for your project and opens the development environment. If your app supports more than one form factor, Android Studio creates a module folder with complete source files for each of them as shown in below:



mobile/.../activity_main.xml - My Application - [~/Downloads/MyApplication2]

Figure 14: Project Structure for a Newly Created App

2.22 Import an Existing Project

To import an existing project into Android Studio, proceed as follows:

1. Click **File** > **New** > **Import Project**.
2. In the **Select Eclipse or Gradle Project to Import** window that appears, navigate to the root directory of the project you want to import.
3. Click **OK**. [5]

Chapter 3: Implementation

3.1 Design

My app is a business card for those devices which are run by android operating system. The full project contains such things as listed below

- Signup & login
- User Profile
- Cards Preview
- Generate a Card
- Share the generated Card via Email, Print, Linkedin

3.2 Implementation Procedure

Creating a field for android development:

Here are some simple pre-requisites one must have to develop this application.

3.2.1 Developer Requirements

- Advanced knowledge of java
- Basic knowledge of XML
- Some Skill of Photoshop
- Database Schema
- User interface design
- Online resources

3.2.2 Hardware Requirements

The PC where I developed the app must have speedy. I have that speedy Laptop.

I am using a HP Probook 4430s.

Performance	**Design**	**Storage**	**Battery**
Core i5 2nd Gen	14.0 inches (35.56 cm)	500 GB HDD	Li-Ion
2.5 Ghz	1366 x 768 pixels	SATA5400	6 Cell
8 GB DDR3 RAM	2.04 Kg, 28.1 mm thick	RPM	3 Hrs [6]

3.3 System Requirements

Before you download and build the Android source, ensure your system meets the following requirements. Then see Establishing a Build Environment for installation instructions by operating system. [7]

3.3.1 Hardware

Your development workstation should meet or exceed these hardware requirements:

- A 64-bit environment is required for Gingerbread (2.3.x) and newer versions, including the master branch. You can compile older versions on 32-bit systems.
- At least 100GB of free disk space to check out the code and an extra 150GB to build it. If you conduct multiple builds or employ cache, you will need even more space.
- If you are running Linux in a virtual machine, you need at least 16GB of RAM/swap. [7]

3.3.2 Software

The Android Open Source Project (AOSP) master branch is traditionally developed and tested on Ubuntu Long Term Support (LTS) releases, but other distributions may be used. See the list below for recommended versions.

You workstation must have the software listed below. See Establishing a Build Environment for additional required packages and the commands to install them. [7]

3.3.3 Operating System

Android is typically built with a GNU/Linux or Mac OS operating system. It is also possible to build Android in a virtual machine on unsupported systems such as Windows.

GNU/Linux

- Android 6.0 (Marshmallow) - AOSP master: Ubuntu 14.04 (Trusty)
- Android 2.3.x (Gingerbread) - Android 5.x (Lollipop): Ubuntu 12.04 (Precise)
- Android 1.5 (Cupcake) - Android 2.2.x (Froyo): Ubuntu 10.04 (Lucid)

Mac OS (Intel/x86)

Android 6.0 (Marshmallow) - AOSP master: Mac OS v10.10 (Yosemite) or later with Xcode 4.5.2 and Command Line Tools

- Android 5.x (Lollipop): Mac OS v10.8 (Mountain Lion) with Xcode 4.5.2 and Command Line Tools
- Android 4.1.x-4.3.x (Jelly Bean) - Android 4.4.x (KitKat): Mac OS v10.6 (Snow Leopard) or Mac OS X v10.7 (Lion) and Xcode 4.2 (Apple's Developer Tools)
- Android 1.5 (Cupcake) - Android 4.0.x (Ice Cream Sandwich): Mac OS v10.5 (Leopard) or Mac OS X v10.6 (Snow Leopard) and the Mac OS X v10.5 SDK [7]

3.3.4 Java Development Kit (JDK)

Please note, since there are no available supported OpenJDK 8 packages for Ubuntu 14.04, the Ubuntu 15.04 packages must be installed manually. See JDK for Ubuntu LTS 14.04 for precise instructions.

- The master branch of Android in AOSP: Ubuntu - OpenJDK 8, Mac OS - jdk 8u45 or newer
- Android 5.x (Lollipop) - Android 6.0 (Marshmallow): Ubuntu - OpenJDK 7, Mac OS - jdk-7u71-macosx-x64.dmg
- Android 2.3.x (Gingerbread) - Android 4.4.x (KitKat): Ubuntu - Java JDK 6, Mac OS - Java JDK 6
- Android 1.5 (Cupcake) - Android 2.2.x (Froyo): Ubuntu - Java JDK 5 [7]

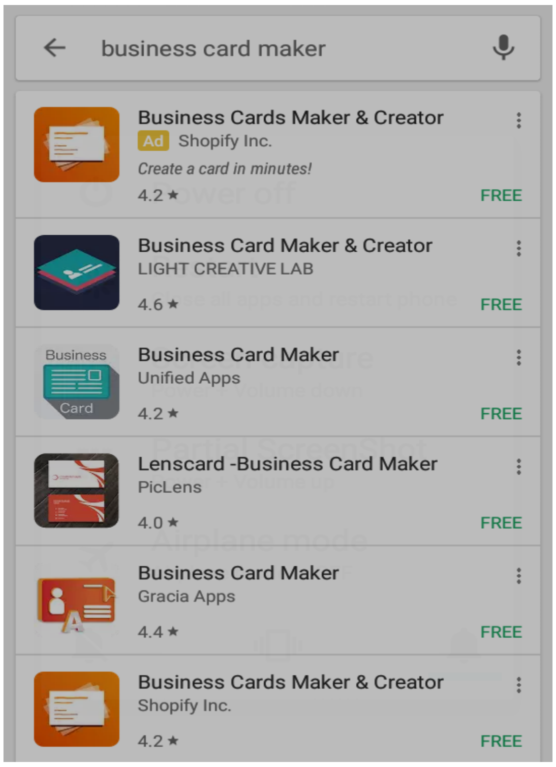
3.4 Comparison with existing cards

There are lots of business card available in the Google play store. They are divided into 2 versions as their features available for the users. They are called premium version and free version. In premium version, we have limited features or we have to upgrade those features by money to use them. In free version, we can use them easily without any cost. But in free version app, we have less features than premium version. My app is totally free but I have few extra features that make my app more unique from them. I used few apps to find out their lacking and limitation to make my app more proactive. The few key features of my app that have more than another free apps are given bellow:

- User friendly design
- Sign up and login features
- Profile management system
- More acceptable devices from SDK 16 to SDK 26

- Latest mobile version will support up to 26
- Database can modify according to the requirements of the users

Below there are screenshots of few business card maker app:



business card maker

Business Cards Maker & Creator
Ad Shopify Inc.
Create a card in minutes!
4.2 ★ FREE

Business Card Maker & Creator
LIGHT CREATIVE LAB
4.6 ★ FREE

Business Card Maker
Unified Apps
4.2 ★ FREE

Lenscard -Business Card Maker
PicLens
4.0 ★ FREE

Business Card Maker
Gracia Apps
4.4 ★ FREE

Business Cards Maker & Creator
Shopify Inc.
4.2 ★ FREE

Figure 15: Few business card screenshot from Google play store

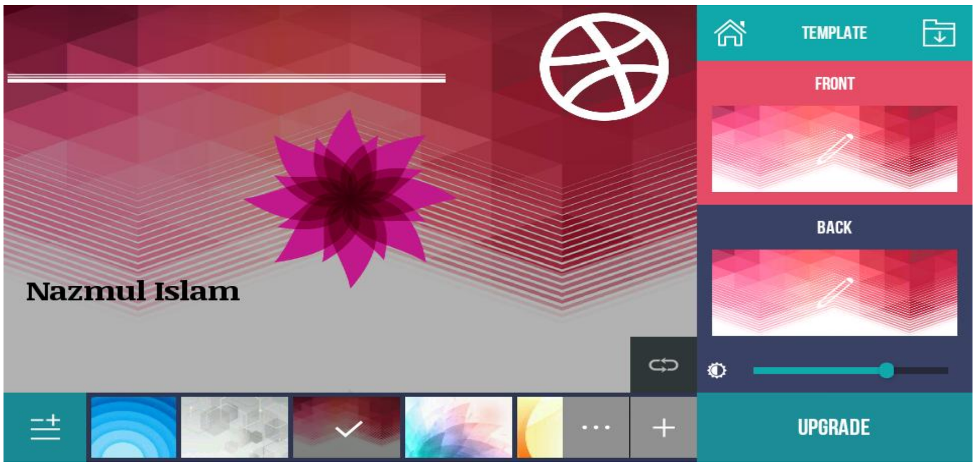
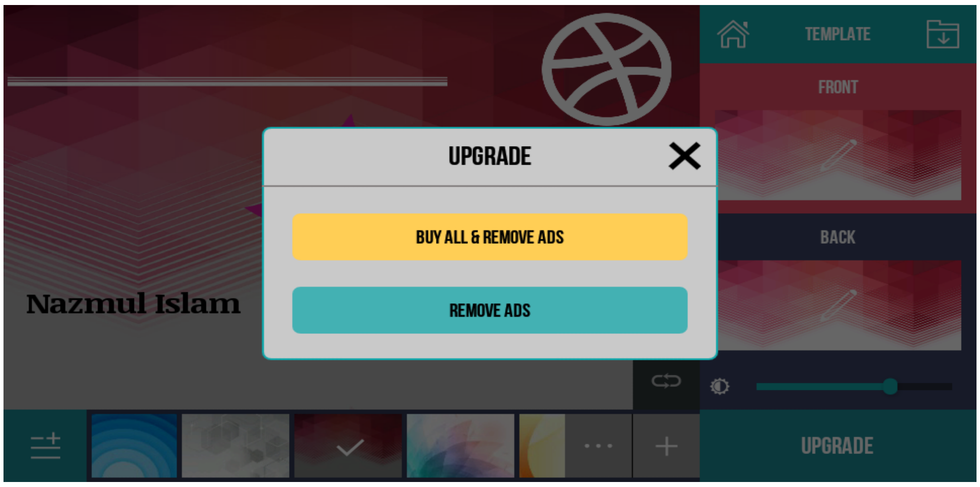
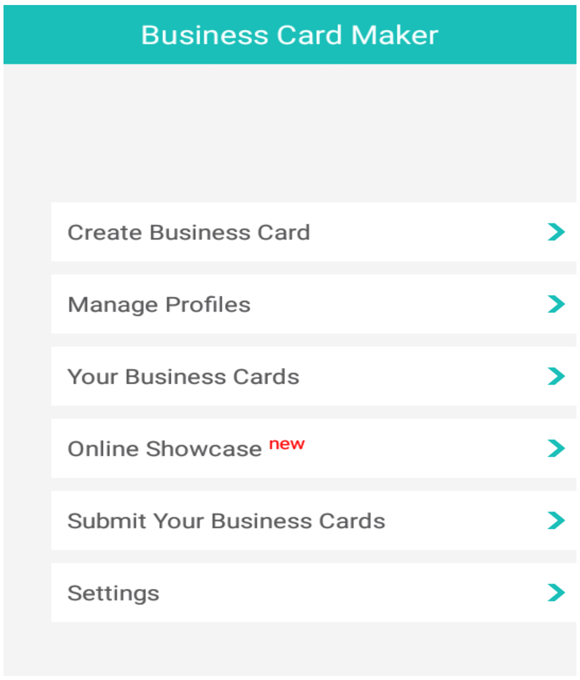


Figure 16.1: A Sample of Premium business Card



16.2: A Sample of Premium business Card



17: A Sample of Free business Card

3.5 Project Overview

- Signup & login
- User Profile
- Cards Preview
- Generate a Card
- Share the generated Card via Email, Linkedin, Facebook

- Signup & login

  A person can sign up by providing info such as user name, email and password. After sign up, user can login by using password.

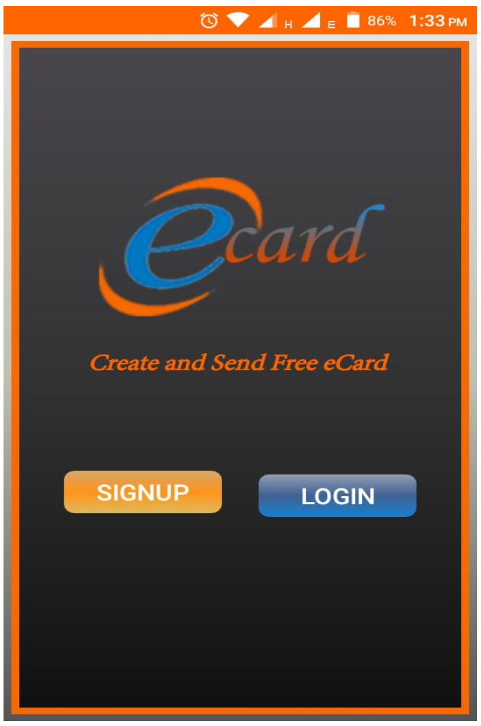


Figure 18: Launch Interface

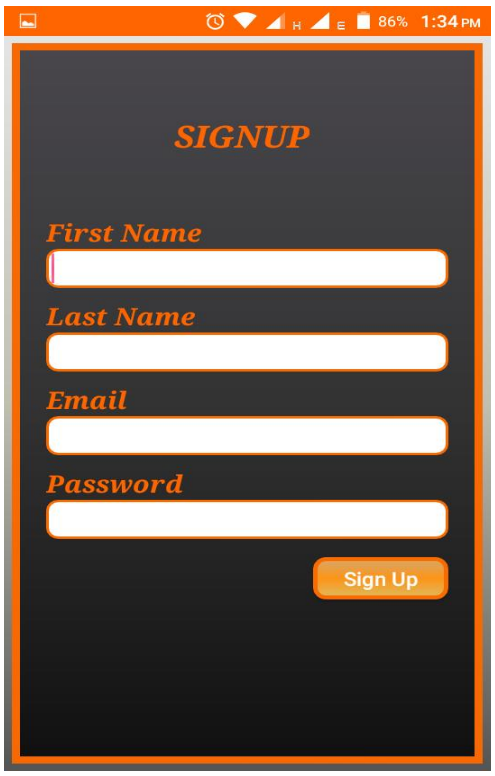


Figure 19: Sign Up Interface

Sample XML Code of Signup:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/main_bg"
    android:padding="@dimen/activity_padding"
    android:orientation="vertical"
    tools:showIn="@layout/activity_sign_up"
    tools:context="com.academic.project.ecard.SignUp">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical"
        android:background="@drawable/sign_up_custom_border">
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:gravity="center_horizontal|center_vertical"
            android:layout_marginTop="100dp"
            android:layout_marginLeft="110dp"
    android:text="SignUp"
    android:textAllCaps="true"
    android:textSize="25dp"
    android:textColor="#ff6600"
    android:textStyle="italic|bold|normal"
    android:typeface="serif" />
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="50dp"
    android:layout_marginLeft="45dp"
    android:text="Full Name"
    android:textSize="20dp"
    android:textColor="#ff6600"
    android:textStyle="italic|bold|normal"
    android:typeface="serif" />
<EditText
    android:id="@+id/et_sign_up_user_name"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:inputType="text"
    android:layout_marginLeft="45dp"
    android:layout_marginRight="45dp"
    android:background="@drawable/edit_text"
    android:padding="4dp"/>
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="10dp"
    android:layout_marginLeft="45dp"
    android:layout_marginRight="45dp"
    android:text="Email"
    android:textSize="20dp"
    android:textColor="#ff6600"
    android:textStyle="italic|bold|normal"
    android:typeface="serif" />
<EditText
    android:id="@+id/et_sign_up_email"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:inputType="text"
    android:layout_marginLeft="45dp"
    android:layout_marginRight="45dp"
    android:background="@drawable/edit_text"
    android:padding="4dp"/>
<TextView
```

```
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="10dp"
    android:layout_marginLeft="45dp"
    android:layout_marginRight="45dp"
    android:text="Password"
    android:textSize="20dp"
    android:textColor="#ff6600"
    android:textStyle="italic|bold|normal"
    android:typeface="serif" />
<EditText
    android:id="@+id/et_sign_up_password"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:inputType="textPassword"
    android:layout_marginLeft="45dp"
    android:layout_marginRight="45dp"
    android:background="@drawable/edit_text"
    android:padding="4dp" />
<Button
    android:id ="@+id/registration_button"
    android:layout_width="100dp"
    android:layout_height="35dp"
```

```
        android:layout_marginTop="15dp"

        android:layout_marginRight="45dp"

        android:layout_gravity="right"

        android:text="Sign Up"

        android:textAllCaps="false"

        android:background="@drawable/common_bordered_orrange_button"

        android:paddingLeft="5dp"

        android:paddingRight="5dp"

        android:textSize="16dp"

        android:textColor="#fff" />

    </LinearLayout>

</LinearLayout>
```

Sample Java Code of Signup**:**

```
package com.academic.project.ecard;

import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.Button;

public class SignUp extends AppCompatActivity {
    private static Button button_reg;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_sign_up);
        onClickSignUpButtonListener();
    }
    public void onClickSignUpButtonListener(){
        button_reg = (Button)findViewById(R.id.registration_button);
        button_reg.setOnClickListener(
                new View.OnClickListener() {
                    @Override
                    public void onClick(View v) {
                        Intent sign_up_intent = new Intent(SignUp.this,
CompleteProfileIndicator.class);
                        startActivity(sign_up_intent);
                    }
                }
        );
    }
}
```

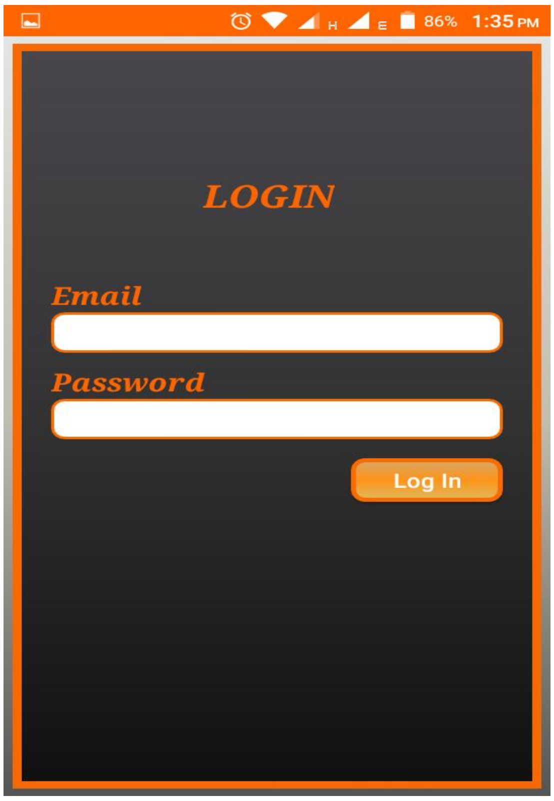
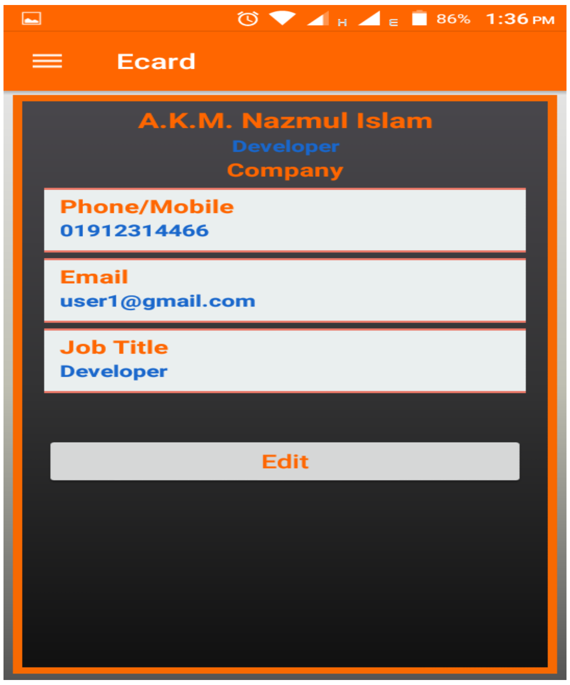


Figure 20: Login Interface

❖ User Profile
User can update own profile information that is related to generate a card.



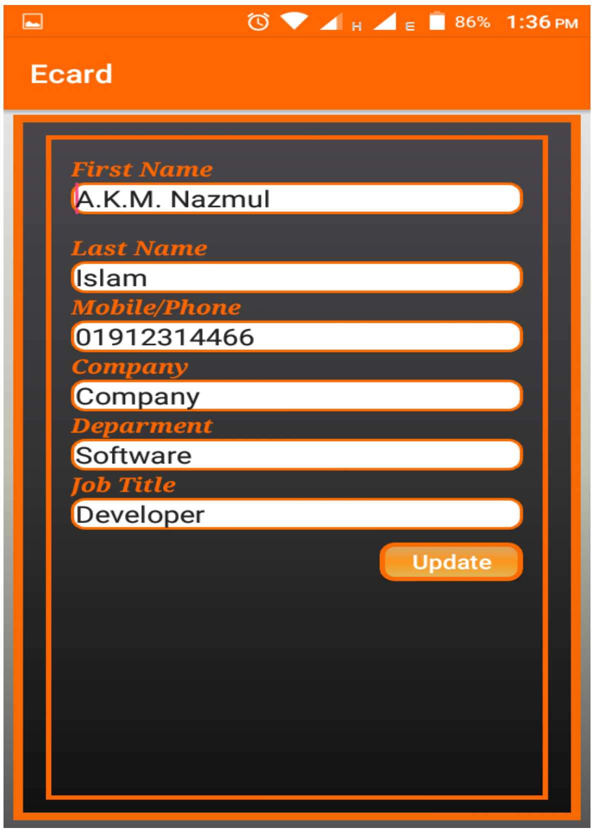
A.K.M. Nazmul Islam
Developer
Company

Phone/Mobile
01912314466

Email
user1@gmail.com

Job Title
Developer

Figure 21.1: User Profile Interface



First Name
A.K.M. Nazmul

Last Name
Islam

Mobile/Phone
01912314466

Company
Company

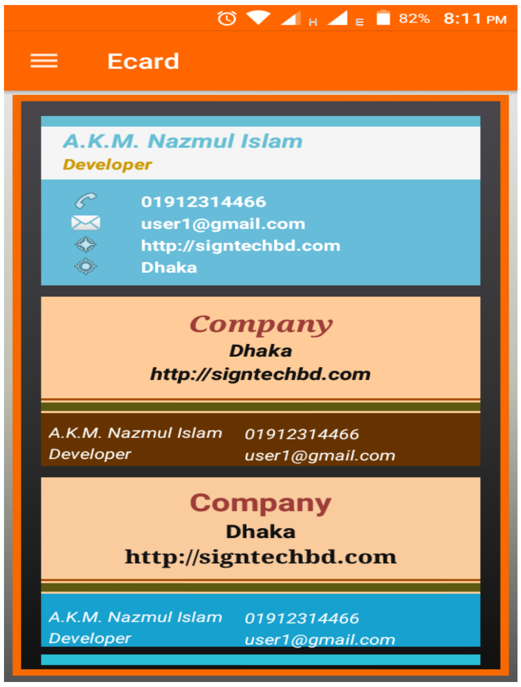
Deparment
Software

Job Title
Developer

Figure 21.2: User Edit Profile Interface

- ❖ Card Preview

  Several Cards will preview based on the information that provided by user.



A.K.M. Nazmul Islam
Developer
01912314466
user1@gmail.com
http://signtechbd.com
Dhaka

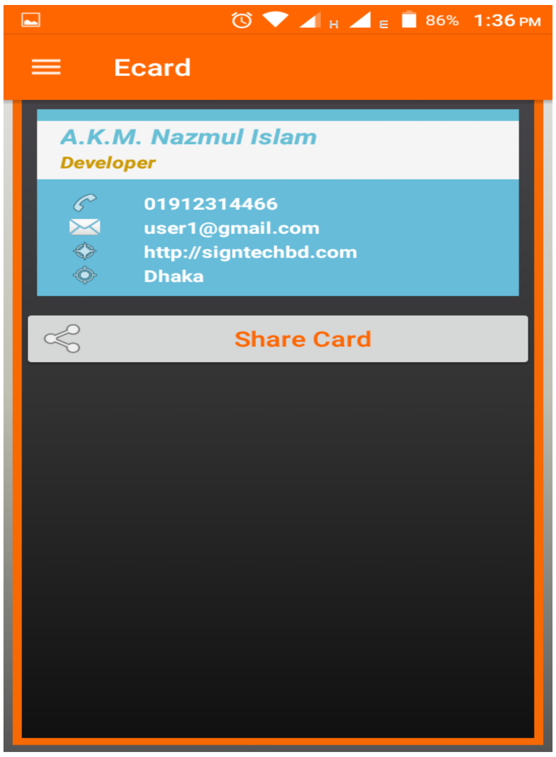
Company
Dhaka
http://signtechbd.com
A.K.M. Nazmul Islam 01912314466
Developer user1@gmail.com

Company
Dhaka
http://signtechbd.com
A.K.M. Nazmul Islam 01912314466
Developer user1@gmail.com

Figure 22: Card Preview Interface

- ❖ Generate a Card

  From several Cards, user pick up a single card to generate own card to send another.



A.K.M. Nazmul Islam
Developer
01912314466
user1@gmail.com
http://signtechbd.com
Dhaka

Figure 23: Single Card Generate Interface

- ❖ Share the generated Card via FaceBook, Linkedin , Email

  The user can able to share the generated card via FaceBook, Linkedin , Email.

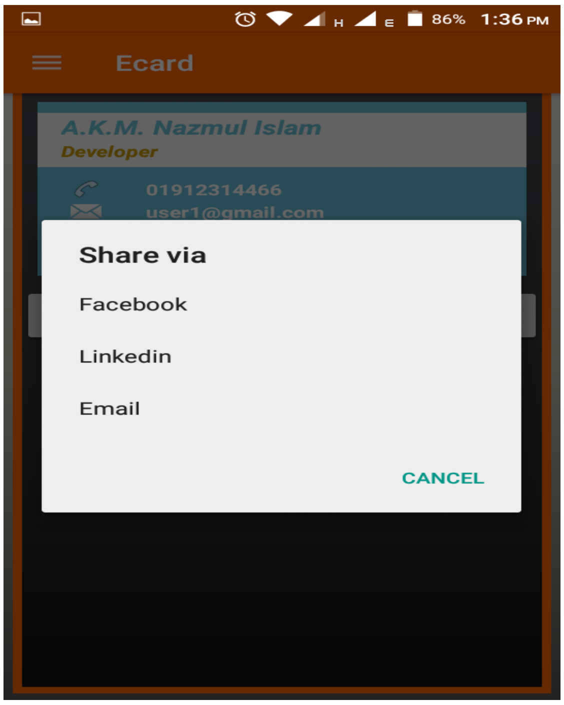


Figure 24: Card Sharing Interface

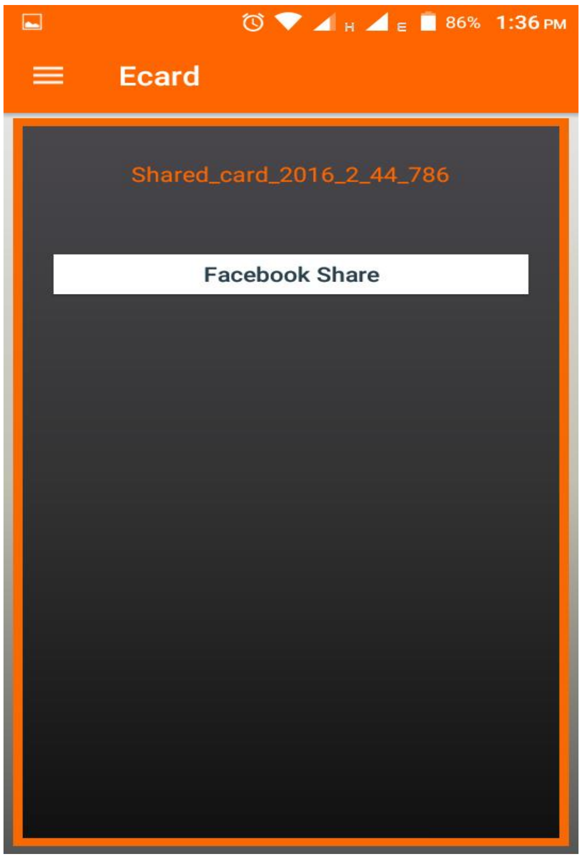


Figure 25.1: Facebook Card Sharing

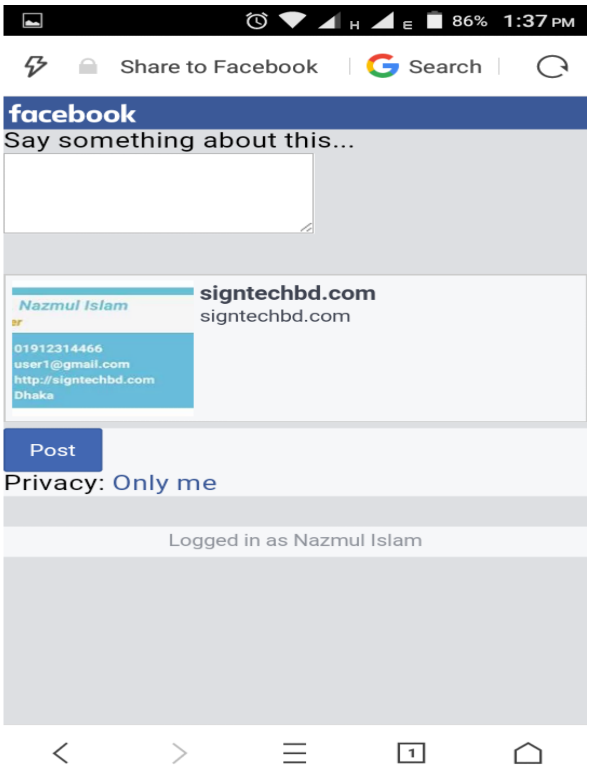


Figure 25.2: Facebook Card Sharing

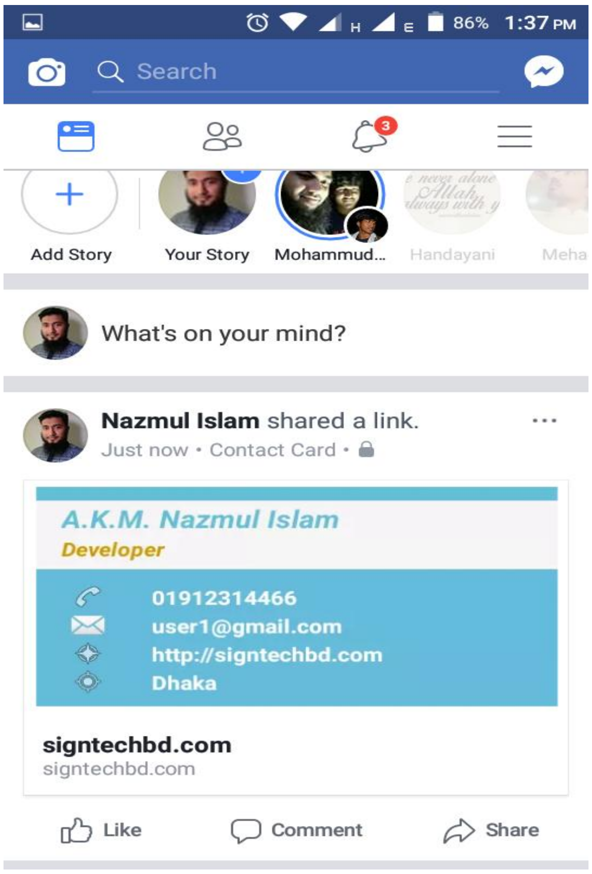


Figure 25.3: Facebook Card Sharing

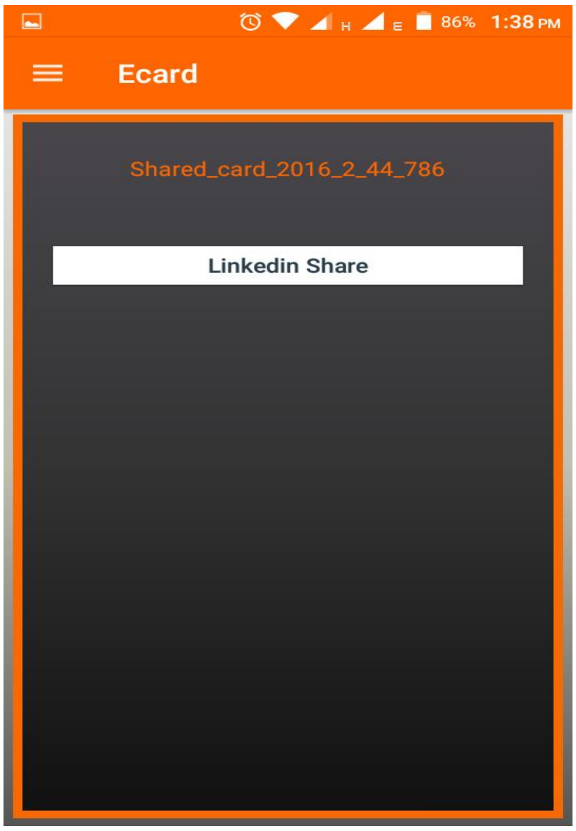
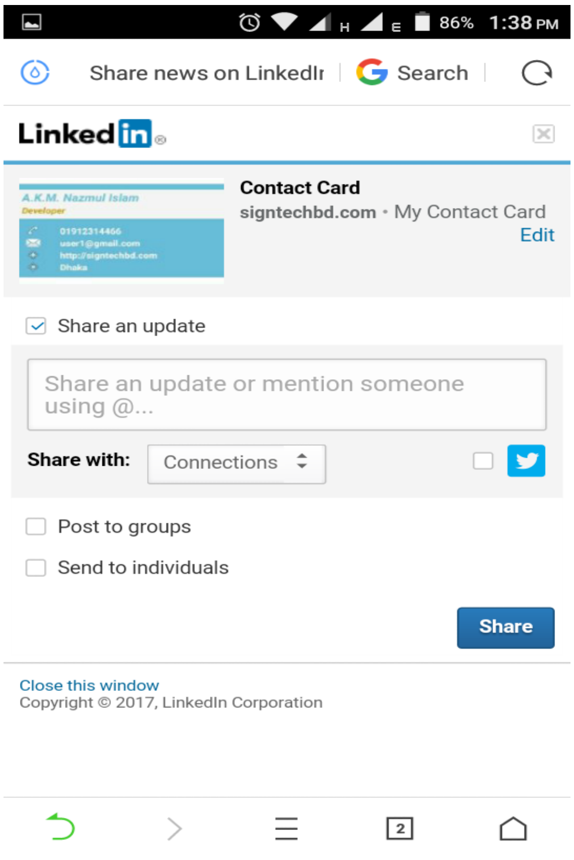


Figure 26.1: Linkedin Card Sharing



Contact Card
signtechbd.com • My Contact Card
Edit

☑ Share an update

Share an update or mention someone using @...

Share with: Connections

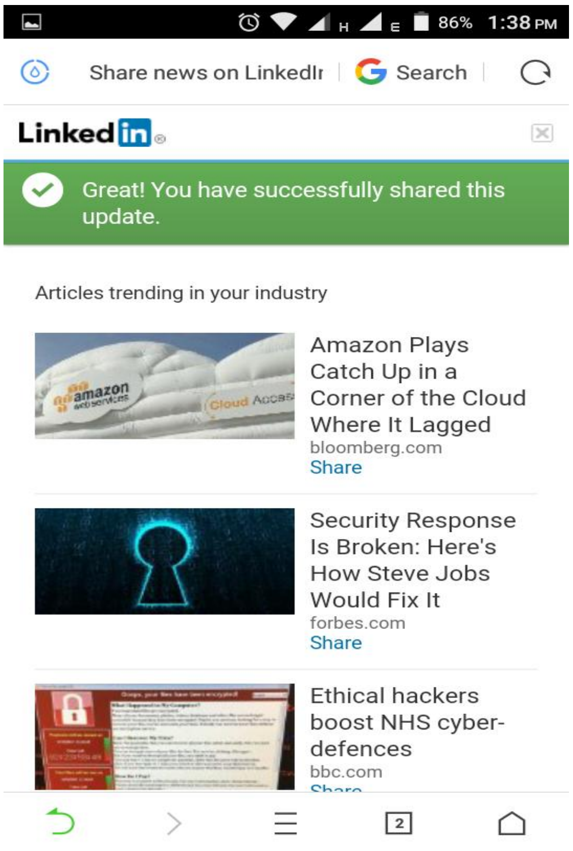
☐ Post to groups

☐ Send to individuals

Share

Close this window
Copyright © 2017, LinkedIn Corporation

Figure 26.2: Linkedin Card Sharing



86% 1:38 PM

Share news on LinkedIn | Search

LinkedIn

Great! You have successfully shared this update.

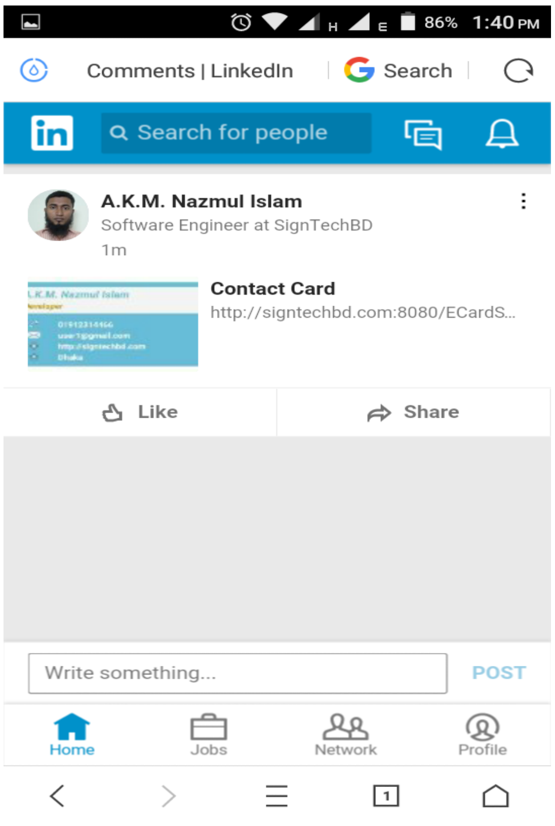
Articles trending in your industry

Amazon Plays Catch Up in a Corner of the Cloud Where It Lagged
bloomberg.com
Share

Security Response Is Broken: Here's How Steve Jobs Would Fix It
forbes.com
Share

Ethical hackers boost NHS cyber-defences
bbc.com

Figure 26.3: Linkedin Card Sharing



86% 1:40 PM

Comments | LinkedIn | Search

Search for people

A.K.M. Nazmul Islam
Software Engineer at SignTechBD
1m

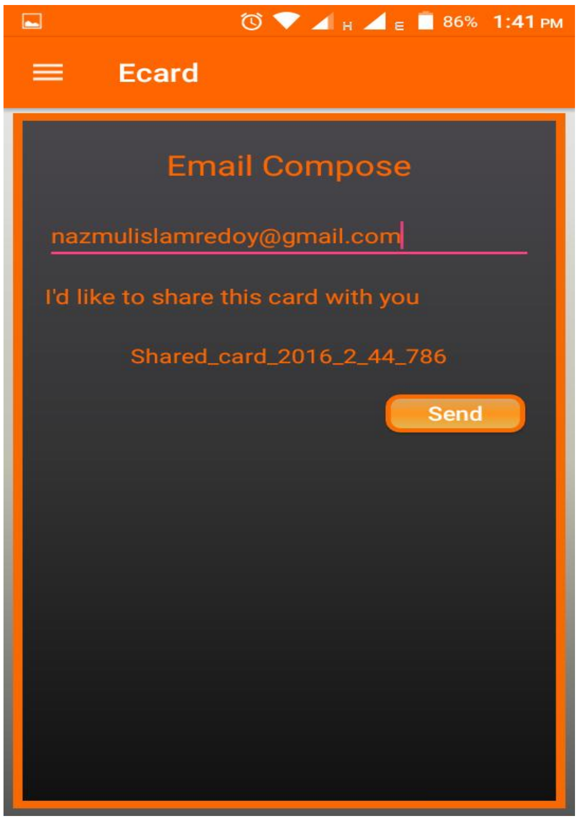
Contact Card
http://signtechbd.com:8080/ECardS...

Like | Share

Write something... POST

Home | Jobs | Network | Profile

Figure 26.4: Linkedin Card Sharing

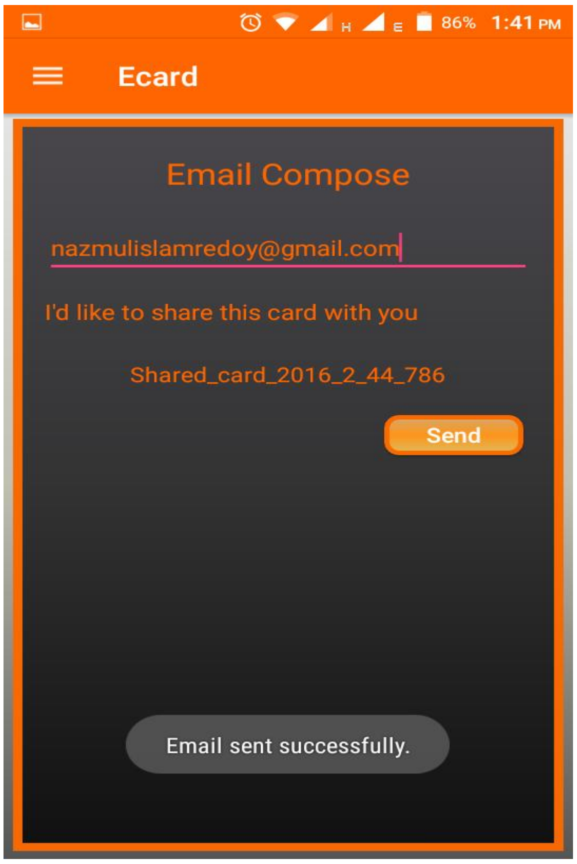


nazmulislamredoy@gmail.com

I'd like to share this card with you

Shared_card_2016_2_44_786

Figure 27.1: Email Card Sharing



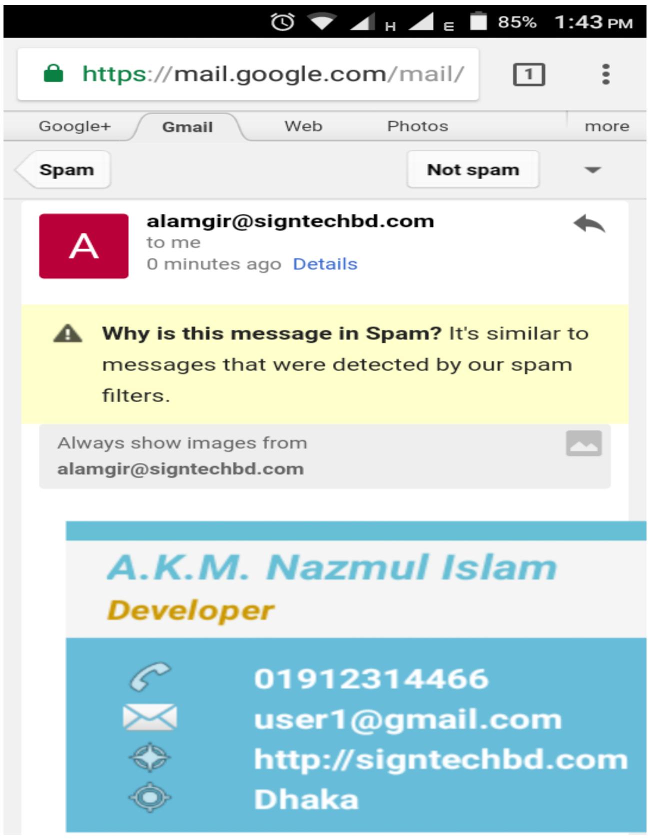
nazmulislamredoy@gmail.com

I'd like to share this card with you

Shared_card_2016_2_44_786

Email sent successfully.

Figure 27.2: Email Card Sharing



alamgir@signtechbd.com
to me
0 minutes ago

Why is this message in Spam? It's similar to messages that were detected by our spam filters.

Always show images from alamgir@signtechbd.com

A.K.M. Nazmul Islam
Developer
01912314466
user1@gmail.com
http://signtechbd.com
Dhaka

Figure 27.3: Email Card Sharing

3.6 Working Process of the Project

The interface is showing the front page of the app. In the page, there are two buttons – signup and login. At the beginning, user presses the sign up button to sign up. After signup, user will always use the login page to login the app. After filling up the signup page with first name, last name, email and password, then user will show the next page where user will get the option of all cards. This page will show the full name and email what are getting from signup page. Then user will press the "profile" button from the drawer menu and user will see a user information page where user can update his/her own information. After completing all the fields, user will come back to the all cards page and 7 business cards will display according with all the information got from user while signing up and updating profile . This page has 4 menu options. They are profile, all cards, settings and logout. In the "profile" option, user will see own profile and in this page, there is an "edit profile" button. By clicking this button, user can edit his/her own updated information. In the "all cards" option, user will see all the 7 cards. After clicking one of the particular cards, the card will show the next page with a "share card" button. The image of the single card will save automatically in the server after clicking a particular card. After clicking the share card button, user will see 3 sharing options to send a card via FaceBook, Linkedin and Email. In the “settings” option, user will see the change password option and if user clicks this option then it will go to the change password page. Thus the user can change own old password. But this change Password section or settings section is for future work. By clicking the “logout” option, user will sign out from the app.

3.7 Testing Devices

- Symphony V85
- Symphony V32
- Samsung S5
- Samsung J2
- Samsung J7
- Huawei Honor 4x
- Xiaomi Redmi 4 Prime

Chapter 4: Conclusion and Future Work

4.1 Conclusion

Thanks to Almighty Allah for the completion of the an android project within the timeline. From the starting of my university life, I always dreamed to build a career with android as an android developer and learn new technology related with it. For that reason, I am pursuing my Computer Science and Engineering degree from a renounce institution. I have learned Java Programming during my Bachelor and Master degree as academic and personal interest. I already joined few Java Programming contest as well. In 2015, I develop my 1st android platform as official purpose. Then I am planning to develop my own app and why nit this is for my project work? For this project, I choose android OS because android is the most popular mobile operating system right now.

I was trying to develop my own game which support all devices and all versions for android mobile. But it's this too much extensive work as a beginner and within a limited timeline. There is no single framework I used. All the code is written in pure Java Language. All the code is self-explanatory. Every method or function is passing value each other incorporate the flow of the app. In the process of making this app, I have learned lots of things which will make me a good android developer in my professional career. I am able to design a new app, create app logic, app flow and connecting together with the app. Some websites are really helpful to learn new things and make my basic clear. I am very glad to them to have in the web. I think that android will grow up further and never ending demands of the creating a new app will lead the developer busy and economically healthy life.

4.2 Future Work

Everything has to be uploaded regularly to keep the interest of the app user. So, my future goals are:

- First of all, in sign up page we have to add an extra field to confirm password so that user have no chance to mistake.
- After sign up page, we have to add verification by the email or mobile text code of the user so that the account of the user be safe and more secured.
- Category use card generate such as personal card, business card, company card and so on.

- More colorful cards will be added for user.
- Manage multiple type of users.
- Stored facilities of other user's cards in the user account.
- Notifications facilities should can apply to notify the end user.
- User Screen facility only used for portrait screen, landscape screen facility should be added.
- Live Card generate option should be added.
- Profile information should be updated into different categories.

4.3 References

1. https://en.wikipedia.org/wiki/Android_(operating_system)
2. https://en.wikipedia.org/wiki/Android_software_development
3. https://developer.android.com/
4. https://developer.android.com/studio/intro/index.html
5. https://developer.android.com/studio/projects/create-project.html
6. https://www.91mobiles.com/hp-4430s-core-i5-2nd-gen-4-gb-500-gb-dos-laptop-price-in-india-6442
7. https://source.android.com/source/requirements
8. https://source.android.com/