AGRICULTURE YIELD INTERACTION BETWEEN FARMERS AND RETAILERS (FarmEra)

A PROJECT REPORT

Submitted in partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING

Submitted by

M.V.SURYA TEJA
Roll No: 16731A0573

P. SUDHEER
Roll No: 16731A0574

M.CHAITANYA
Roll No: 1631A0572

D.V.SAI NAVEEN
Roll No: 16731A0570

Under the Guidance of
Dr.D.SRUJAN CHANDRA REDDY
Professor

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

PBR VISVODAYA INSTITUTE OF TECHNOLOGY & SCIENCE

KAVALI-524 201, NELLORE DISTRICT, A.P.

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTHAPUR
ANANTHAPURAM, A.P., INDIA

MARCH 2020

PBR VISVODAYA INSTITUTE OF TECHNOLOGY & SCIENCE

KAVALI

BONAFIDE CERTIFICATE

Certified that this project report **"AGRICULTURE YIELD INTERACTION BETWEEN FARMERS AND RETAILERS (FarmEra)"** is the Bonafide work done by **"M.V.SURYA TEJA (1631A0573), M.CHAITANYA (1671A0572) , P.SUDHEER (16731A0574) , D.V.SAI NAVEEN (16731A0570)",**who carried out the project under my guidance during the year 2019-2020, towards partial fulfillment of the requirements of the Degree of Bachelor of Technology in Computer Science & Engineering from Jawaharlal Nehru Technological University Anantapur , Anantapuram. The results embodied in this report have not been submitted to any other University for the award of any degree.

Dr. D. SRUJAN CHANDRA REDDY	**Dr. D. SRUJAN CHANDRA REDDY**
Professor	Professor
PROJECT GUIDE	**HEAD OF THE DEPARTMENT**
DEPARTMENT OF C.S.E.	DEPARTMENT OF C.S.E.

External Viva Voce conducted on______________________

INTERNAL EXAMINER **EXTERNAL EXAMINER**

PBR VISVODAYA INSTITUTE OF TECHNOLOGY & SCIENCE
KAVALI

CERTIFICATE OF AUTHENTICATION

We solemnly declare that this project report "**AGRICULTURE YIELD INTERACTION BETWEEN FARMERS AND RETAILERS (FarmEra)**"is the bonafide work done purely by us, carried out under the supervision of **Dr. D. SRUJAN CHANDRA REDDY, Professor** towards partial fulfillment of the requirements of the Degree of **BACHELOR OF TECHNOLOGY** in **Computer Science & Engineering** from Jawaharlal Nehru Technological University Anantapur, Ananthapuram during the year 2019 - 2020.

It is further certified that this work has not been submitted, either in part of in full, to any other department of the Jawaharlal Nehru Technological University, or any other University, institution or elsewhere, or for publication in any form.

DATE: **SIGNATURE OF THE STUDENTS**

M.V.SURYA TEJA (16731A0573)

M.CHAITANYA (16731A0572)

P.SUDHEER (16731A0574)

D.V.SAI NAVEEN (16731A0570)

ACKNOWLEDGEMENT

We express heartfelt gratitude towards our institution, **PBR VISVODAYA INSTITUTE OF TECHNOLOGY & SCIENCE, KAVALI** for giving us an opportunity for the successful completion of our degree.

We express great sense of gratitude and indebtedness to our beloved Chairman **Mr.D.Vidyadharakumar Reddy** and Academic Director **Dr.D.Prathyusha Reddi** for promoting excellent academic environment in the course.

It gives us immense pleasure to express a gratitude to our Director **Wg. Cdr. I.P.C. Reddy, (Retd.)** and to our Principal **Dr. B.Vamsee Mohan ,** who gave us an opportunity in completing this course.

We deeply express profound gratitude and wholehearted thanks to our beloved Head of the Department **Dr. D.Srujan Chandra Reddy** , Professor, Department of Computer Science and Engineering, who provided us with necessary facilities, guidance and endless encouragement which helped us a lot in completing the project with in time.

We express great sense of gratitude and indebtedness to our beloved guide **Dr. D.Srujan Chandra Reddy**, Professor for his inspiring guidance and his encouragement throughout the course and project.

We also express gratitude to our lecturers and lab coordinators, non-teaching staff and friends who directly or indirectly helped us in completing the project successfully and providing us with suitable suggestions and guidance throughout.

ABSTRACT

In modern farming the farmers try to sell the yield or crop to the mediators.But the mediators purchase the crop irrespective to the market price so that he can sell to the retailers for high price so that mediitor will be profitable and farmers will be at loss to avoid these and to make farmers profitable farmera is the best medium.Farm Era is an mobile application which enables the direct communication between farmers and retailers without involving the mediators.

Farmers and retailers need to register with valid credentials.farmers should give all their valid information like type of the crop no of acres yield time etc.so that the retailer can view their profiles and purchase the crop of their choice with current market prices so that farmer can easily sell to that retailer after the yield so there is no waiting for the seller and farmer will be profitable

Farmer after getting the crop or yield he needs to wait for the seller so that he need to preserve the crop as farmer can’t afford the warehouse or the cold storage to store that amount of crop the crop gets damaged. FarmEra prevents this to happen as the agreement is done between farmer and retailer before the farming started so retailer can buy the crop as soon as crop is produced in case of dealy as retailer payed some advance amount the farmer can use that amount to store the crop.it is also benifitable to the retailer he can continously monitor the crop status it also prevents the farmer to sell the crop to other retailer for more price.

.

LIST OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

GDP	Gross Domestic Products
API	Application Program Interface
GPS	Global Positioning System
SDK	Software Development Kit
AVD	Android Virtual Device
IDE	Integrated Development Environment
ADT	Android Development Tools
UI	User Interface
SQL	Structured Query Language
UML	Unified Modeling Languages

INTRODUCTION

1.1 ABOUT PROJECT

The history of **Agriculture in India** dates back to Indus Valley Civilization and even before that in some places of Southern India. India ranks second worldwide in farm outputs. As per 2016, agriculture employed 49.9% of the Indian work force and contributed 16.9–17.9% to country's GDP.

In 2016, agriculture and allied sectors like animal husbandry forestry and fisheries accounted for 15.4% of the GDP (gross domestic product) with about 31% of the workforce in 2014. India ranks first in the world with highest net cropped area followed by US and China. The economic contribution of agriculture to India's GDP is steadily declining with the country's broad-based economic growth. Still, agriculture is demographically the broadest economic sector and plays a significant role in the overall socio-economic fabric of India.

India exported $38 billion worth of agricultural products in 2013, making it the seventh largest agricultural exporter worldwide and the sixth largest net exporter. Most of its agriculture exports serve developing and least developed nations. Indian agricultural/horticultural and processed foods are exported to more than 120 countries, primarily to the Japan, Southeast Asia, SAARC countries, the European Union and the United States.

As per the 2014 FAO world agriculture statistics India is the world's largest producer of many fresh fruits like banana, mango, guava, papaya, lemon and vegetables like chickpea, okra and milk, major spices like chili pepper, ginger, fibrous crops such as jute, staples such as millets and castor oil seed. India is the second largest producer of wheat and rice, the world's major food staples.

India is currently the world's second largest producer of several dry fruits, agriculture-based textile raw materials roots and tuber crops pulses, farmed fish, eggs, coconut, sugarcane and numerous vegetables. India is ranked under the world's five largest producers of over 80% of agricultural produce items, including many cash crops such as coffee and cotton, in 2010.[10] India is one of the world's five largest producers of livestock and poultry meat, with one of the fastest growth rates, as of 2011.

One report from 2008 claimed that India's population is growing faster than its ability to produce rice and wheat. While other recent studies claim that India can easily feed its growing population, plus produce wheat and rice for global exports, if it can reduce food staple spoilage/wastage, improve its infrastructure and raise its farm productivity like those achieved by other developing countries such as Brazil and China.

In fiscal year ending June 2011, with a normal monsoon season, Indian agriculture accomplished an all-time record production of 85.9 million tonnes of wheat, a 6.4% increase from a year earlier. Rice output in India hit a new record at 95.3 million tonnes, a 7% increase from the year earlier. Lentils and many other food staples production also increased year over year. Indian farmers, thus produced about 71 kilograms of wheat and 80 kilograms of rice for every member of Indian population in 2011. The per capita supply of rice every year in India is now higher than the per capita consumption of rice every year in Japan.

India exported $39 billion worth of agricultural products in 2013, making it the seventh largest agricultural exporter worldwide, and the sixth largest net exporter. This represents explosive growth, as in 2004 net exports were about $5 billion. India is the fastest growing exporter of agricultural products over a 10-year period, its $39 billion of net export is more than double the combined exports of the European Union (EU-28). It has become one of the world's largest supplier of rice, cotton, sugar and wheat. India exported around 2 million metric tonnes of wheat and 2.1 million metric tonnes of rice in 2011 to Africa, Nepal, Bangladesh and other regions around the world.

Aquaculture and catch fishery is amongst the fastest growing industries in India. Between 1990 and 2010, the Indian fish capture harvest doubled, while aquaculture harvest tripled. In 2008, India was the world's sixth largest producer of marine and freshwater capture fisheries and the second largest aquaculture farmed fish producer. India exported 600,000 metric tonnes of fish products to nearly half of the world's countries. Though the available nutritional standard is 100% of the requirement, India lags far behind in terms of quality protein intake at 20% which is to be tackled by making available protein rich food products such as eggs, meat, fish, chicken etc. at affordable prices

India has shown a steady average nationwide annual increase in the kilograms produced per hectare for some agricultural items, over the last 60 years. These gains have come mainly from India's green revolution, improving road and power generation infrastructure, knowledge of gains and reforms. Despite these recent accomplishments, agriculture has the potential for major productivity and total output gains, because crop yields in India are still just 30% to 60% of the best sustainable crop yields achievable in the farms of developed and other developing countries. Additionally, post harvest losses due to poor infrastructure and unorganised retail, caused India to experience some of the highest food losses in the world

Safeguard yourself financially against natural risks like natural disasters/ calamities, insect, pests & diseases and adverse weather conditions. Š Take benefit of appropriate crop insurance scheme applicable in your area. Š Four insurance schemes are being implemented namely, Pradhan Mantri Fasal Bima Yojana (PMFBY), Weather Based Crop Insurance Scheme (WBCIS), Coconut Palm Insurance Scheme (CPIS) and Pilot Unified Package Insurance Scheme (UPIS) (45 districts). Š Coverage under PMFBY/WBCIS/CPIS/UPIS is compulsory, if you avail crop loan for notified crops. Š Coverage is voluntary for non-loanee farmers. Š Contact District Agriculture officers of State Govt./nearest branch of bank/PACS Common Service Centre (CSC) or crop insurance company operating in your area for availing the benefits under the Crop Insurance Scheme.

1.2 EXISTING SYSTEM

- In existing system there is no specific system to sale crop to retainer
- The sale of crops is through middle persons (Mediators)

1.3 DISADVANTAGES

1 Expensive to buy.

2 Less security.

3 Backup data cannot be easily generated.

4 Record keeping is complex.

5. Not Easy To sale

1.4 PROPOSED SYSTEM

- In proposed system the former can sale crop very easily just uploading crop details
- Retailer also can easily check crop information
- This app is very much suitable for formers and retailers and here we are maintaining all security precautions

1.5 ADVANTAGES

- The proposed system is user friendly and storing of data is fast and data is maintained efficiently.
- We can improve the efficiency of the system, thus overcome the drawbacks of the existing system.
- Less human error
- Strength and strain of registers and papers can be reduced
- High security
- Data consistency
- Easy to handle
- Easy data updating
- Easy record keeping
- Backup data can be easily generated

LITERATURE SURVEY

2.1 Introduction

Literature survey is the most important step in software development process. Before developing the tool it is necessary to determine the time factor, economy n company strength. Once these things are satisfied, ten next steps are to determine which operating system and language can be used for developing the tool. Once the programmers start building the tool the programmers need lot of external support. This support can be obtained from senior programmers, from book or from websites. Before building the system the above consideration are taken into account for developing the proposed system.

2.2 History of Android

Android, Inc., was founded in Palo Alto, California in October 2003 by Andy Rubin (co-founder of Danger), Rich Miner (co-founder of Wildfire Communications, Inc.),Nick Sears (once VP at T-Mobile) and Chris White (headed design and interface development at WebTV) to develop "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. Though, when it was realized that the market for the devices was not large enough, the company diverted its efforts toward producing a smartphone operating system that would rival Symbian and Microsoft Windows Mobile. Despite the past accomplishments of the founders and early employees, Android Inc. operated secretly, revealing only that it was working on software for mobile phones. That same year, Rubin ran out of money. Steve Perlman, a close friend of Rubin, brought him $10,000 in cash in an envelope and refused a stake in the company.

In July 2005, Google acquired Android Inc. for at least $50 million. Its key employees, including Rubin, Miner and White, stayed at the company after the acquisition. Not much was known about Android Inc. at the time, but many assumed that Google was planning to enter the mobile phone market with this move.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 component and software partners and signaled to carriers that it was open to various degrees of cooperation on their part.

Speculation about Google's intention to enter the mobile communications market continued to build through December 2006.An earlier prototype codenamed "Sooner" had a closer resemblance to a BlackBerry phone, with no touchscreen, and a physical, QWERTY keyboard, but was later re-engineered to support a touchscreen, to compete with other announced devices such as the 2006 LG Prada and 2007 Apple iPhone.In September 2007, Information Week covered an survey 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, Sony and Samsung, wireless carriers such as Sprint Nextel and T-Mobile, and chipset makers such as Qualcomm and Texas Instruments, unveiled itself, with a goal to develop open standards for mobile devices. That day, Android was unveiled as its first product, a mobile device platform built on the Linux kernel. The first commercially available smartphone running Android was the HTC Dream, released on October 22, 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; for example, version 1.5 "Cupcake" was followed by 1.6 "Donut". In 2010, Google launched its Nexus series of devices – a line of smartphones and tablets running the Android operating system, and built by manufacturing partners. HTC collaborated with Google to release the first Nexus smartphone,the Nexus One. Google has since updated the series with newer devices, such as the Nexus 5 phone (made by LG) and the Nexus 7 tablet (made by Asus). Google releases the Nexus phones and tablets to act as their flagship Android devices, demonstrating Android's latest software and hardware features. From 2013 until 2015, Google offered several Google Play Edition devices over Google Play.

While not carrying the Google Nexus branding, these were Google-customized Android phones and tablets that also ran the latest version of Android, free from manufacturer or carrier modifications.

From 2010 to 2013, Hugo Barra served as product spokesperson, representing Android at press conferences and Google I/O, Google's annual developer-focused conference. Barra's product involvement included the entire Android ecosystem of software and hardware, including Honeycomb, Ice Cream Sandwich, Jelly Bean and KitKat operating system launches, the Nexus 4 and Nexus 5 smartphones, the Nexus 7 and Nexus 10 tablets and other related products such as Google Now and Google Voice Search, Google's speech recognition product comparable to Apple's Siri. In 2013, Barra left the Android team for Chinese smartphone maker Xiaomi. The same year, Larry Page announced in a blog post that Andy Rubin had moved from the Android division to take on new projects at Google. He was replaced by Sundar Pichai who became the new head of Android and Chrome OS and later, by Hiroshi Lockheimer when Pichai became CEO of Google.

In 2014, Google launched Android One, a line of smartphones mainly targeting customers in the developing world. In May 2015, Google announced Project Brillo as a cut-down version of Android that uses its lower levels (excluding the user interface), intended for the "Internet of Things" (IoT) embedded systems.

In October 2015, researchers at the University of Cambridge concluded that almost 90% of Android phones in use had known but unpatched security vulnerabilities due to lack of updates and support.Ron Amadeo of Ars Technica wrote in August that "Android was originally designed, above all else, to be widely adopted. Google was starting from scratch with zero percent market share, so it was happy to give up control and give everyone a seat at the table in exchange for adoption.Now, though, Android has around 75-80 percent of the worldwide smartphone market—making it not just the world's most popular mobile operating system but arguably the most popular operating system, period. As such, security has become a big issue. Android still uses a software update chain-of-command designed back when the Android ecosystem had zero devices to update, and it just doesn't work".

2.3 History of Navigation

GPS is primarily a navigational system, so a background on navigation will give insight as to how extraordinary the Global Positioning System is.

People first navigated only by means of landmarks - mountains, trees, or leaving trails of stones. This would only work within a local area and the environment was subject to change due to environmental factors such as natural disasters.

For traveling across the ocean a process called dead reckoning, which used a magnetic compass and required the calculation of how fast the ship was going, was applied. The measurement tools were crude and inaccurate. It was also a very complicated process.

When traveling over the ocean, people began using the stars as guidelines. The stars appear different from different locations on Earth so analysing the stars gave sailors the basic direction to follow. Celestial navigation was our primary means of navigation for hundreds of years. It was a time-consuming and complicated task of measuring the angles between stars - a process of triangulation. The degree of precision was limited. The sextant was developed during this time but since it only measured latitude, a timepiece was also invented so that the longitude could also be calculated. This type of navigation only worked at night and in clear weather which was a great disadvantage.

It was not until the 20th century that ground-based radio navigation systems were introduced. Some are still in use today. GPS is a satellite radio navigation system, but the first systems were ground-based. They work in the same way as does GPS: users (receivers) calculate how far away they are from a transmitting tower whose location is known. When several towers are used, the location can be pinpointed. This method of navigation was a great improvement, yet it had its own difficulties. An example of such a system is LORAN. Each tower had a range of about 500 miles and had accuracy good to about 250 meters. LORAN was not a global system and could not be used over the ocean. Because ground based systems send signals over the surface of the earth, only two- dimensional location can be determined. The altitude cannot be calculated so this system could not be applied to aviation. The accuracy of such systems could be affected by geography as well. The frequency of the signal affected accuracy; a higher frequency would allow for greater accuracy, but the user would need to remain within the line of sight. The first global navigation system was called OMEGA. It was a ground-based system but has been terminated as of 1997.

Satellite navigation systems can provide high frequency signals allowing for high accuracy, as well as global access because the satellites are so high up that remaining within the line of sight of the satellites is easy.

2.4 History of GPS

Prior to the development of the GPS system, the first satellite system was called Transit and was operational beginning in 1964. Transit had no timing devices aboard the satellites and the time it took a receiver to calculate its position was about 15 minutes. Yet, much was learned from this system. GPS is a great improvement over the Transit system. The original use of GPS was as a military positioning, navigation, and weapons aiming system to replace not only Transit, but other navigation systems as well. It has higher accuracy and stable atomic clocks on board to achieve precise time transfer. The first GPS satellite was launched in 1978 and the first products for civilian consumers appeared in the mid 1980's. It was in 1984 that President Reagan announced that a portion of the capabilities of GPS would be made available to the civil community. The system is still being improved and new, better satellites are still being launched to replace older ones.

How does GPS work?

Each of the GPS satellites transmits radio signals. GPS receivers pick up these signals and measure the distance to a satellite by multiplying the speed of the signal by the time it takes the signal to get there. The speed of the signal is the speed of light and the time is encoded within the signal. The satellites also send information on their exact location.In order to find longitude, latitude, and altitude, four satellites are needed. If a measurement is taken using just one satellite, then all that is known is that the receiver is on the surface of a sphere with radius equal to the distance to the satellite. If two satellites are used, then the receiver must be on the surface of both spheres which is the intersection of the two spheres or the perimeter of a circle. If a third satellite is used, then the location of the user is narrowed down to the two points where the three spheres intersect. Three measurements are enough for land receivers since the lower of the two points would be selected. But when in the air or space, four satellites are needed: the intersection of all four spheres will be the receiver's location. When more than four satellites are used, greater accuracy can be achieved.

Services

There are two types of GPS services. Precise Positioning Service (P-code) is more accurate and reserved for the U.S. military and select government agency users. The other service is the Standard Positioning Service which is freely available to all users. The SPS code (C/A code) has errors purposefully encoded into it for U.S. national security reasons and is used for non-military applications. One source of error is Selective Availability (SA) and is implemented into the signal in order to keep non U.S. military users from attaining high accuracy. The errors in the signal are constantly changing. SA affects signals concerning the satellite's clock and thereby gives false information on how far the satellite is from the user which makes the receiver give less accurate values.

Applications

The applications of the Global Positioning System fall into five categories: location, navigation, timing, mapping, and tracking. Each category contains uses for the military, industry, transportation, recreation and science.

SYSTEM ANALYSIS

3.1 INTRODUCTION

Android provides a rich application framework that allows you to build innovative apps and games for mobile devices in a Java language environment. The documents listed in the left navigation provide details about how to build apps using Android's various APIs. If you're new to Android development, it's important that you understand the following fundamental concepts about the Android app framework:

Apps provide multiple entry points

Android apps are built as a combination of distinct components that can be invoked individually. For instance, an individual activity provides a single screen for a user interface, and a service independently performs work in the background.

From one component you can start another component using an intent. You can even start a component in a different app, such as an activity in a maps app to show an address. This model provides multiple entry points for a single app and allows any app to behave as a user's "default" for an action that other apps may invoke.

STEP BY STEP PROCEDURE

Install Android SDK, Eclipse, and Emulator (AVDs)

Android is a large and fast-growing segment of the mobile phone market. With potentially 350,000 users activating a new Android phone every day and multiple Android App Stores popping up (Google’s Android Market was recently revamped, and now other markets are coming online, like Amazon’s Android App Store), now is a great time to jump into Android development.

This walk-through will get you started installing the Android Software Development Kit (Android SDK), installing and configuring the Eclipse IDE for Android development, and choosing and installing Android Virtual Devices (AVDs) to emulate the Android environment right on your local computer. After following these steps, you will be ready to create your first Android application!

1. Download the Android Software Development Kit (SDK)

http://developer.android.com/sdk/index.html

The very first step is to download the Android Software Development Kit (SDK) that will let you emulate Android on your local computer. It is not too large (only ~30MB, compared to the monolithic XCode/iPhone SDK, which is almost 4GB!). From the Android SDK Download Page, make sure to choose the version that is correct for your operating system.

After your Android SDK download is complete, unzip and move the new folder to a permanent location (*not* your downloads directory). I use a folder in my home directory (~/Android/android-sdk-mac_x86/) but you can move it anywhere you would like. There is no wrong location. Wherever you choose will hereby known as $ANDROID for future reference.

2. Download Eclipse IDE for Java Developers

http://www.eclipse.org/downloads/

When I develop for Android, I choose to use Eclipse as my Integrated Development Environment (IDE). Eclipse can be suitably adapted for Android development since you can get plugins to help with creating your Android project, launching your Android emulator, and preparing your Android application for the Android Market. It is not an ideal IDE, but the pros outweigh the cons for Android Development.

From the Eclipse Downloads Page, choose the "Eclipse IDE for Java Developers. Make sure you are getting the correct version for your operating system. Eclipse is fairly large (~100MB) but still a lot smaller than the 4GB XCode for the iPhone!

After your Eclipse IDE download is complete, unzip and move to a permanent folder. I use the applications directory in my home folder (~/Applications/).

For OSX users: we will want to have access to Eclipse.app from within our Applications folder. To do this:

- Make an alias of the Eclipse.app file (CTRL-Click then “Make Alias”)
- Move the alias into root applications folder (/Applications/) so that it shows up next to your other applications

Now you can use Eclipse just like any other application, including adding it to your dock.

I do this with all of programs that I download that do not have installers (i.e., they are just .app files). This may not be a necessary step (I am fairly new to OSX), so please let me know if there is a better way!

3. Install the Android Development Tools (ADT) plugin for Eclipse

http://developer.android.com/sdk/eclipse-adt.html#downloading

Next, we will use Eclipse to install the Android Development Tools (ADT) using Eclipse’s built-in plug-in system. From within Eclipse:

1. Choose “Help” > “Install New Software….”
2. Click the “Add…” button and create a new entry:
    - Name: “Android ADT” (this space is for your own personal use, so name it whatever you want)
    - Location: “https://dl-ssl.google.com/android/eclipse/” (try just http:// if the https:// does not work for you)
3. Check all the boxes to install all the tools
4. Just keep clicking “I agree”, “Next”, “Yes”, etc. until it asks you to restart
5. Go ahead and restart Eclipse when prompted to

4. Connect Android SDK with Eclipse IDE

This next step connects the Android SDK from Step #1 to the Eclipse IDE from Step #2. In Step #3, you should have restarted Eclipse. If you have not done so, do that now. From within Eclipse:

1. Click on the "Eclipse" menu (next to the apple logo for OSX) and choose "Preferences"
2. Click on "Android" heading in the menu-tree to open our Android Eclipse preferences
3. Click the "Browse…" button to the right of the "SDK Location" box
4. Enter the location of your Android SDK (the $ANDROID path from Step #1)

5. (Optional) Additional Android/Eclipse Configure

While we are in the Android section of our Eclipse preferences, let's change a few more things. These steps are entirely optional. For these additional preferences, we need to expand the menu-tree under the "Android" heading:

1. Click on the triangle next to the "Android" heading in the preferences tree to expand our options
2. Click on the "DDMS" sub-heading and change the "Logging Level" settings to "Verbose" so that we see everything that goes on while developing
3. In the "Usage Stats" sub-heading, click the checkbox to allow Google to know how we are using the SDK (seems fair enough, right ?)

6. Decide Which Android Platforms You Will Support

http://developer.android.com/resources/dashboard/platform-versions.html

This graph will help you decide which ones are relevant and worth your time. I recommend checking the Android Platform Versions Graph every month or so to see how rapidly it changes! When I first found this graph less than a year ago, Android 1.5 and Android 1.6 together represented ~50% of the graph. Today, they are only ~8%. For me, this dramatic change in Android 1.5/1.6 deployment means that my efforts will be better spent focusing on Android 2.1+. For you, it may be worth it to support 1.5/1.6 for those 8% of users. Only you can make that decision.

When deciding, consider that you will need to test, debug, and provide customer support for every version of Android that you support, and for each individual device that runs those versions! For example, deciding to support 2.1 means that there are a whole host of different hardware devices that you may receive feedback regarding; whereas supporting 2.3.3 (for right now) only means the Nexus One and the Nexus S. Just keep that in mind while deciding.

From a resources standpoint, not only do you need to test and support each version that you plan to release for, but you also need to have those Android SDKs and Android Emulators on your machine (which takes up space; which, on my smaller SSD, is a finite resource). It may not be an issue for your, but it is just yet another thing to keep in mind.

Finally, no matter which versions you choose to support right now, make sure to check the Android Platform Versions Graph every month or so to see how it is changing, and to adjust accordingly. This will let you know when you can stop supporting older versions of Android and, most importantly, will let you know when you need to start supporting newer versions of Android as they grow and gain traction.

7. Install Android SDK Components

http://developer.android.com/sdk/adding-components.html

Android is packaged in such a way that the base Android SDK (downloaded in Step #1) is distinct and separate from each API version of the Android SDK. This means that for each version we want to support (from Step #6), we need to download a separate Android SDK for that version. This can be very annoying when installing (notice how many steps we have done by now), but in the long-run is a very beneficial design for us Android Developers. As new API versions are added and old API versions are phased out, we can install/uninstall the APIs as components, rather than a single huge download like XCode is for iPhone (4GB! I just can’t get over that! Who has a 4GB download for a minor version change?!)

We need to download the Software Development Kits (SDKs) for the Android versions that we want to support.

To do this, we can use the Eclipse IDE + Android ADT that we installed in Step #3. From within Eclipse:

- Click on “Window” then “Android SDK and AVD Manager”
- In “Available packages”, select the platforms you want to support. You can either choose all, or pick-and-choose what you want to develop for. For example, 2.1, 2.2, and 2.3.3 are all I care about, so I am using API 7, 8, and 10. In the “Android Repository” package, I checked the boxes next to:
    - Android SDK Platform-tools, revision 3
    - SDK Platform Android 2.3.3, API 10, revision 1
    - SDK Platform Android 2.2, API 8, revision 1
    - SDK Platform Android 2.1, API 7, revision 1
    - Samples for SDK API 10, revision 1
    - Samples for SDK API 8, revision 1
    - Samples for SDK API 7, revision 1
    - Android Compatibility package, revision 1
- In the “Third party Add-ons”, decide what you are interested in. If you are going to be using Google Maps (or anything Google beyond Android), you want to install their APIs. If you want their licensing / billing packages, get those too. I checked the boxes next to:
    - Google APIs by Google Inc., Android API 10, revision 1
    - Google APIs by Google Inc., Android API 8, revision 1
    - Google APIs by Google Inc., Android API 9, revision 1
    - Google Market Licensing package, revision 1
    - Google Market Billing package, revision 1
- Choose “Install Selected”, then the “Accept All” radio button, then “Install”. This may take a while. If it seems like your download has paused, check for any

confirmation dialogs that you need to click “Accept” to. These can sometimes be hiding in the background.

8. Create Your Android Virtual Devices (AVDs)

http://developer.android.com/guide/practices/screens_support.html#testing

Last but not least, we need to create Android Virtual Devices (AVDs) that will be our Android Emulators for running and testing our Android applications on our local computer. In the same “Android SDK and AVD Manager” from Step #7, choose “Virtual Devices” on the left and create “New…” ones. I like to create AVDs to represent different Android versions that I want to test, as well as different hardware specs and screen densities my users are likely to be using.

The main idea is to test different versions of the Android API, as well as different screen resolutions and densities. I tend to pair older versions of Android (most likely running on older hardware) with lower screen densities, and new versions of Android (most likely running on newer hardware) with better screen resolutions.

3.2 Android Libraries

This category encompasses those Java-based libraries that are specific to Android development. Examples of libraries in this category include the application framework libraries in addition to those that facilitate user interface building, graphics drawing and database access. A summary of some key core Android libraries available to the Android developer is as follows −

android.app − Provides access to the application model and is the cornerstone of all Android applications.

android.content − Facilitates content access, publishing and messaging between applications and application components.

android.database − Used to access data published by content providers and includes SQLite database management classes.

android.opengl − A Java interface to the OpenGL ES 3D graphics rendering API.

android.os − Provides applications with access to standard operating system services including messages, system services and inter-process communication.

android.text − Used to render and manipulate text on a device display.

android.view − The fundamental building blocks of application user interfaces.

android.widget − A rich collection of pre-built user interface components such as buttons, labels, list views, layout managers, radio buttons etc.

android.webkit − A set of classes intended to allow web-browsing capabilities to be built into applications.

Having covered the Java-based core libraries in the Android runtime, it is now time to turn our attention to the C/C++ based libraries contained in this layer of the Android software stack.

Android Runtime

This is the third section of the architecture and available on the second layer from the bottom. This section provides a key component called Dalvik Virtual Machine which is a kind of Java Virtual Machine specially designed and optimized for Android.

The Dalvik VM makes use of Linux core features like memory management and multi-threading, which is intrinsic in the Java language. The Dalvik VM enables every Android application to run in its own process, with its own instance of the Dalvik virtual machine.

The Android runtime also provides a set of core libraries which enable Android application developers to write Android applications using standard Java programming language.

Application Framework

The Application Framework layer provides many higher-level services to applications in the form of Java classes. Application developers are allowed to make use of these services in their applications.

The Android framework includes the following key services −

Activity Manager − Controls all aspects of the application lifecycle and activity stack.

Content Providers − Allows applications to publish and share data with other applications.

Resource Manager − Provides access to non-code embedded resources such as strings, color settings and user interface layouts.

Notifications Manager − Allows applications to display alerts and notifications to the user.

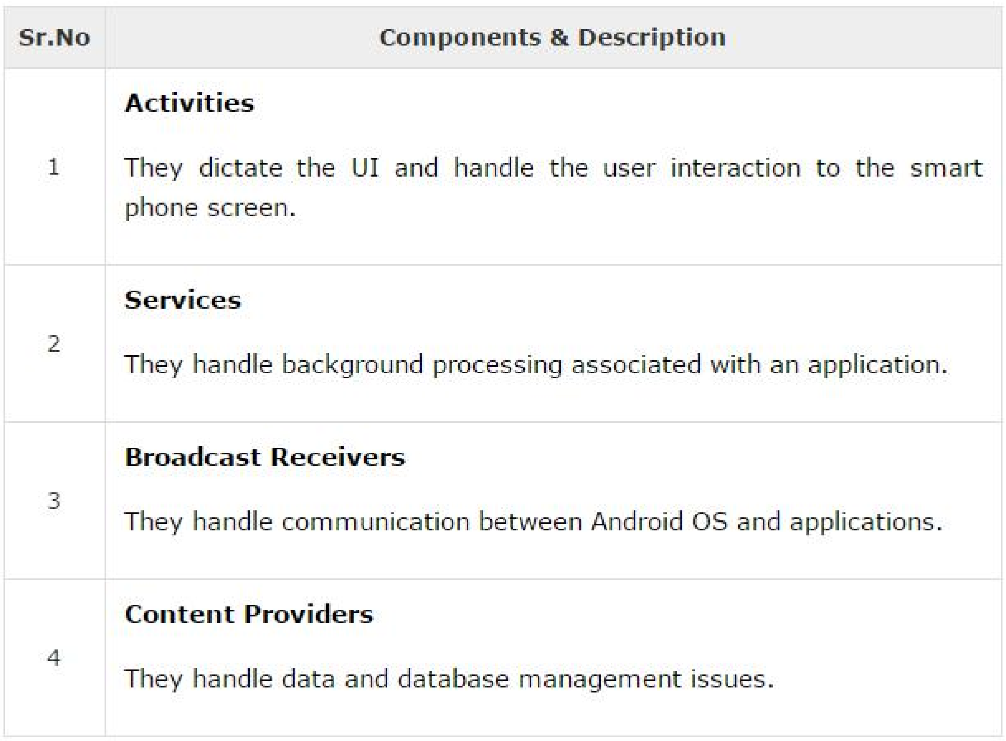
View System − An extensible set of views used to create application user interfaces.

Applications

You will find all the Android application at the top layer. You will write your application to be installed on this layer only. Examples of such applications are Contacts Books, Browser, Games etc.

Application components are the essential building blocks of an Android application. These components are loosely coupled by the application manifest file AndroidManifest.xml that describes each component of the application and how they interact.

There are following four main components that can be used within an Android application:



Sr.No	Components & Description
1	**Activities** They dictate the UI and handle the user interaction to the smart phone screen.
2	**Services** They handle background processing associated with an application.
3	**Broadcast Receivers** They handle communication between Android OS and applications.
4	**Content Providers** They handle data and database management issues.

Table 3.2.1: Components of Android

Activities

An activity represents a single screen with a user interface,in-short Activity performs actions on the screen. For example, an email application might have one activity that shows a list of new emails, another activity to compose an email, and another activity for reading emails. If an application has more than one activity, then one of them should be marked as the activity that is presented when the application is launched. An activity is implemented as a subclass of Activity class as follows −

```
public class MainActivity extends Activity {

        }
```

Services

A service is a component that runs in the background to perform long-running operations. For example, a service might play music in the background while the user is in a different application, or it might fetch data over the network without blocking user interaction with an activity. A service is implemented as a subclass of Service class as follows −

```
public class MyService extends Service {
}
```

Broadcast Receivers

Broadcast Receivers simply respond to broadcast messages from other applications or from the system. For example, applications can also initiate broadcasts to let other applications know that some data has been downloaded to the device and is available for them to use, so this is broadcast receiver who will intercept this communication and will initiate appropriate action. A broadcast receiver is implemented as a subclass of BroadcastReceiver class and each message is broadcaster as an Intent object.

```
public class MyReceiver  extends  BroadcastReceiver {
  public void onReceive(context,intent){}
}
```

Content Providers

A content provider component supplies data from one application to others on request. Such requests are handled by the methods of the ContentResolver class. The data may be stored in the file system, the database or somewhere else entirely.

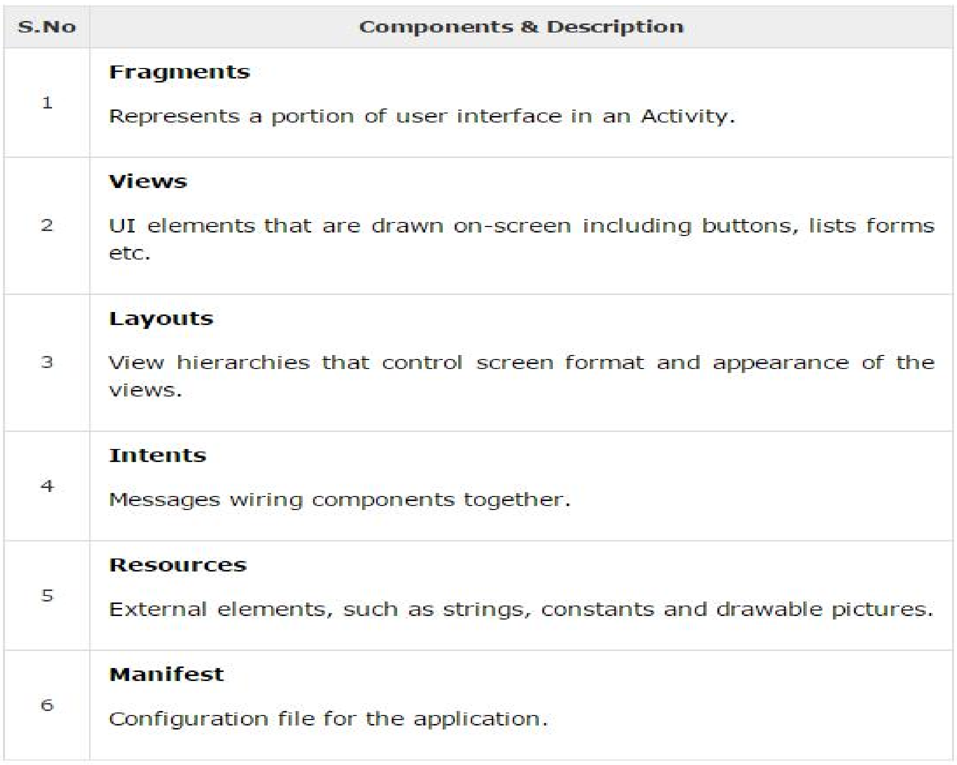
A content provider is implemented as a subclass of ContentProvider class and must implement a standard set of APIs that enable other applications to perform transactions.

```
public class MyContentProvider extends  ContentProvider {
          public void onCreate(){}
}
```

We will go through these tags in detail while covering application components in individual chapters.

Additional Components

There are additional components which will be used in the construction of above mentioned entities, their logic, and wiring between them. These components are –



S.No	Components & Description
1	**Fragments** Represents a portion of user interface in an Activity.
2	**Views** UI elements that are drawn on-screen including buttons, lists forms etc.
3	**Layouts** View hierarchies that control screen format and appearance of the views.
4	**Intents** Messages wiring components together.
5	**Resources** External elements, such as strings, constants and drawable pictures.
6	**Manifest** Configuration file for the application.

Table 3.2.2: Additional Components of Android

Create Android Application

The first step is to create a simple Android Application using Android studio. When you click on Android studio icon, it will show screen as shown below

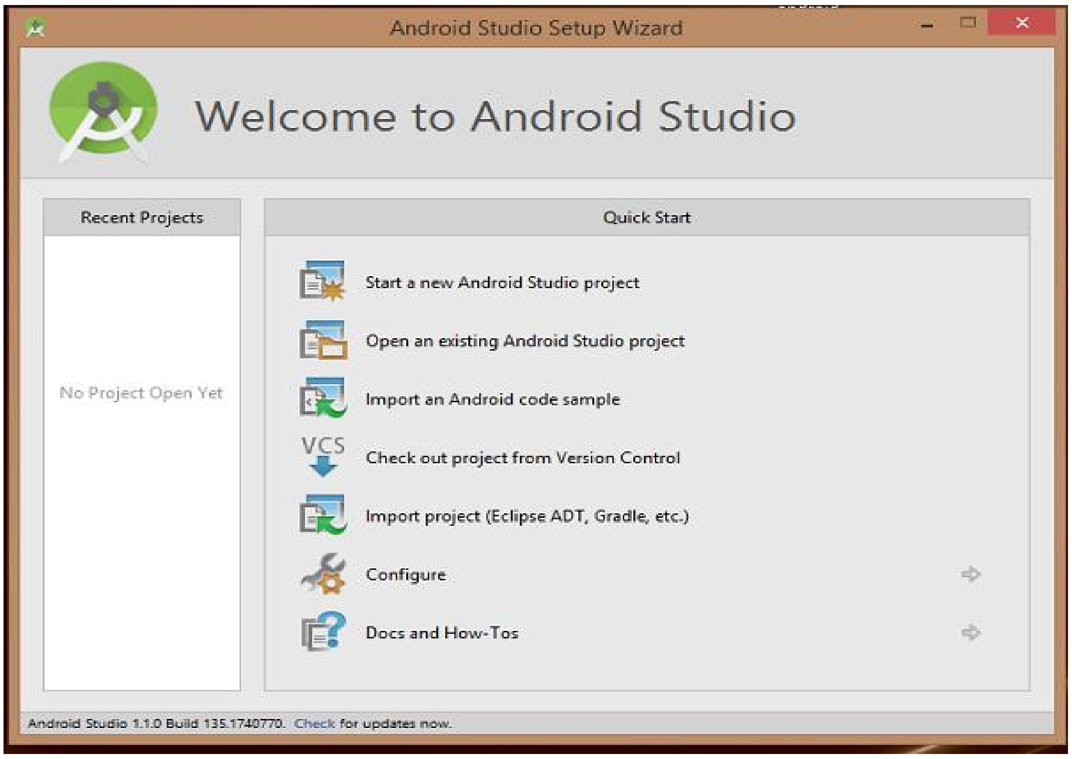
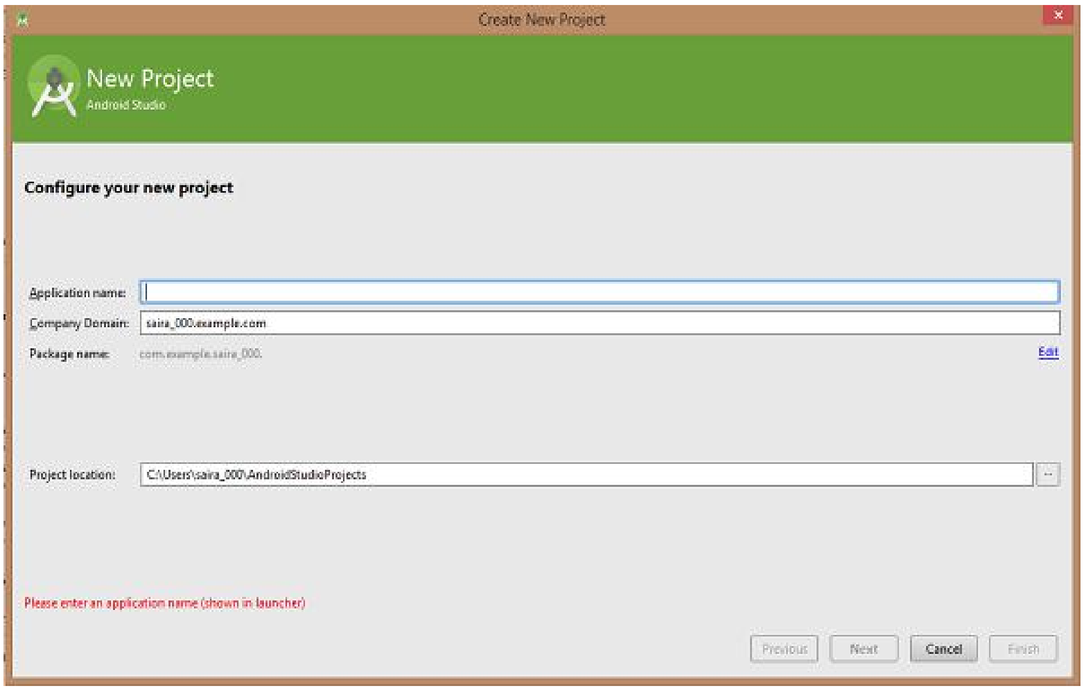

Android Studio Setup Wizard
Welcome to Android Studio
Recent Projects
No Project Open Yet
Quick Start
Start a new Android Studio project
Open an existing Android Studio project
Import an Android code sample
VCS
Check out project from Version Control
Import project (Eclipse ADT, Gradle, etc.)
Configure
Docs and How-Tos
Android Studio 1.1.0 Build 135.1740770. Check for updates now.

Fig 3.2.3: Android Studio Setup

You can start your application development by calling start a new android studio project. in a new installation frame should ask Application name, package information and location of the project.−



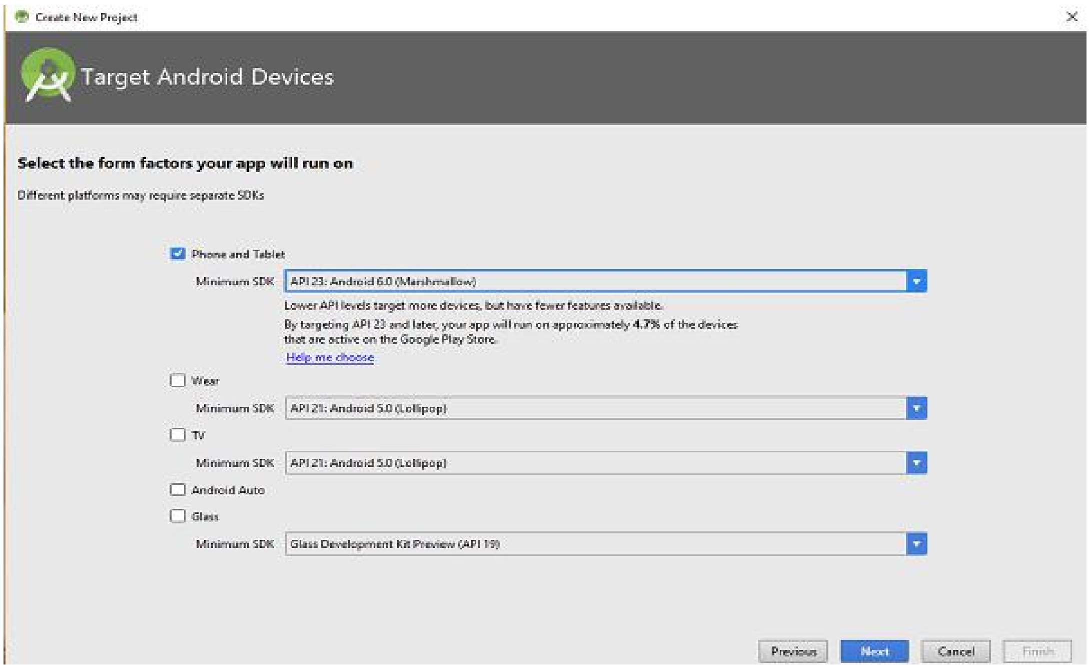
Configure your new project

Application name:
Company Domain: saira_000.example.com
Package name: com.example.saira_000. Edit
Project location: C:\Users\saira_000\AndroidStudioProjects

Please enter an application name (shown in launcher)

Fig:3.2.4 Creating New Android Project

After entered application name, it going to be called select the form factors your application runs on, here need to specify Minimum SDK, in our tutorial, I have declared as API23: Android 6.0(Mashmallow) −



Select the form factors your app will run on

Different platforms may require separate SDKs

[✓] Phone and Tablet
Minimum SDK: API 23: Android 6.0 (Marshmallow)
Lower API levels target more devices, but have fewer features available.
By targeting API 23 and later, your app will run on approximately 4.7% 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)

Fig 3.2.5 : Setting Up Android Device

The next level of installation should contain selecting the activity to mobile, it specifies the default layout for Applications.

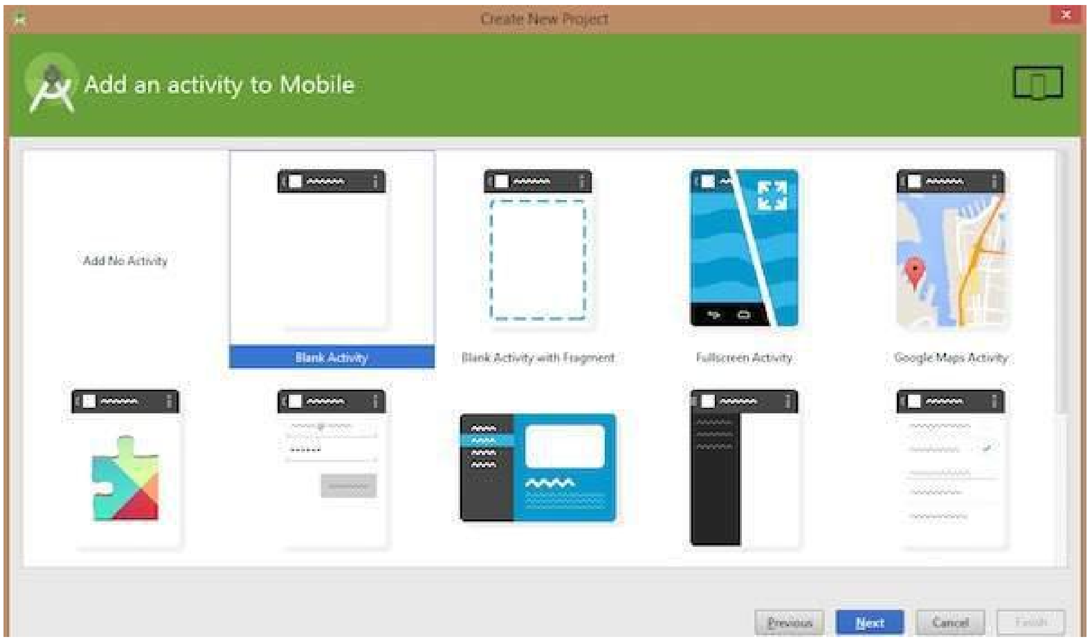


Fig:3.2.6 : selecting Activity

At the final stage it going to be open development tool to write the application code.

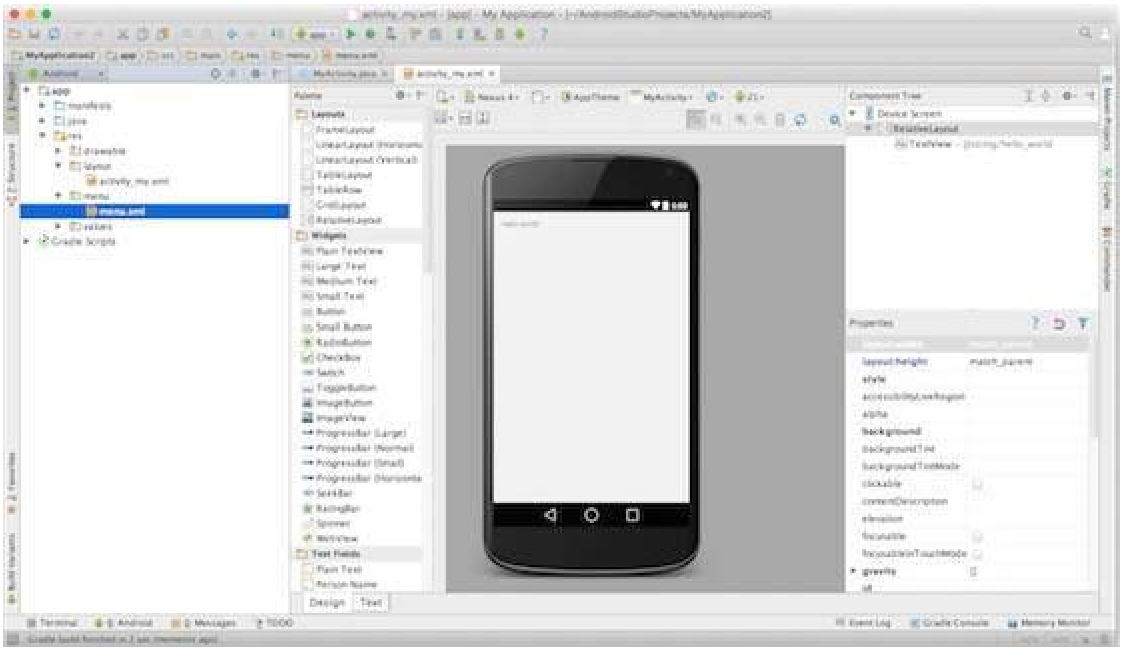


Fig 3.2.7 Android Emulator

Anatomy of Android Application

Before you run your app, you should be aware of a few directories and files in the Android project −

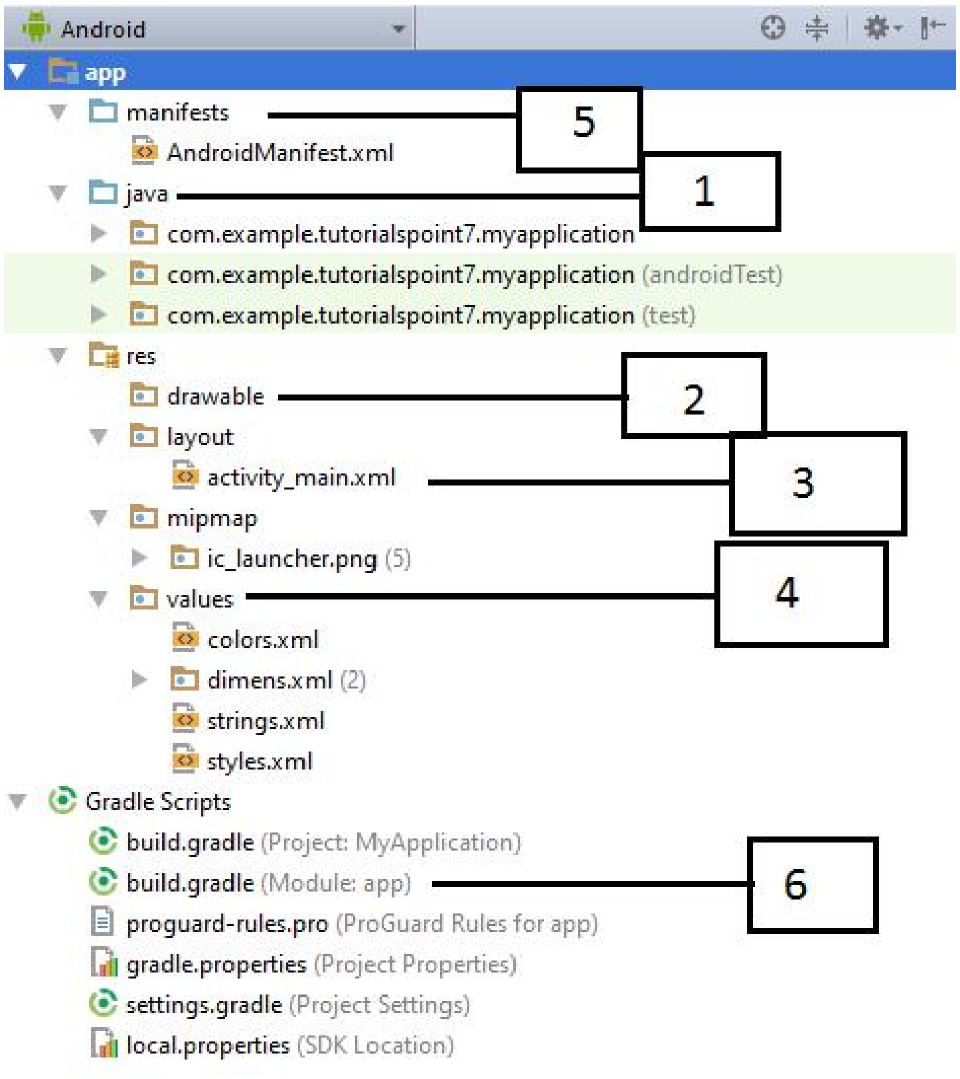
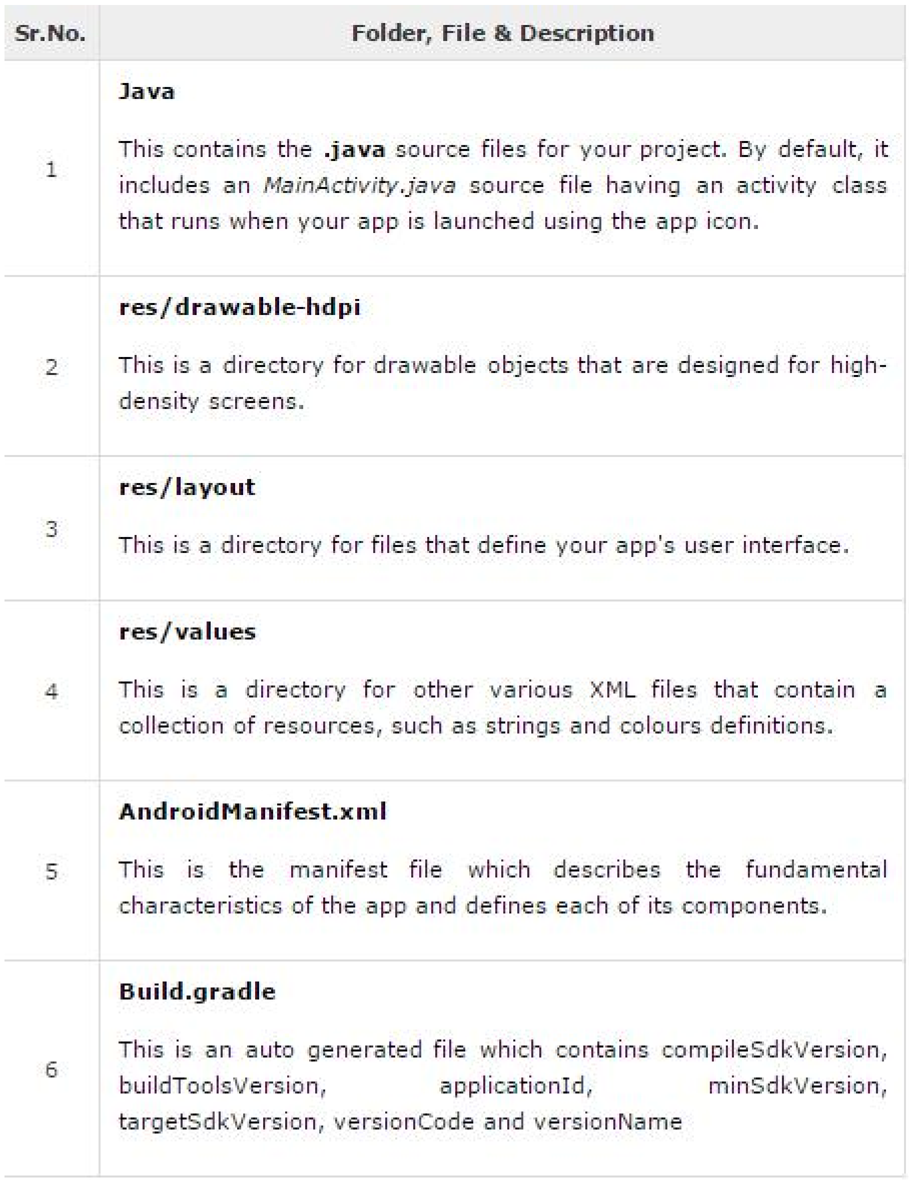

Android
app
manifests
AndroidManifest.xml
java
com.example.tutorialspoint7.myapplication
com.example.tutorialspoint7.myapplication (androidTest)
com.example.tutorialspoint7.myapplication (test)
res
drawable
layout
activity_main.xml
mipmap
ic_launcher.png (5)
values
colors.xml
dimens.xml (2)
strings.xml
styles.xml
Gradle Scripts
build.gradle (Project: MyApplication)
build.gradle (Module: app)
proguard-rules.pro (ProGuard Rules for app)
gradle.properties (Project Properties)
settings.gradle (Project Settings)
local.properties (SDK Location)
5
1
2
3
4
6

Fig 3.2.8 :Android File Directories



Sr.No.	Folder, File & Description
1	**Java** This contains the **.java** source files for your project. By default, it includes an *MainActivity.java* source file having an activity class that runs when your app is launched using the app icon.
2	**res/drawable-hdpi** This is a directory for drawable objects that are designed for high-density screens.
3	**res/layout** This is a directory for files that define your app's user interface.
4	**res/values** This is a directory for other various XML files that contain a collection of resources, such as strings and colours definitions.
5	**AndroidManifest.xml** This is the manifest file which describes the fundamental characteristics of the app and defines each of its components.
6	**Build.gradle** This is an auto generated file which contains compileSdkVersion, buildToolsVersion, applicationId, minSdkVersion, targetSdkVersion, versionCode and versionName

Table 3.2.3: Android File description

Running the Application

Let's try to run our **Hello World!** application we just created. I assume you had created your **AVD** while doing environment set-up. To run the app from Android studio, open one of your project's activity files and click Run icon from the tool bar. Android studio installs the app on your AVD and starts it and if everything is fine with your set-up and application, it will display following Emulator window −

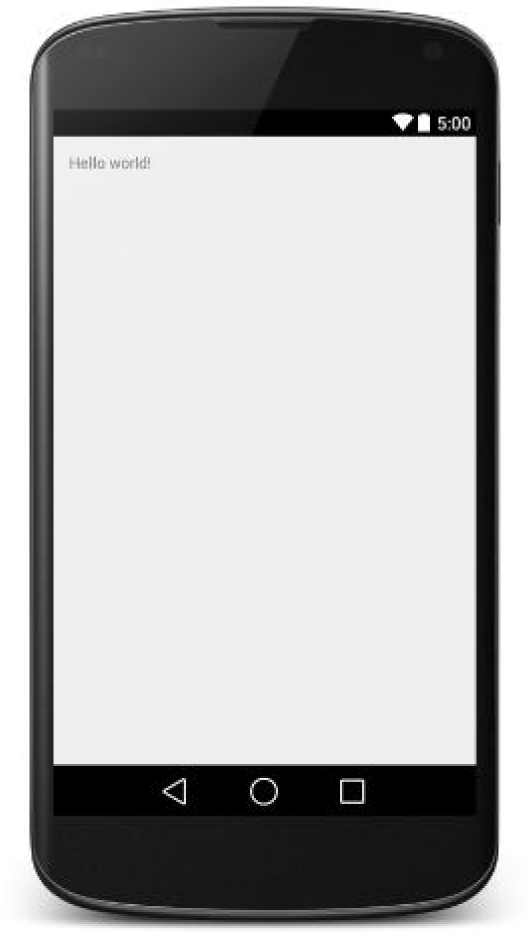


Fig 3.2.9: Running an Application

3.3 SOFTWARE REQUIREMENT SPECIFICATION

3.3.1 User Requirements

Requirement Specification plays an important role to create quality software solution. Requirements are refined and analysed to assess the clarity. Requirements are represented in a manner that ultimately leads to successful software implementation. Each requirement must be consistent with the overall objective.

3.3.2 Software Requirements

The software requirements specification is produced at the end of the analysis task. Software requirement is a difficult task, only decided after testing whether it fits the requirements.

Operating system	:	Windows 7
Coding Language	:	Java
IDE	:	Android Studio
Database	:	Firebase Cloud Database

3.3.3 Hardware Requirements

This is an IoT project so hardware plays an important role. Selection of hardware also plays an important role in existence and performance of any software. The size and capacity are main requirements.

Mobile which has below features:

*Android 1.5 or higher

SYSTEM DESIGN

4.1 INTRODUCTION

Software design sites at the technical kernel of the software engineering process and is applied regardless of the development paradigm and the area of application. Design is the first step in the development phase for any engineered product or system. The designer's goal is to produce a model or representation of an entity that will later be built. Beginning, once system requirements have been specified and analysed, system design is the first of the three technical activities – design, code and test that is required to build and verify software.

During design, progressive refinement of data structure, program structure and procedural details are developed, reviewed and documented. System design can be viewed from either technical or project management perspective. From the technical point of view, design is comprised of four activities – architectural design, data structure design, interface design, procedural design.

4.2 UML DIAGRAMS

The Unified Modelling Language (UML) can be a customary visual modelling language purported to be used for modelling business and similar processes, analysis, design, and implementation of software-based systems

UML can be a typical language for business analysts, software system package architects and developers accustomed describe, specify, design, and document existing or new business processes, structure and behavior of artifacts of software system package systems.

UML could also be applied to varied application domains (e.g., banking, finance, internet, aerospace, health care, etc.) it's going to be used with all major object and half software system package development methods and for various implementation platforms (e.g., J2EE, .NET).

UML can be a customary modelling language, not a software system package development methodology. UML 1.4.2 Specification explained that process:

provides steerage on the order of a team's activities, specifies what artifacts ought to be developed,

directs the tasks of individual developers and therefore the team as an entire, and offers criteria for observation and activity a project's merchandise and activities.

UML is designedly methodology freelance and can be applied inside the context of assorted processes. Still, it's best suited to be used case driven, unvarying and progressive development processes. Associate in Nursing example of such methodology is Rational Unified methodology (RUP).

UML is not complete and it isn't totally visual. Given some UML diagram, we are going to not ensure to understand drawn [*fr1] or behaviour of the system from the diagram alone. Some information are often designedly omitted from the diagram, some information pictured on the diagram might need altogether completely different interpretations, and a number of concepts of UML have no graphical notation within the least, so there is no due to depict those on diagrams.

For example, linguistics of multiplicity of actors and multiplicity of use cases on use case diagrams is not made public precisely among the UML specification and can mean either coincident or successive usage of use cases.

Name of academic degree abstract classifier is shown in italics whereas final classifier has no specific graphical notation, so there is no because of ensure whether or not or not classifier is final or not from the diagram.

There square measure a pair of broad categories of diagrams thus square measure all over again divided into sub-categories:

- ✓ Structural Diagrams
- ✓ Behavioral Diagrams

Structural Diagrams:

The structural diagrams represent the static facet of the system. These static aspects represent those components of a diagram that forms the most structure and so stable.

These static components are represents by categories, interfaces, objects, parts and nodes. The four structural diagrams are:

- Class diagram
- Object diagram
- Component diagram
- Deployment diagram

Behavioral Diagrams:

Any system will have 2 aspects, static and dynamic. therefore a model is taken into account as complete once each the aspects area unit lined totally.

Behavioral diagrams essentially capture the dynamic side of a system. Dynamic side are often any delineated because the changing/moving components of a system.

UML has the subsequent 5 forms of activity diagrams:

- ✓ Use case diagram
- ✓ Sequence diagram
- ✓ Collaboration diagram
- ✓ Statechart diagram
- ✓ Activity diagram

4.2.1 USE CASE DIAGRAMS

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

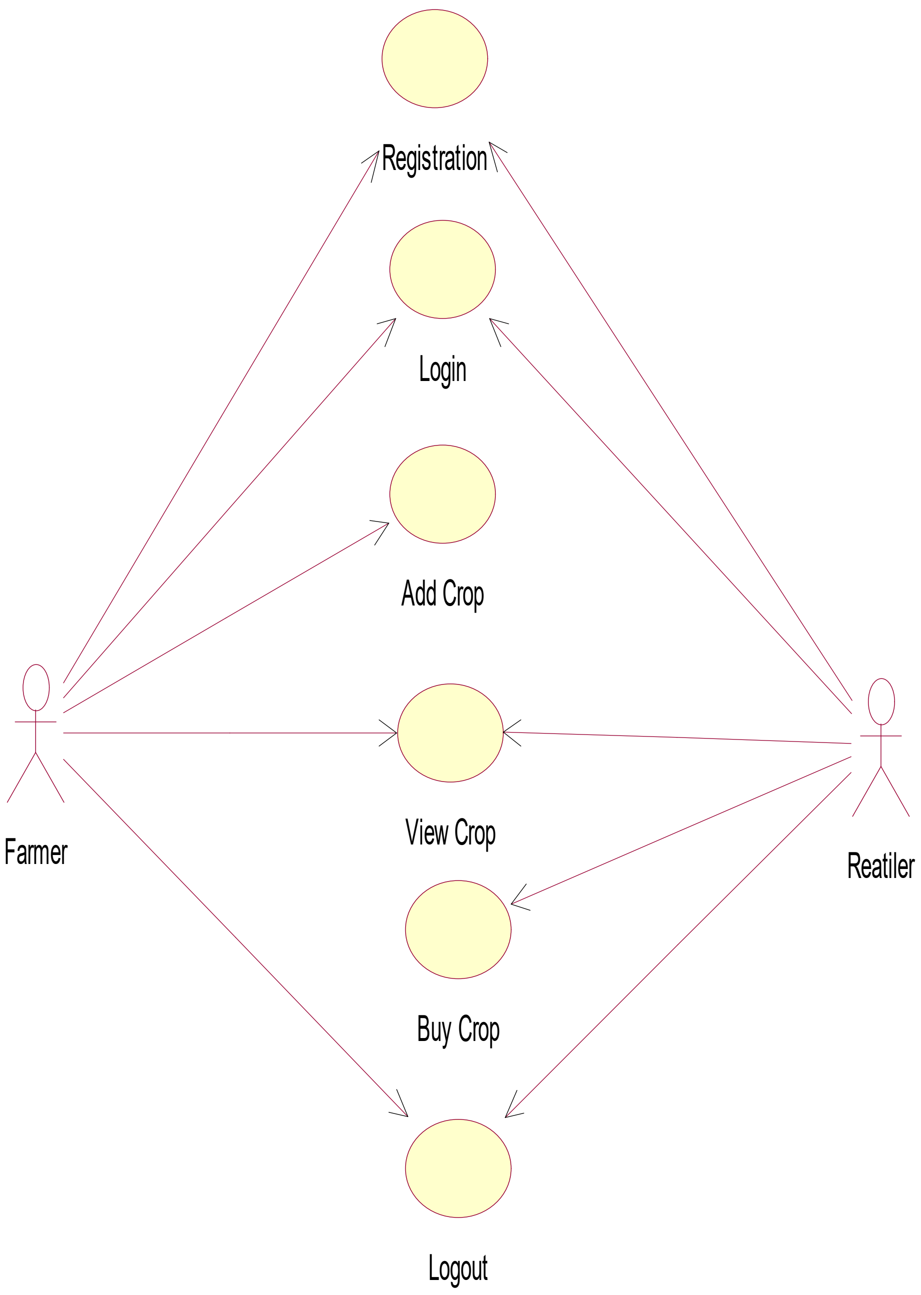


Fig 4.4.1 use case diagrams

4.2.2 CLASS DIAGRAMS

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

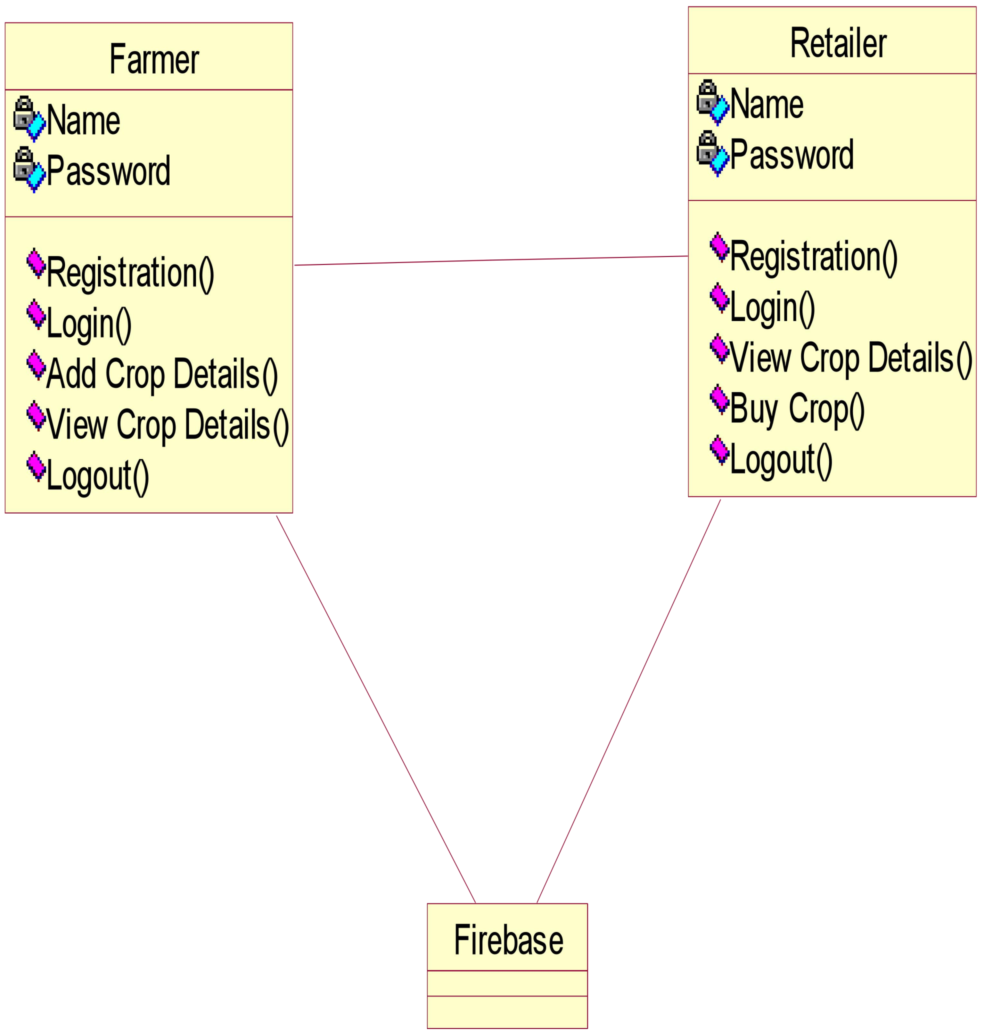


fig 4.4.2 class diagrams

4.2.3 sequence diagrams

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams

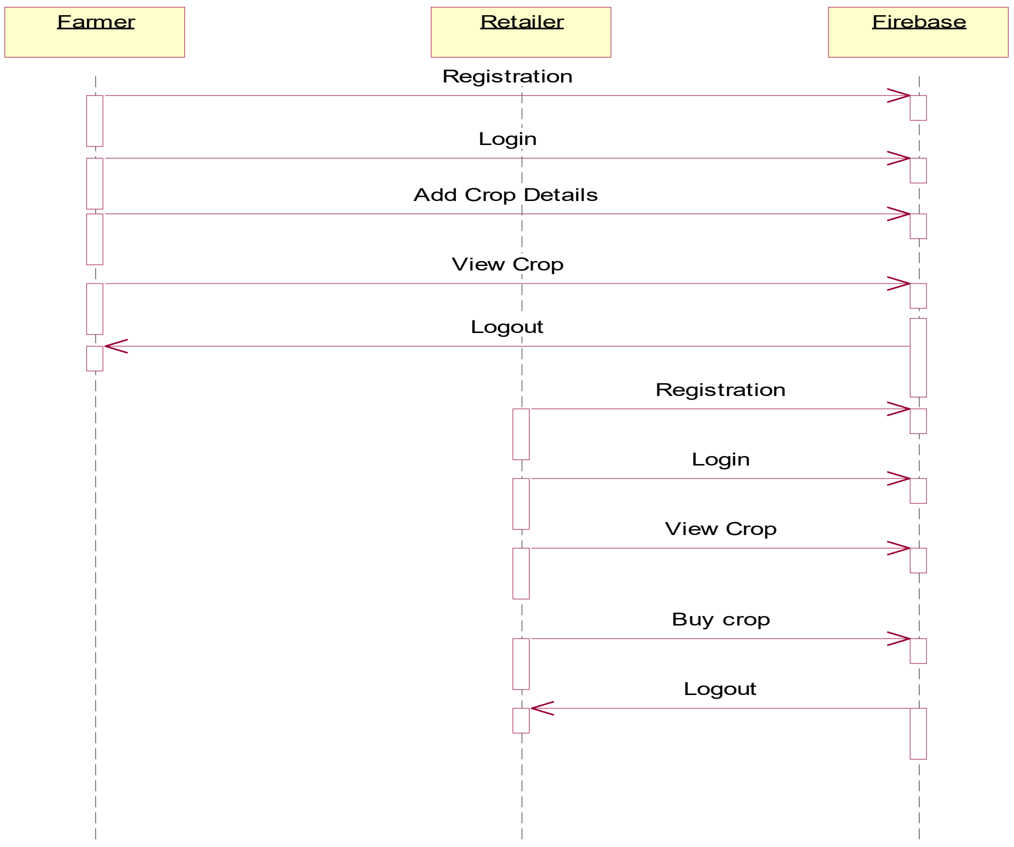


Fig 4.4.3 Sequence Diagrams

IMPLEMENTATION AND TECHNOLOGIES

5.1 MODULE DESIGN

The System Design Document describes the system requirements, operating environment, system and subsystem architecture, files and database design, input formats, output layouts, human-machine interfaces, detailed design, processing logic, and external interfaces.

MODULES:

- FARMER
- RETAILER

MODULE DISCRIPTION :-

FARMER

*In this module,the Farmer upload their Forms and manages them.

*the Forms are stored in Firebase Database, as light weight base.

*the Legal information of Farmer are held in secure

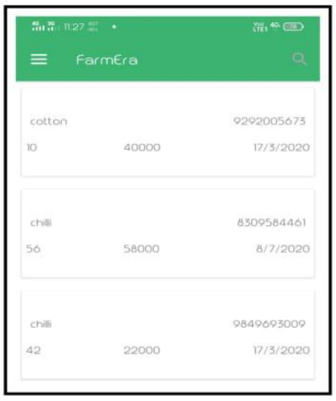
RETAILER

*Retailer can choose the crop forms as [er they want.

*The legal information of Retailer will be held in secure.

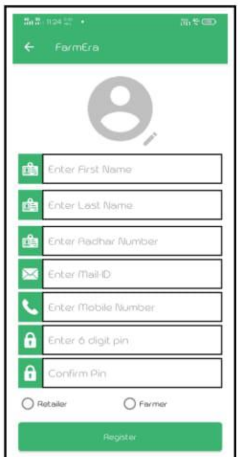
*the retailer can view the crop and select as they want.

5.2 SCREEN SHOTS


FarmEra

cotton		9292005673
10	40000	17/3/2020
chilli		8309584461
56	58000	8/7/2020
chilli		9849693009
42	22000	17/3/2020

form crops


FarmEra

Enter First Name

Enter Last Name

Enter Aadhar Number

Enter Mail-ID

Enter Mobile Number

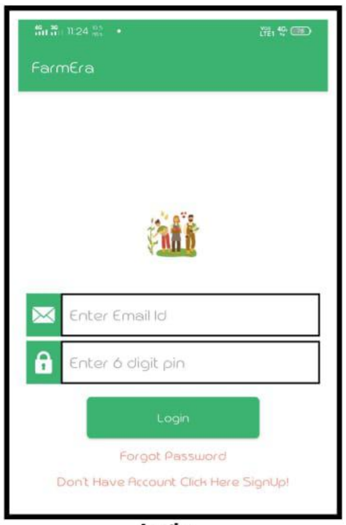
Enter 6 digit pin

Confirm Pin

○ Retailer ○ Farmer

Register

Registration


FarmEra

Enter Email Id

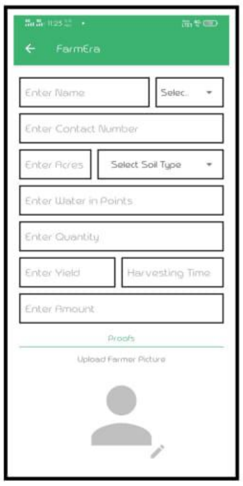
Enter 6 digit pin

Login

Forgot Password

Don't Have Account Click Here SignUp!

Login


FarmEra

Enter Name | Selec..

Enter Contact Number

Enter Acres | Select Soil Type

Enter Water in Points

Enter Quantity

Enter Yield | Harvesting Time

Enter Amount

Proofs

Upload Farmer Picture

Form

SYSTEM TESTING

The goal of testing is to acquire errors. Testing is that the technique of trying to get each possible error or weakness in an extremely work product. It provides the way to observe the practicality of parts, sub-assemblies, assemblies and or a finished product it is the technique of effort code with the concentrating of guaranteeing that the software meets its requirements and user hopes and does not fail in an undesirable manner. There are numerous sorts of check. Every check sort reports a designated testing demand.

Testing objectives:

The key objective of testing is to determine a mass of errors, systematically and with minimum effort and time. Stating formally, we can say, testing is a process of executing a program with resolved of discover an error.

- A successful test is one that determines an as however undiscovered error.
- A good test case is one that has possibility of discover an error, if it exists.
- The test is insufficient to detect possibly present errors.
- The software more or less approves to the quality and unswerving standards.

6.1 Types of Testing

The basic levels of Testing:

$$\begin{tikzcd}
\text{Client needs} \arrow[r, leftrightarrow] \arrow[d] & \text{acceptance testing} \\
\text{requirements} \arrow[r, leftrightarrow] \arrow[d] & \text{system testing} \arrow[u] \\
\text{Design} \arrow[r, leftrightarrow] \arrow[d] & \text{integration testing} \arrow[u] \\
\text{Code} \arrow[r, leftrightarrow] & \text{unit testing} \arrow[u]
\end{tikzcd}$$

Fig. 6.1 Levels of Testing

Functional Testing

Real tests give efficient protests that functions tested are attainable as specific by the business and technical requirements, system documentation, and user manuals.

Functional testing is focused on the subsequent items:

Valid Input : known categories of valid input should be accepted.

Invalid Input : known categories of invalid input should be rejected.

6.2 TYPES OF TESTS

Unit testing:

A unit is the smallest piece of source code that can be tested. It is also known as a module which consists of numerous lines of code that are processed by a single programmer. The key purpose of performing unit testing is to expose that a particular unit doesn't satisfy the specified functional requirements and also to show that the structural implementation is not like to the projected structure designed.

Integration testing:

Integration tests are intended to test incorporated programming segments to figure out whether they really keep running as one system. Testing is occasion driven and is more worried with the fundamental result of screens or fields. Reconciliation tests exhibit that in spite of the fact that the parts were separately fulfillment, as appeared by effectively unit testing, the blend of segments is right and comprised. Integration testing is specifically aimed at revealing the problems that rise from the mixture of components.

Functional test

Functional tests give efficient challenges that capacities tried are accessible as determined by the business and specialized necessities, framework documentation, and client manuals Functional testing is centered on the following items:

Valid Input: identified classes of valid input must be accepted.

Invalid Input: identified classes of invalid input must be rejected.

Functions: identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Association and arrangement of practical tests is centered around prerequisites, key capacities, or unique experiments. Likewise, efficient scope relating to recognize Business procedure streams; information fields, predefined procedures, and progressive procedures must be considered for testing. Before utilitarian testing is finished, extra tests are distinguished and the powerful estimation of current tests is resolved.

System Test

System testing guarantees that the entire coordinated programming framework meets prerequisites. It tests a design to guarantee known and unsurprising results. A sample of framework testing is the arrangement situated framework combination test. Framework testing depends on procedure portrayals and streams, stressing pre-driven procedure connections and mix focuses.

White Box Testing

It is a testing in which the product analyzer has information of the internal workings, structure and dialect of the product, or if nothing else its motivation. It is reason. It is utilized to test ranges that can't be gotten a handle on from a discovery level.

Black Box Testing

It is the testing the product with no information of within workings, structure or dialect of the part being tried. Discovery tests, as most different sorts of tests, must be composed from a complete source report, for example, prerequisite or necessities archive, for example, determination or necessities record. It is a trying in which the product under test is dealt with, as a discovery .you can't "see" into it. The test gives inputs and reacts to yields without considering how the product functions.

Unit Testing:

Unit testing is by and large appeared as a major aspect of a joined code and unit test period of the product lifecycle, in spite of the fact that it is not exceptional for coding and unit testing to be directed as two unmistakable stages.

Test strategy and approach

Ground testing will be done physically and functional tests will be inscribed in detail.

Test objectives

- All field admissions essentially work appropriately.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Validate that the accesses are of the correct format
- No duplicate entries should be allowed
- Entire links must gross the user to the accurate page.

Integration Testing

Software integration testing is the incremental combination analysis of two or more joint software components on a single platform to generate failures created by boundary faults.

The task of the integration test is to design those components or s/w applications, e.g. modules in a software system or – one step up – software presentations at the company level

– interact without faults.

Test Results: All the test cases stated above passed effectively. No defects met.

Acceptance Testing

User Acceptance Testing is a serious phase of any project and needs important contribution by the end user. It also guarantees that the system encounters the functional requirements.

Test case id	Test case description	Actual value	Entered value	Status
1	Register user details in registration page	Fill all the fields while registering user	All the fields are filled	Pass
2	Give user name in text box	User name must be given in alphabets	User name given in alphabets and numeric values	Fail
3	Password to be entered in password box	Password must be given correctly	Password is entered wrongly	Fail
4	Phone number must be entered in phone number box during registration	Phone number must be given in 10 digits	Phone number given in 10 digits	Pass
5	Validating the functionality of Browse button	System should select the corresponding file	selected the file what we expected	Pass

Fig 6.2: Testcase Template

Test Results: All the test cases stated above passed effectively. No defects met.

7.CONCLUSION

Here, We have designed a project “Agriculture Yield Interaction between Farmers and Retailers (FarmEra) " using ANDROID.

All the application codes are in android and fire base backend.

Agriculture Yield Interaction between Farmers and Retailers (FarmEra) is very beneficial for both farmer and retailer we presented a project which implements both proposed system and satisfies all the functional Requirements.

8 BIBILOGRAPHY

REFRENCES

1. https://www.tutorialspoint.com/android/android_resources.htm
2. https://developer.android.com/guide/index.html
3. https://www.engineersgarage.com/articles/what-is-android-introduction
4. http://www.beginandroid.com/intro.shtml
5. http://www.gcflearnfree.org/androidbasics/intro-to-android-devices/1/
6. https://en.wikipedia.org/wiki/Android
7. "Industry Leaders Announce Open Platform for Mobile Devices". Open Handset Alliance. November 5, 2007. Retrieved March 12, 2017.
8. Kaplan, E. (1996). Understanding GPS - Principles and Applications. Boston: Artech House.
9. Critical Technologies Institute (1995)- A Policy Direction for the Global Positioning
   System: Balancing National Security and Commercial Interests.
10. Institute of Navigation. Global positioning systems: papers published in navigation. Alexandra, VA, Institute of Navigation. 1984-1999. 7 v.
11. L. Ma, L. Gu, J. Wang, "Research and Development of Mobile Application for Android Platform,"International Journal of Multimedia and Ubiquitous Engineering, vol. 9, pp. 187–198, 2014.
12. L. Priyanka, A. Priyanka, K. Monali, M. Sandhya, "Smart Shopping: Location Based An Android Application,"Imperial Journal of Interdisciplinary Research, vol. 2, issue 1, 2016.