A Project Report
On
"Gaze"

Submitted for the partial fulfillment of the requirement for the award of the
Degree of Bachelor Of Technology
in
Information Technology

Submitted By:

Aksa Zehra 1406813012
Manan Gupta 1406813037
Marvi Khan 1406813040

Under the guidance of
Mr. Sumit Singh Siddhu
(Assistant professor, IT)

Department of Information Technology
MEERUT INSTITUTE OF ENGINEERING & TECHNOLOGY
Meerut – 250005
Approved by A.I.C.T.E.

Affiliated to

APJ Abdul Kalam Technical University, Lucknow

(Batch: 2014 – 2018)

<u>**Gaze**</u>

Submitted By:

Aksa Zehra	1406813012
Manan Gupta	1406813037
Marvi Khan	1406813040

Department of Information Technology

MEERUT INSTITUTE OF ENGINEERING & TECHNOLOGY

Meerut

APRIL, 2018

TABLE OF CONTENT

Page

CERTIFICATE………………………………………………………………………… v
DECLARATION………………………………………………………………………. vi
ACKNOWLEDGEMENTS……………………………………………………………. vii
ABSTRACT…………………………………………………………………………. viii
LIST OF FIGURES…………………………………………………………………….. ix
CHAPTER 1 INTRODUCTION…………………………………………………............. 1
1.1. PURPOSE……………………………………………………………... 1
1.2. PROBLEM STATEMENT……………………………………………………. 2
CHAPTER 2 SOFTWARE REQUIREMENT SPECIFICATIONS……………….......... 3
2.1. INTRODUCTION……………………………………………..……………… 3
2.1.1. PURPOSE………………………………………………………………. 3
2.1.2. SCOPE………..………………………………………………………. 3
2.2. OVERALL DESCRIPTION………………….…………………………….. 4
2.3. FEASIBILTY STUDY…….…………………..……………………………. 4
2.3.1. TECHNICAL FEASIBILITY……….…………………………..……. 4
2.3.2. ECONOMICAL FEASIBILITY………..……………….……………. 4
2.3.3. OPERATIONAL FEASIBILITY………..……………….…………… 5
CHAPTER 3 SOFTWARE AND HARDWARE REQUIREMENTS……………........ 6
3.1. SYSTEM SPECIFICATIONS…….……………………………………………. 6
3.1.1. HARDWARE REQUIREMENTS…………….…………………..….. 6
3.1.2. SOFTWARE REQUIREMENTS..……..…………….…………….. 6
3.2. TECHNOLOGIES USED…………………………………………………. 7
3.2.1. ANDROID TECHNOLOGIES……….…………………………..………. 7
3.2.2. FEATURES..……..……………….……………………………………. 8
3.2.3. APPLICATION FRAMEWORK……….……………………………..…….. 9
3.2.4. DALVIK VIRTUAL MACHINE..……..…………….…………….... 9
3.2.5. INTEGRATED BROWSER……….……………………..………….. 9
3.2.6. OPTIMIZED GRAPHICS..……..…………….…………………….... 10
3.2.7. JSON……….……………………..……………………………………... 10
3.2.8. JAVA VIRTUAL MACHINE..……..…………….…………………….. 10
3.2.9. DEVELOPMENT ENVIRONMENT..……..…………….…………….. 10

3.3. ARCHITECTURE………………………………………………………….......... 14
3.3.1. ANDROID STUDIO 2.1…….………………………..……………..... 15
3.3.2. MOBILE DRIVERS…….………………………..………………...... 15
3.3.3. JAVA..……..……………………………………..……………...... 15
3.3.4. JDK……..………………………………………….……..……...... 16
3.3.5. XML…….……………………………………………….……...... 16
CHAPTER 4 DEVELOPMENT PLATFORM…………………………..................... 18
4.1. ANDROID SDK…………………………………………………….................. 19
4.2. ANDROID DEBUG BRIDGE…………………………………….................... 21
4.3. ANDROID STUDIO………………………………………………….................. 23
4.4. ADOBE PHOTOSHOP…………………………………………….................. 26
CHAPTER 5 SOFTWARE DESIGN DOCUMENT... 27
5.1. USE CASE…………………………………………………………....... 27
5.2. DFD………………………………………………………………. 28
5.3. ER DIAGRAM……………………………………………………… 30
5.4. FLOWCHART………………………………………………..….. 31
CHAPTER 6 IMPLEMENTATION……………………………………................ 32
6.1. PROJECT DEFINITION.. 32
6.2. FRAGMENTS OF GAZE APPLICATION…………………….....……...... 34
CHAPTER 7 TESTING……………………………………………….................... 40
7.1. TEST PLAN…………………………………………..……….......... 40
7.1.1. UNIT TESTING…….……………………………..…….......... 40
7.1.2. INTEGRATION TESTING..……..………………….…………......... 41
7.1.3. VALIDATION TESTING OR SYSTEM TESTING……..……......... 41
7.1.4. ACCEPTANCE TESTING…….…………………..……........... 42
7.1.5. STATIC AND DYNAMIC TESTING..……..……………………......... 42
7.1.6. OUTPUT TESTING…….…………………..……………........... 43
CHAPTER 8 CONCLUSION…………………………………………….............. 44
CHAPTER 9 FUTURE SCOPE AND LIMITATIONS……………………................... 45
9.1. FUTURE SCOPE……………………………………………..…............ 45
9.2. LIMITATIONS…………………………………………..…............... 45
CHAPTER 10 APPENDIX………………………………………………............... 46
10.1. APIS……………………………………………………………................ 46

10.1.1. DATAPARSER……………………………………………………..... 46
10.1.2. DOWNLOADURL…………………………………………….... 49
10.1.3. GETNEARBYPLACES……………………………..……….. 51
10.2.4. NEARBYPLACES…………………………………………..….. 52
10.2. JAVA FILES FOR FIRST ACTIVITY……………………………….......... 53
10.2,1. LISTACTIVITY…………………………………………………….... 53
10.2.2. MYRECYCLERADAPTER ………………………………………. 55
10.2.3. RECYCLERTOUCHLISTENER…………………………….……… 57
10.2.4. CATEGORY………………………………..…………………….... 59
10.2.5. CATEGORYLIST…………………………………………........... 60
10.3. JAVA FILES FOR SECOND ACTIVITY…………………………………...... 61
10.3.1. MAINACTIVITY……………………………………………………... 61
10.3.2. VIEWPAGERADAPTER………………………………………….. 68
10.3.3. COORDINATE………………………………………………….. 69
10.3.4. MYMAPFRAGMENT…………………………………..……........ 70
10.3.4. MYPLACESLISTFRAGMENT……………………………...….. 73
10.4. XML FILES………………………………………………………….......... 76
10. 4.1. ACTIVITYLIST…………………………………………………...... 76
10.4.2. ACTVITYLISTITEM…………………………………………………... 76
CHAPTER 11 REFERENCES…………………………………………………….......... 78

CERTIFICATE

This is to certify that Project Report entitled "Gaze" which is submitted by Aksa Zehra , Marvi Khan and Manan Gupta in partial fulfillment of the requirement for the award of degree B. Tech. in Department of Information Technology of APJ Abdul Kalam Technical University, is a record of the candidate own work carried out by him under my/our supervision. The matter embodied in this thesis is original and has not been submitted for the award of any otherdegree.

Date:

Supervisor:

Mr. Sumit Singh Siddhu (IT)

DECLARATION

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

Name: Aksa Zehra	Name: Manan Gupta
Roll No.: 1406813012	Roll No.: 1406813037
Signature:	Signature:
Date:	Date:

Name: Marvi Khan

Roll No.: 1406813040

Signature:

Date:

ACKNOWLEDGEMENT

It gives us a great sense of pleasure to present the report of the B. Tech Project undertaken during B. Tech. Final Year. We owe special debt of gratitude to Professor Sumit Singh Siddhu, Department of Information Technology, Meerut Institute Of Engineering And Technology, Meerut for his constant support and guidance throughout the course of our work. His sincerity, thoroughness and perseverance have been a constant source of inspiration for us. It is only his cognizant efforts that our endeavors have seen light of the day.

We also take the opportunity to acknowledge the contribution of Professor Ravindra Chauhan, Head, Department of Information Technology, Meerut Institute Of Engineering And Technology, Meerut for his full support and assistance during the development of the project.

We also do not like to miss the opportunity to acknowledge the contribution of all faculty members of the department for their kind assistance and cooperation during the development of our project. Last but not the least, we acknowledge our friends for their contribution in the completion of the project.

Name: Aksa Zehra	Name: Manan Gupta
Roll No.: 1406813012	Roll No.: 1406813037
Signature:	Signature:
Date:	Date:

Name: Marvi Khan

Roll No.: 1406813040

Signature:

Date:

ABSTRACT

This is the project report on "Gaze android application". During the making/developing of this project we explored new ideas and functionality behind the working of the android studio.

This project is the output of our planning, schedule, programming skill and the hard work, and this report reflects our step taken at various levels of programming skill, planning and schedule.

We have learnt a lot during this project and liked the improvement in our testing skills and deep concepts related to these kinds of projects.

Our project is "Gaze app". This is a location-based augmented reality android application software which helps people to find places in the most suitable way . It is useful in a way that it makes an easier way to delve the world digitally.

In this application, you simply select one of the pre-defined categories, hold up your smartphone and watch as the world around you populates with digitally generated signposts.

In this project, we have basically two modules . The first module includes the Google places.

Google places lets you to find nearby places . It helps to find useful places around you - such as restaurants, ATMs, hotels, movie theatre and more. It gives more information and comprehensive view about the places around the user.

The second module is Augmented Reality or AR Core. Augmented Reality is helpful in integrating the digital information with the user's environment in real time. It uses your phone's camera to drive intuitively to places of your interest .

LIST OF FIGURES

S No.	Description
3.1	An Android Device
3.2	Android OS Features
3.3	Different Version of Android OS
3.4	Android OS Architecture
4.1	SDK Manager
4.2	Android Debug Bridge
4.3	Android Studio
4.4	Adobe Photoshop
5.1	Use - Case Diagram
5.2	Context Diagram
5.3	Level 1 DFD Diagram
5.4	ER Diagram
5.5	Flow Chart
6.1	List Activity in Gaze Application
6.2	Map View of Restaurant Section
6.3	List Activity
6.4	Camera View of Restaurant Section
6.5	Map Fragment
6.6	List Fragment
6.7	Camera Fragment

CHAPTER 1

INTRODUCTION

1.1. Purpose

The era of mobile technology opens the windows to the android app. The websites are vanishing and mobile phones are emerging. It’s the time to change from conventional websites to mobile apps, which has become the part of our daily routine. Android is the emerging device in today’s world. It has occupied large percentage in the mobile use. Android has not only be used just for the making call but several application can be installed in it and user can get benefited through it.
With keeping view of the emerging growth and use of android, we are introducing
“GAZE”, the android application software which would provide a unique way to find useful places around you - such as restaurants, ATMs, hotels, movie theatre and more. Its sole purpose is to explore the world digitally. Gaze is a location-based augmented reality app that brings Google Maps to life, so all you need to do is hold up your phone and the streets in front of you will be transformed with information about nearby facilities.

Gaze uses your phone’s camera to give you a completely new way of exploring a place, anywhere in the world. To use the app you simply select one of the pre-defined categories, hold up your Smartphone and watch as the world around you populates with digitally generated signposts. If you want more information, you simply click on the signpost to get full contact details, user reviews and directions.

1.2. Problem Statement

Imagine you are just visiting a city like Delhi for the first time on a travel tour and you have no idea that where to eat the delicious food as per your choice and also satisfy your pockets ? You are standing at a place and have no idea about the nearest hang out spots. What will you do?

On the other hand if you urgently need cash and want to find an ATM nearest to you. So for all the above situations like above, there is a need for a mobile application which could help us to find useful places around us and also tell us that how near we are to a place as we make our way towards it.

Also while visiting a place, nearby locations can be found but it can be difficult to match the maps with what we see. Thus we need an application which let us find places in the way most suitable to us..

CHAPTER 2

SOFTWARE REQUIREMENT SPECIFICATION

2.1. Introduction

2.1.1. Purpose

The document is written for the clients, the GAZE android application, toexplicitly describe the software product that is to be delivered. Keeping in view the emerging growth and use of android technology, we are introducing GAZE, the android application which would be helpful to find nearby places.

Its sole purpose is to explore the world digitally. The application provides a unique way to find useful places around you.

2.1.2. Scope

The document is meant to describe the entire product, its intended features andprescribed use. Based on determinations made by the requirements elicitation team. It is determined that the intended user of this system is all the people of rural as well urban areas, This android application lets you find places in the way most suitable to you. It can be in the form of lists, maps or augmented reality. The augmented reality feature combined with geo-tagging makes it easy to know which direction you should be going and the app makes it easy to quickly access information about the facilities around you.

The application ensures that it becomes easier for the person to find information about the place he/she want to visit.

The requirements were gathered within the specified time period of 2 weeks. There was no cost associated with the gathering of these requirements, and no special software was used to develop them.

2.2. Overall Description

Gaze is a location-based augmented reality application. This application sources its information from Google Places so you can be guaranteed that the app is comprehensive and that there's a decent pool of user reviews, no matter where you are in the world. It lets you check out user reviews, directions, phone numbers, opening hours, price tier, photos and more just from the virtual signboards. You can also choose between camera, list and map views to navigate. You can get the real-time distance to a point of interest and find out how near you are to a place even as you make your way towards it.

2.3. Feasibility Study

Feasibility study is an assessment of the practicality of a proposed project or system. It aims to objectively and rationally uncover the strengths and weaknesses of the proposed system, opportunities and threats present in the environment, the resources required to carry through, and ultimately the prospects for success. In its simplest terms, the two criteria to judge feasibility are cost required and value to be attained. Generally, feasibility studies precede technical development and project implementation.

The key consideration in feasibility analysis is:

- Technical Feasibility
- Economic Feasibility
- Operational Feasibility

2.3.1. Technical Feasibility

Technical feasibility raises questions like, is it possible that thework can be done with the current equipment; software technology required that could be used to develop the project. The project can be developed by using various technologies like Java and XML. Thus,project is technically feasible

.

2.3.2. Economic Feasibility

It looks at the financial aspects of the project. It determineswhether the management has enough resources and budget to invest in the proposed system and the estimated time for the recovery of cost incurred. It also determines whether it is worthwhile to invest the money in the proposed project. Economic feasibility is determined by the means of cost benefit analysis. This project requires no investment for selling and buying product apart from the product cost. In case of this project, an economically feasible solution is assumed.

2.3.3. Operational Feasibility

It deals with the user friendliness of the system, i.e. will thesystem be used if it is developed and implemented? or will there be resistance from users?
Generally, all users are open to android applications and internet and they can easily understand the application. Therefore, the project is operationally feasible.

CHAPTER 3

SOFTWARE AND HARDWARE REQUIREMENTS

3.1. System Specifications

3.1.1. Hardware Requirements

The project “GAZE” requires following hardware forits successful implementation.

Processor	:	I3 or above
RAM	:	8 GB or above
Hard Disk	:	20 GB or above
Monitor	:	TFT or LCD

3.1.2. Software Requirements

The project “GAZE” requires following software forits successful implementation.

Operating system	:	Windows
Programming Language	:	JAVA, XML& JSON
IDE	:	Android Studio
Mobile Driver	:	As per the mobile device available

3.2. Technologies Used

3.2.1. Android Technology

Figure3.1. An Android device

Android is a mobile phone operating system. It was originally developed by Android Inc., which was acquired by Google in July 2005. Today, development is overseen by the

Android Open Source Project (AOSP), led by Google. The AOSP is “tasked with the maintenance and further development of Android”.

As of the 3rd Quarter of 2010 Android has a market share of 25% making it the second most popular phone operating system of the market (second only to Nokia’s Symbian). This is a major rise from the 3.5percent share Android had in 3Q200.

Android, Inc. was founded in Palo Alto, California, United States 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, in Rubin's words "...smarter mobile devices that are more aware of its owner's location and preferences. Despite the obvious 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.

Google acquired Android Inc. on August 17, 2005, making Android Inc. a wholly owned subsidiary of Google. Key employees of Android Inc., including Andy Rubin, Rich Miner and Chris White, stayed at the company after the acquisition

3.2.2. Features

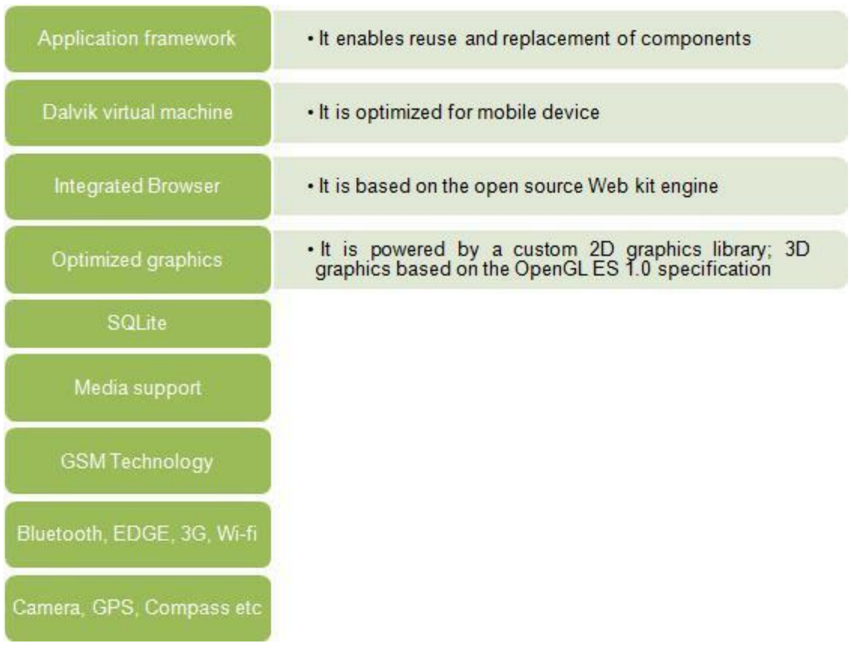

Application framework
• It enables reuse and replacement of components
Dalvik virtual machine
• It is optimized for mobile device
Integrated Browser
• It is based on the open source Web kit engine
Optimized graphics
• It is powered by a custom 2D graphics library; 3D graphics based on the OpenGL ES 1.0 specification
SQLite
Media support
GSM Technology
Bluetooth, EDGE, 3G, Wi-fi
Camera, GPS, Compass etc

Figure 3.2 Android OS Feature

World is contracting with the growth of mobile phone technology. As the number of users is increasing day by day, facilities are also increasing. Starting with simple regular handsets which were used just for making phone calls, mobiles have changed our lives and have become part of it. Now they are not used just for making calls but they have innumerable uses and can be used as a Camera, Music player, Tablet PC, T.V., Web browser etc. and with the new technologies, new software and operating systems are required.

Today Android is very popular among the user. Today the mobiles are not only thought to be used in the just for communication but is a mini computer which has the computing and manipulating capacity as that of computers. The wide use and popularity of the android is due to the various features available in it.

3.2.3. Application Framework

It is used to write application for Android. Unlike otherembedded mobile environment, Android application are all equal, for instance ,an application which come with the phone are no different than those that any developer writes. The framework is supported by numerous open source libraries such as openness, SQLite. It is also supported by the android core libraries. From the point of security, the framework is based on UNIX file system permission that assure application have only those abilities that mobile phone owner gave team at install time.

3.2.4 Dalvik Virtual Machine

It is extremely low memory based virtual machine, whichwas designed especially for Android to run on embedded systems and work well in low power situations. It is also tuned to CPU attributes .The Dalvik VM creates a special file format (.DEX) that is created through build time post processing. Conversion between Java classes and .DEX is done by included “dx” tool.

3.2.5. Integrated Browser

Google made a choice on WebKit as open source web browser.They added a two pass layout and frame flattening. Two pass layout loads a page without waiting for blocking elements such as external CSS or JavaScript.

3.2.6. Optimized Graphics

An Android has 2D graphics library and 3D graphics based on OpenGL ES 1.0.

3.2.7. JSON

JavaScript Object Notation is a syntax for storing and exchanging data. JSON is text, written with JavaScript object notation.

When exchanging data between a browser and a server, the data can only be text. JSON is text, and we can convert any JavaScript object into JSON, and send JSON to the server. We can also convert any JSON received from the server into JavaScript objects, This way we can work with the data as JavaScript object, with no complicated parsing and translation.

3.2.8. Java Virtual Machine

Java virtual machine is a virtual machine that canexecute Java byte code. It is the code execution component of the Java software platform. Sun Microsystems has stated that there are over 5.5 billion JVM-enableddevices.A Java virtual machine is a program which executes certain other programs, namely those containing Java byte code instructions. JVMs are most often implemented to run on an existing operating system, but can also be implemented to run directly on hardware.

3.2.9. Development Environment

It includes a device emulator, tools for debugging,memory and performance profiling, a plug-in for the eclipse IDE. There are a number of hardware dependent features, for instance, a huge media and connections support, GPS, GSM telephony. A great work was done for the developers to start work with Android using device emulator ,tools for debugging and plug-in for eclipse. The different versionof the android operating system that are launched are listed here.

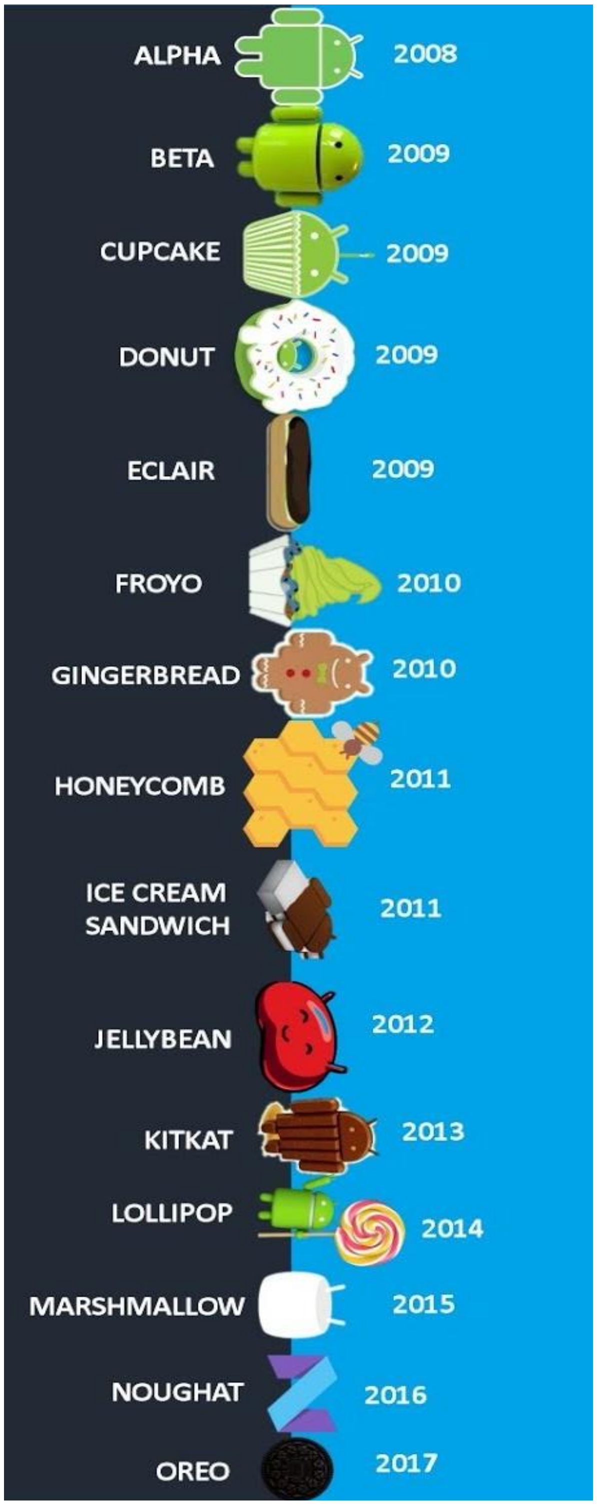


Figure 3.3 Different Versions of Android OS

Table 1.1 Different versions of Android OS

Code name	Version number	Initial release date	API level	Security patches[2]
No codename	1.0	September 23, 2008	1	Unsupported
Petit Four	1.1	February 9, 2009	2	Unsupported
Cupcake	1.5	April 27, 2009	3	Unsupported
Donut	1.6	September 15, 2009	4	Unsupported
Eclair	2.0 – 2.1	October 26, 2009	5 – 7	Unsupported
Froyo	2.2 – 2.2.3	May 20, 2010	8	Unsupported
Gingerbread	2.3 – 2.3.7	December 6, 2010	9 – 10	Unsupported
Honeycomb	3.0 – 3.2.6	February 22, 2011	11 – 13	Unsupported
Ice Cream Sandwich	4.0 – 4.0.4	October 18, 2011	14 – 15	Unsupported
Jellybean	4.1 – 4.3.1	July 9, 2012	16 – 18	Unsupported
Kitkat	4.4 – 4.4.4	October 31, 2013	19 – 20	Unsupported[12]

Code name	Version number	Initial release date	API level	Security patches[2]
Lollipop	5.0 – 5.1.1	November 12, 2014	21 – 22	Unsupported[14]
Marshmallow	6.0 – 6.0.1	October 5, 2015	23	Supported
Nougat	7.0 – 7.1.2	August 22, 2016	24 – 25	Supported
Oreo	8.0 – 8.1	August 21, 2017	26 – 27	Supported
Android P	9			Developer preview; not yet supported

3.3. Architecture

Android is based on the Linux Kernel. Android Developers are able to access all the components of the Application Framework used by core applications when creating an application **.**

The figure given below shows the diagram of Android Architecture. The Android OS can be referred to as a software stack of different layers, where each layer is a group of several program components. Together it includes operating system, middleware and important applications. Each layer in the architecture provides different services to the layer just above it.

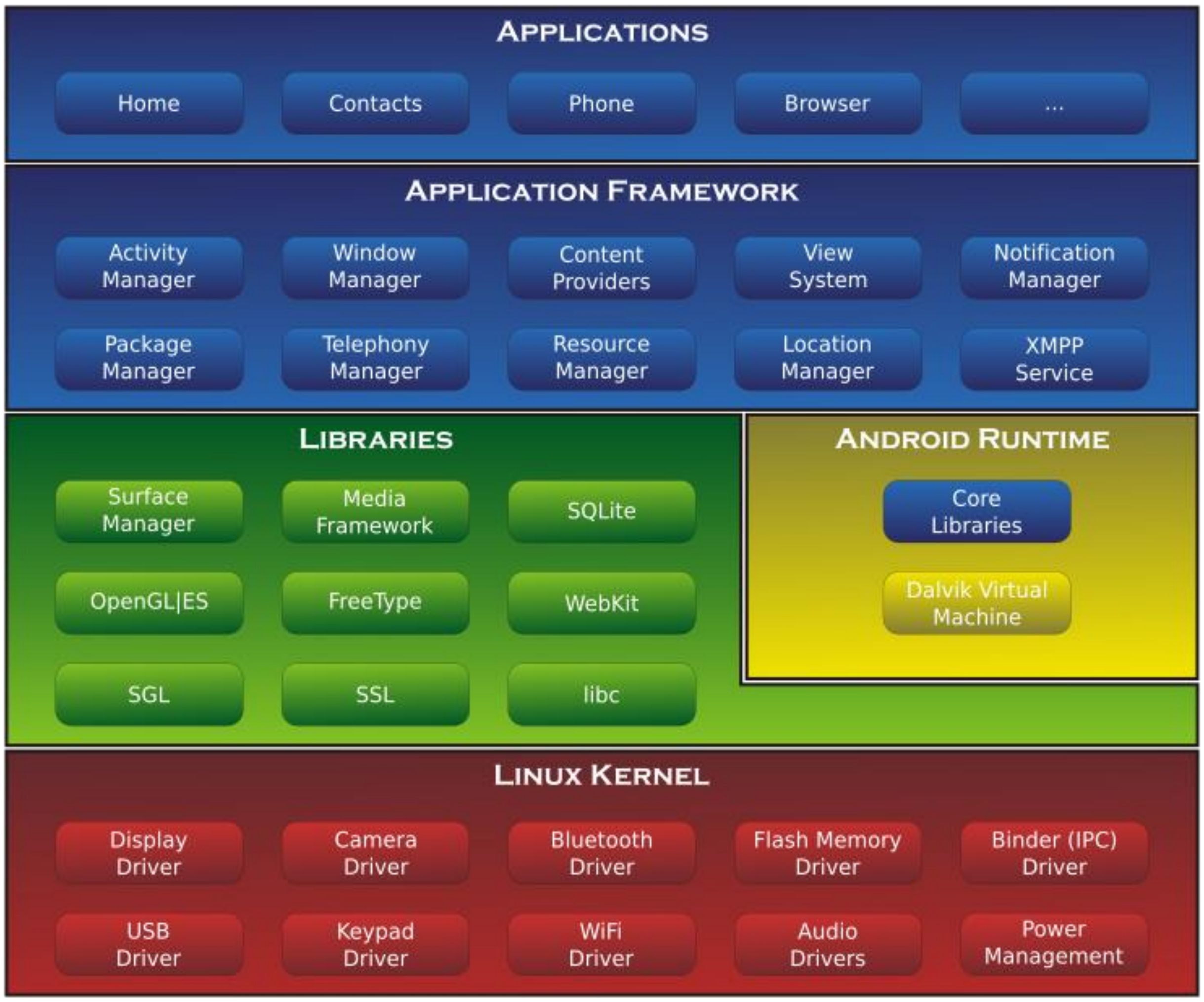


Figure 3.4 Android OS architecture

3.3.1. Android Studio 2.1

i. Android Studio is the official IDE for Android development, and with a single download includes everything you need to begin developing Android apps:

ii. IntelliJ IDEA + Android Studio plug-in

iii. Android SDK Tools

iv. Android Platform-tools

v. A version of the Android platform

vi. Android Emulator with an Android system image including Google Play Services

vii. Fixed minor performance and feature issues.

.

3.3.2. Mobile Drivers

Motorola USB Driver, It allows you to connect smart phone to the computer without any software. It helps the user to connect your smart phone to the computer and transfer data between smart phone and computer.Java is a general purpose programming language that is concurrent, class-based, object-oriented,]and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere" (WORA), meaning that compiled Java code can run on all platforms that support.

3.2.3. Java

Java is a general purpose programming language that is concurrent, class-based, object-oriented, and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere" (WORA), meaning that compiled Java code can run on all platforms that support.

Java without the need for recompilation. Java application are typically compiled to byte code that can run on any Java virtual machine (JVM) regardless of computer architecture. As of 2016, Java is one of the most popular programming languages in use, particularly for client-server web applications, with a reported 9 million developers. Java was originally developed by James Gosling at Sun Microsystems (which has since been acquired by Oracle Corporation) and released in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++, but it has fewer low-level facilities than either of them. One design goal of Java is portability, which means that programs written for the Java platform must run similarly on any combination of hardware and operating system with adequate runtime support. This is achieved by compiling the Java language code to an intermediate representation called Java byte code, instead of directly to architecture-specific machine code. Java byte code instructions are analogous to machine code, but they are intended to be executed by a virtual machine (VM) written specifically for the host hardware. End users commonly use a Java Runtime Environment (JRE) installed on their own machine for standalone Java applications, or in a web browser for Java applets.

3.3.4. JDK

Java development kit, Standard Edition (Java SE) lets you develop and deployJava applications on desktops and servers, as well as in today's demanding embeddedenvironments.

Java offers the rich user interface, performance, versatility, portability, and security that today's applications require. Following libraries of jdk 1.8 are used :

- **Swing**: Java.swing is a GUI widget toolkit for Java. It is part of Oracle's JavaFoundation Classes (JFC) – an API for providing a graphical user interface (GUI) for Java programs.

- **Awt:** Java.eventsub package in Java.awt package provides interfaces and classes fordealing with different types of events fired by components.

- **IO:** Java.io package provides classes for system input and output through data streams,serialization and the file system. This reference will take you through simple and practical methods available in java.io package.

3.3.5. XML

In computing, ExtensibleMarkup Language (XML) is a markup language. The design goals of XML emphasize simplicity, generality, and usability across the Internet. It is a textual data format with strong support via Unicode for different human languages. XML is widely used for the representation of arbitrary data structures such as those used in web services.

Several schema exist to aid in the definition of XML-based languages, while programmers

have developed many application programming interfaces (APIs) to aid the processing of XML data.
Hundreds of document formats using XML have been developed, including RSS, Atom, SVG and XHTML. XML based formats have become the default for many office-productivity tools, including MS-Office, OpenOffice.org and LibreOffice (Open Document), and Apple's iWork. XML has also provided the base language for communication protocols such as XMPP. Applications for the Microsoft .NET Framework use XML files for configuration. Apple has an implementation of a registry based on XML.

XML has come into common use for the interchange of data over the Internet. IETF RFC 7303 gives rules for the construction of Internet Media Types for use when sending XML. It also defines the media types application/xml and text/xml, which say only that the data is in XML, and nothing about its semantics. The use of text/xml has been criticized as a potential source of encoding problems and it has been suggested that it should be deprecated.

CHAPTER 4

DEVELOPMENT PLATFORM

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.

Android is developed in private by Google until the latest changes and updates are ready to be released, at which point the source code is made available publicly. This source code will only run without modification on select devices, usually the Nexus series of devices. The source code is, in turn, adapted by 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-sourced, 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.

4.1. 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 XP or later. As of March 2015, the SDK is not available on Android itself, but the 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. 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.

Android applications are packaged in .apk format and stored under /data/app folder on the Android OS (the folder is accessible only to the root user for security reasons). APK package contains .dex files (compiled byte code files called Dalvik executables), resource files, etc.

Figure 4.1 SDK Manager

4.2. Android Debug Bridge

The Android Debug Bridge (ADB) is a toolkit included in the Android SDK package. It consists of both client and server-side programs that communicate with one another. The ADB is typically accessed through the command-line interface, although numerous graphical user interfaces exist to control ADB.

The format for issuing commands through the ADB is typically:

```
adb [-d|-e|-s <serialNumber>] <command>
```

For example, Android applications can be saved by the command backup to a file, whose name is backup.ab by default.

In a security issue reported in March 2011, ADB was targeted as a vector to attempt to install a rootkit on connected phones using a "resource exhaustion attack".

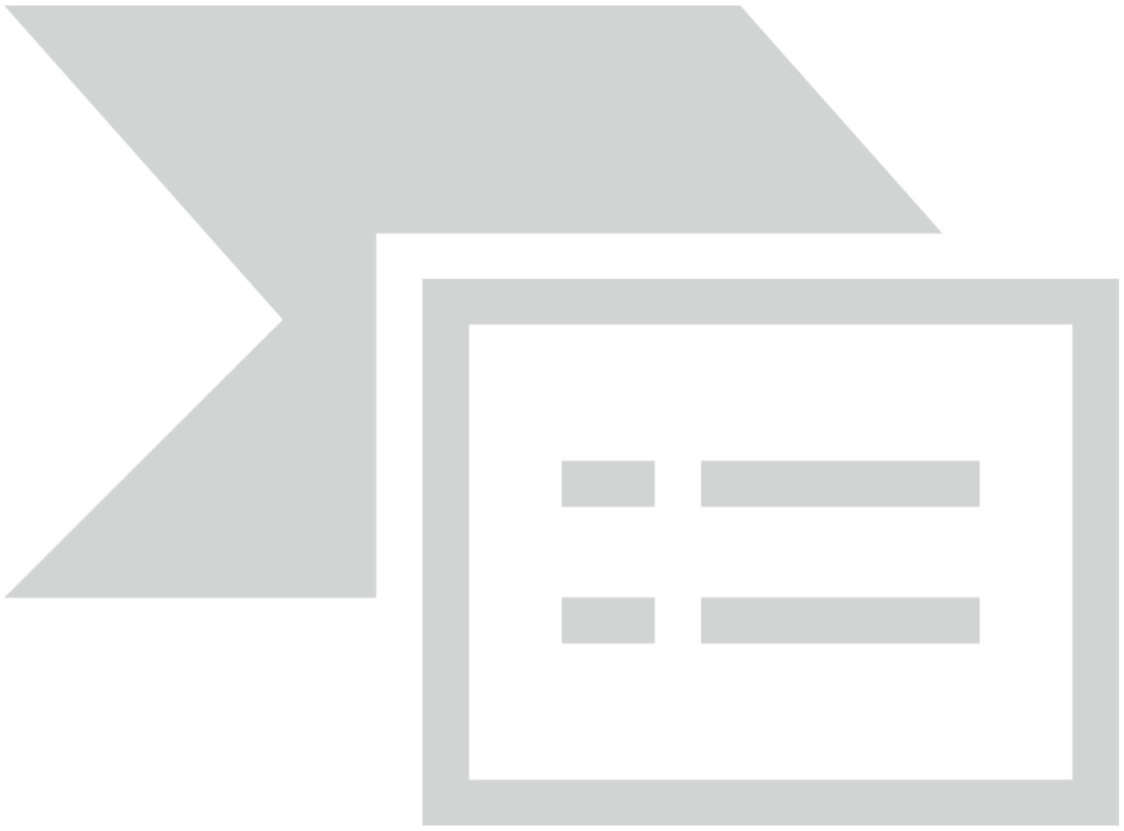


Figure 4.2. Android Debug Bridge

4.3. Android Studio

Android Studio is the official integrated development environment (IDE) for Android platform development.

It was announced on May 16, 2013 at the Google I/O conference. Android Studio is freely available under the Apache License 2.0.

Android Studio was in early access preview stage starting from version 0.1 in May 2013, then entered beta stage starting from version 0.8 which was released in June 2014. The first stable build was released in December 2014, starting from version 1.0.

Based on JetBrains' IntelliJ IDEA software, Android Studio is designed specifically for Android development. It is available for download on Windows, Mac OS X and Linux, and replaced Eclipse Android Development Tools (ADT) as Google's primary IDE for native Android application development.

Features

New features are expected to be rolled out with each release of Android Studio. The following features are provided in the current stable version:

i. Gradle-based build support.

ii. Android-specific refactoring and quick fixes.

iii. Lint tools to catch performance, usability, version compatibility and other problems.

iv. ProGuard integration and app-signing capabilities.

v. Template-based wizards to create common Android designs and components.

vi. A rich layout editor that allows users to drag-and-drop UI components, option to preview layouts on multiple screen configurations.

vii. Support for building Android Wear apps

viii. Built-in support for Google Cloud Platform, enabling integration with Google Cloud Messaging and App Engine.

Version 2.x

	Windows	OS X	Linux
OS version	Microsoft Windows 10/8/7 (32- or 64-bit)	Mac OS X 10.8.5 or higher, up to 10.11.4 (El Capitan)	GNOME or KDE desktop
RAM	2 GB RAM minimum, 8 GB RAM recommended		
Disk space	500 MB disk space for Android Studio, at least 1.5 GB for Android SDK, emulator system images, and caches		
Java version	Java Development Kit (JDK) 8	Java Development Kit (JDK) 6	Java Development Kit (JDK) 8
Screen resolution	1280x800 minimum screen resolution		

Version 1.x

	Windows	OS X	Linux
OS version	Microsoft Windows 10/8.1/8/7/Vista/2003/XP (32 or 64 bit)	Mac OS X 10.8.5 or higher, up to 10.10 to up 10.10.2 up 10.10.3 on 10.10.5 (Yosemite)	GNOME or KDE or Unity desktop on Ubuntu or Fedora or GNU/Linux Debian
RAM	2 GB RAM minimum, 4 GB RAM recommended		
Disk space	500 MB disk space		
Space for Android SDK	At least 1 GB for Android SDK, emulator system images, and caches		

JDK version	Java Development Kit (JDK) 7 or higher
Screen resolution	1280x800 minimum screen resolution

Figure 4.3. Android Studio

4.4. Adobe Photoshop

Adobe Photoshop is a raster graphics editor developed and published by Adobe Systems for Windows and OS X.

Photoshop was created in 1988 by Thomas and John Knoll. Since then, it has become the de facto industry standard in raster graphics editing, such that the word "photoshop" has become a verb as in "to Photoshop an image," "photo shopping" and "photoshop contest", though Adobe discourages such use. It can edit and compose raster images in multiple layers and supports masks, alpha compositing and several color models including RGB, CMYK, Lab color space, spot color and duotone. Photoshop has vast support for graphic file formats but also uses its own PSD and PSB file formats which support all the aforementioned features. In addition to raster graphics, it has limited abilities to edit or render text, vector graphics (especially through clipping path), 3D graphics and video. Photoshop's feature set can be expanded by Photoshop plug-ins, programs developed and distributed independently of Photoshop that can run inside it and offer new or enhanced features.

Photoshop's naming scheme was initially based on version numbers. However, in October 2002, following the introduction of Creative Suite branding, each new version of Photoshop was designated with "CS" plus a number; e.g., the eighth major version of Photoshop was Photoshop CS and the ninth major version was Photoshop CS2. Photoshop CS3 through CS6 were also distributed in two different editions: Standard and Extended. In June 2013, with the introduction of Creative Cloud branding, Photoshop's licensing scheme was changed to that of software as a service and the "CS" suffixes were replaced with "CC". Historically, Photoshop was bundled with additional software such as Adobe ImageReady, Adobe Fireworks, Adobe Bridge, Adobe Device Central and Adobe Camera RAW.

Alongside Photoshop, Adobe also develops and publishes Photoshop Elements, Photoshop Lightroom, Photoshop Express and Photoshop Touch. Collectively, they are branded as "The Adobe Photoshop Family". It is currently a licensed software.

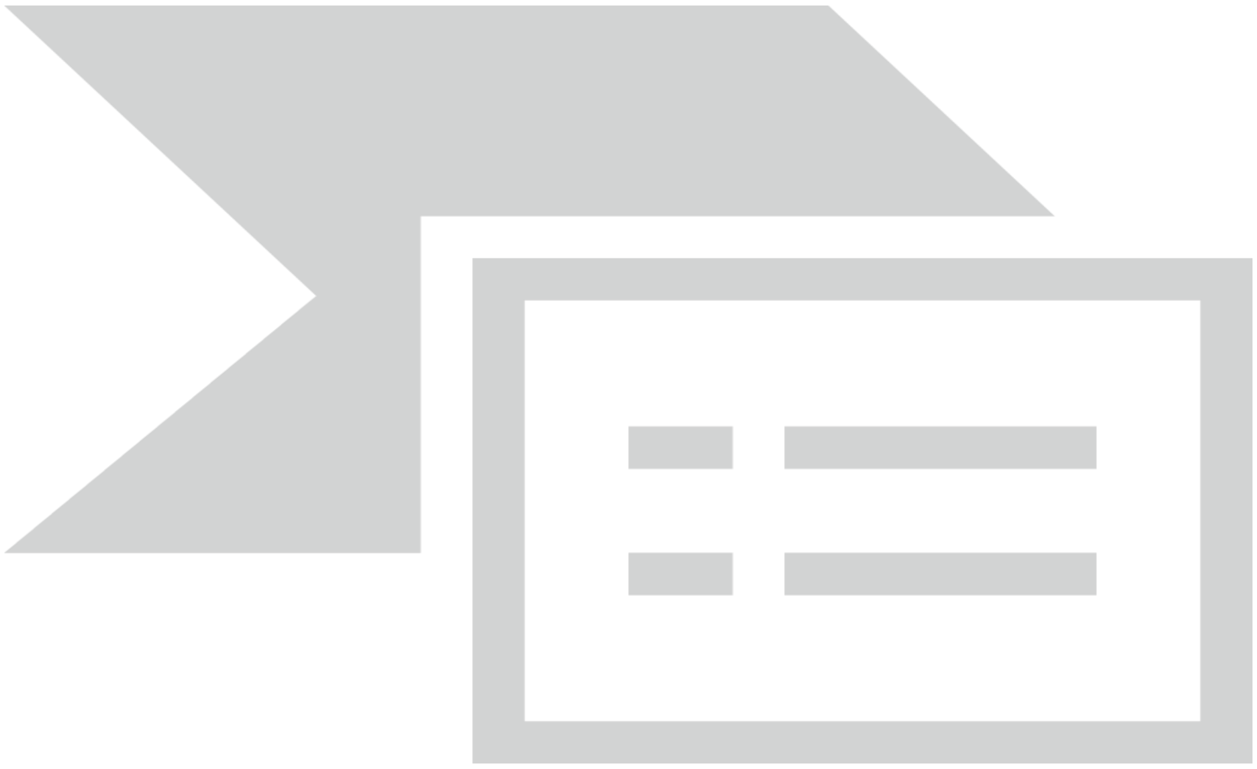


Figure 4.4. Adobe Photoshop

CHAPTER 5

SOFTWARE DESIGN DOCUMENT

5.1. Use Case Diagram

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. These are usually referred to as behavior diagrams used to describe a set of actions (use cases) that some systems (subject) can perform in collaboration with one or more external users of the system (actors).This use case diagram represents two actors, user and administrator. The actors are basically the real world entities who use the proposed system.

The user can view the nearby places. It provide a unique way to find useful places around you - such as restaurants, ATMs, hotels, movie theatre and more.The user can create a new account or, login if the user has already created an account.The user has the facility of choosing an option from the three choices i.e. list, camera view and map.Ituses your phone's camera to give you a completely new way of exploring a place, anywhere in the world. To use the app you simply select one of the pre-defined categories, hold up your Smartphone and watch as the world around you populates with digitally generated signposts

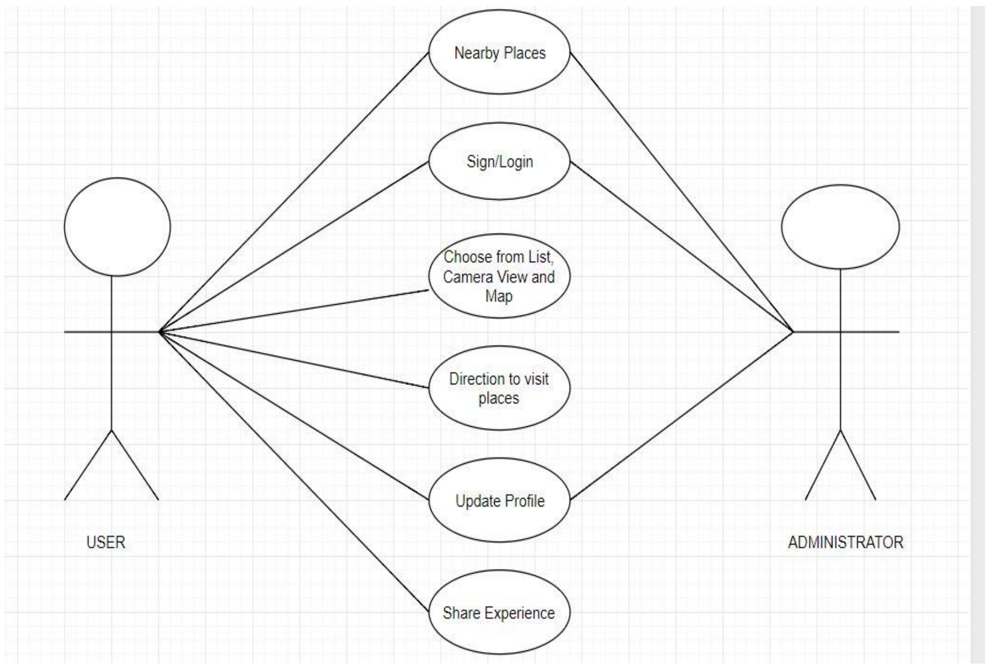


Figure 5.1 Use-Case Diagram

5.2. DFD

A data flow diagram (DFD) is a graphical representation of the flow of data through an information system. DFDs can also be used for the visualization of data processing. On a DFD, data items flow from an external data source or an internal data store to an internal data store or an external data sink, via an internal process.

The context diagram is a level zero data flow diagram. The context diagram for the application shows the various functions the application is capable of executing. The application provides features like get real time distance, find nearest places, choose between camera, list and map view, find places in different categories, check out user review and it has signboards that are interactive.

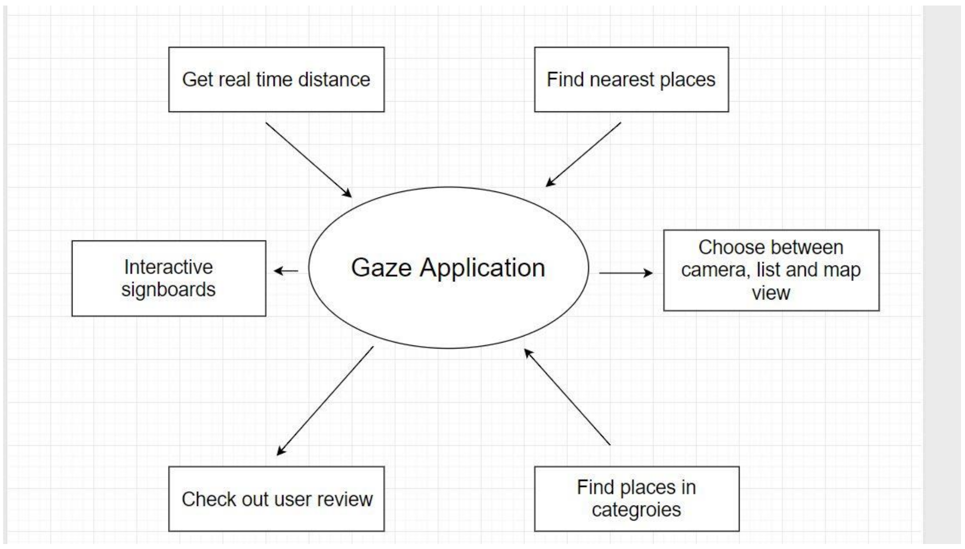


Figure 5.2 Context Diagram

DFD Level 1 provides a more detailed breakout of pieces of the Context Level Diagram. You will highlight the main functions carried out by the system, a you break down the high level process of the Context Diagram into its sub processes.

The above data flow diagram shows the data flow in the appointment activity.

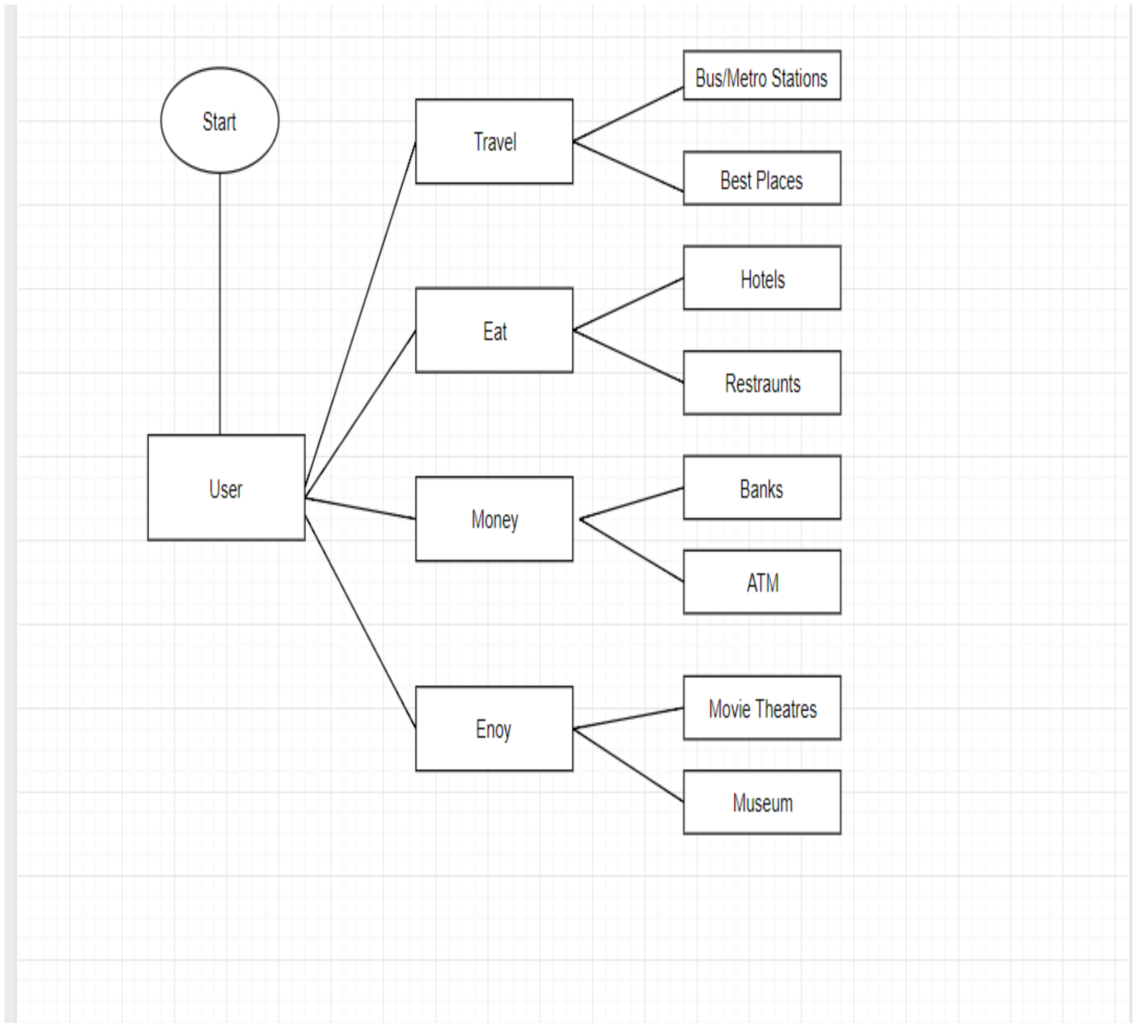


Figure 5.3 Level 1 DFD

5.3 ER DIAGRAM

An **entity–relationship model** (**ER model**) describes inter-related things of interest in a specific domain of knowledge. An ER model is composed of entity types (which classify the things of interest) and specifies relationships that can exist between instances of those entity types.

The ER diagram represents the relationship between entities. Entities are real world objects.

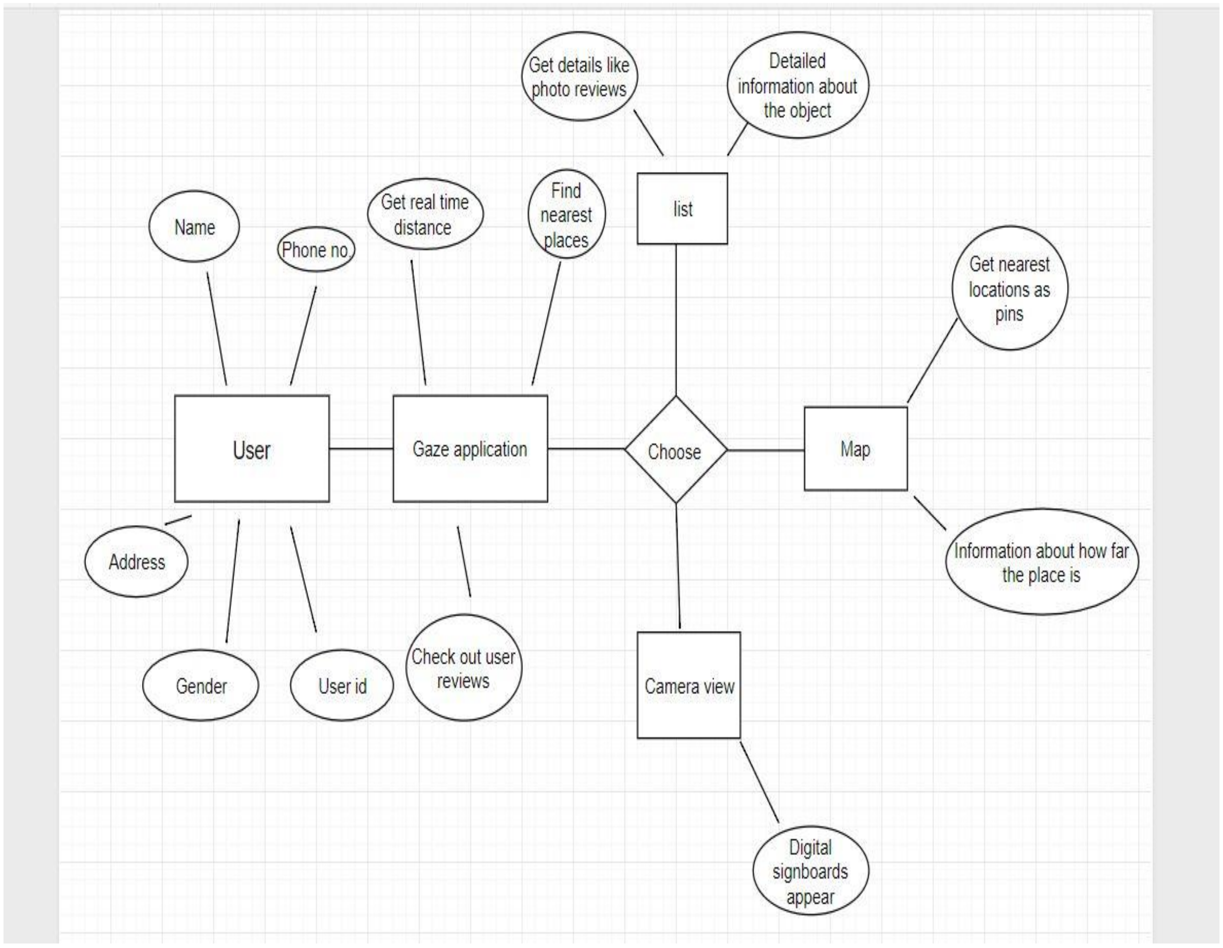

Get details like photo reviews
Detailed information about the object
list
Name
Phone no.
Get real time distance
Find nearest places
Get nearest locations as pins
User
Gaze application
Choose
Map
Address
Information about how far the place is
Gender
User id
Check out user reviews
Camera view
Digital signboards appear

Figure 5.4 ER Diagram

5.4 Flow Chart

A flowchart is a type of diagram thatrepresents an algorithm, workflow orprocess, showingthe steps as boxes of various kinds, and their order by connecting them with arrows. This diagrammatic representation illustrates a solution model to a given problem. Flowcharts are used in analyzing, designing, documenting or managing a process or program in various fields.

The flowchart shows a sequence in which various activities occur. Flowchart for the application shows the sequence in which the user of the application will interact with the application.

First clicking on the application icon the user will start the application. If the user is a new user, then the user has to register herself, if the user already has an account he/she can simply log in.

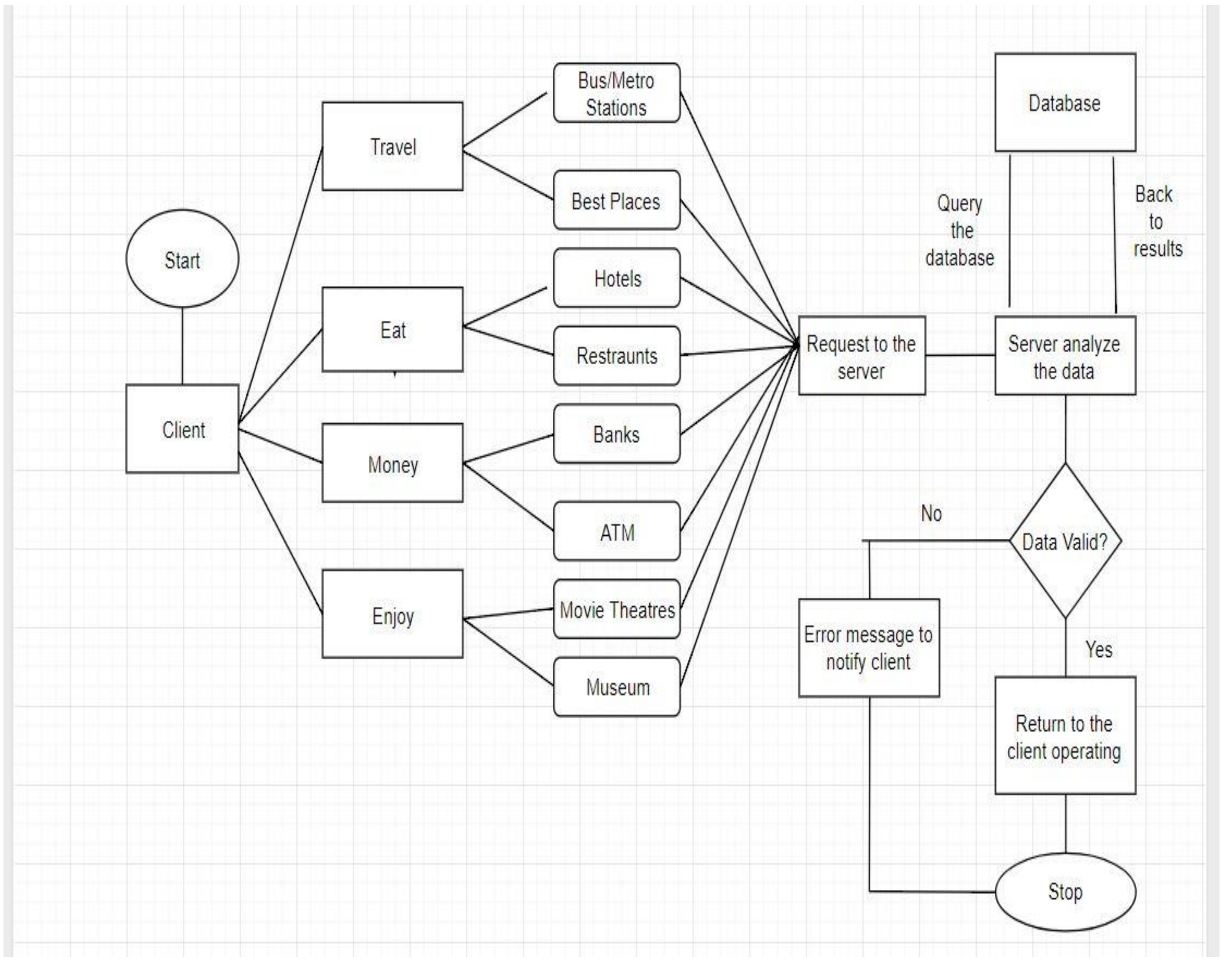


Figure 5.5. Flow Chart

CHAPTER 6

IMPLEMENTATION

6.1 Project definition

Gaze is an augmented reality exploration app intended for Android Operating System. brings Google Maps to life, so all you need to do is hold up your phone and the streets in front of you will be transformed with information about nearby facilities. This application is a unique way to find useful places around you - such as restaurants, ATMs, shops, bus/metro stations and more. Gaze uses your phone’s camera to give you a completely new way of exploring a place, anywhere in the world. To use the app you simply select one of the pre-defined categories, hold up your smartphone and watch as the world around you populates with digitally generated signposts. If you want more information, you simply click on the signpost to get full contact details, user reviews and directions.

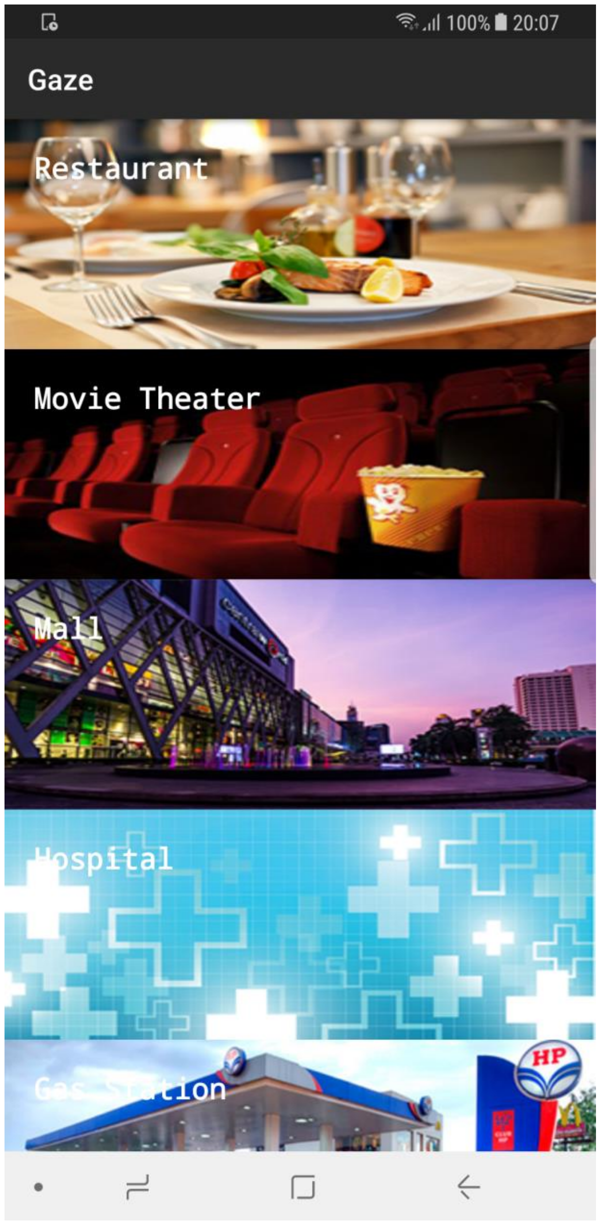


Figure 6.1. List Activity in Gaze Application

6.2. Fragment Of Gaze Application

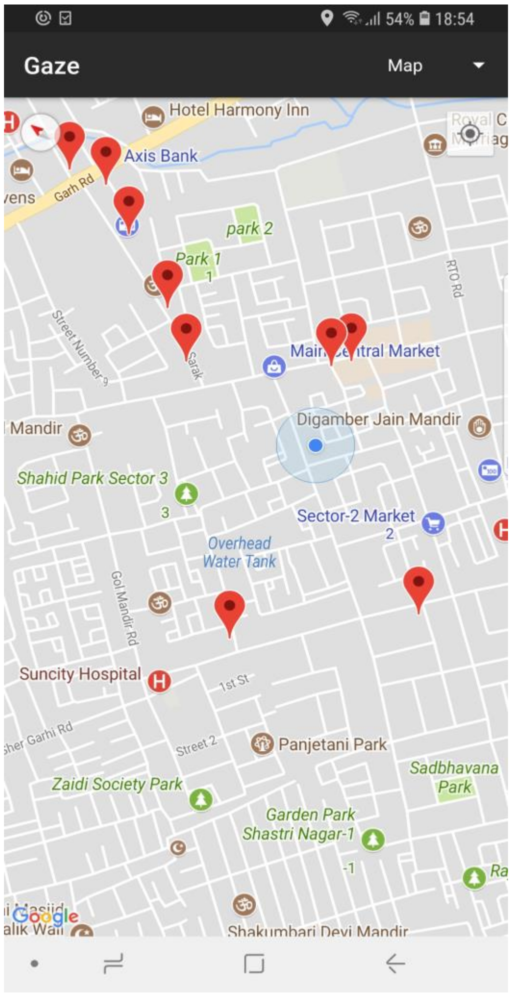
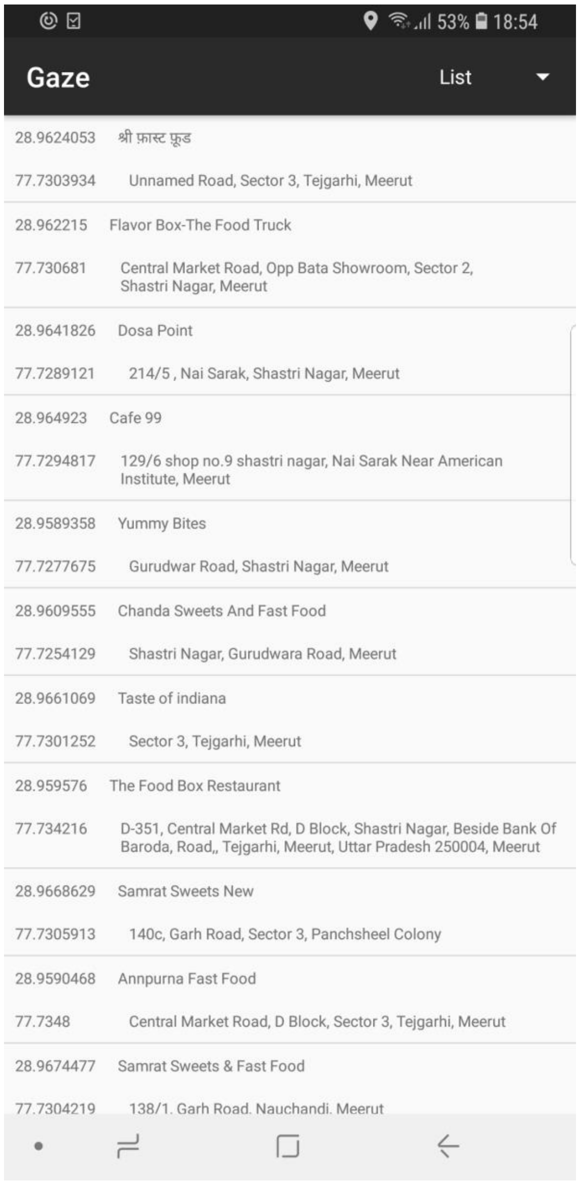


Figure 6.2. Map View Of Restaurant Section



53% 18:54

Gaze List

28.9624053	श्री फ़ास्ट फ़ूड
77.7303934	Unnamed Road, Sector 3, Tejgarhi, Meerut
28.962215	Flavor Box-The Food Truck
77.730681	Central Market Road, Opp Bata Showroom, Sector 2, Shastri Nagar, Meerut
28.9641826	Dosa Point
77.7289121	214/5 , Nai Sarak, Shastri Nagar, Meerut
28.964923	Cafe 99
77.7294817	129/6 shop no.9 shastri nagar, Nai Sarak Near American Institute, Meerut
28.9589358	Yummy Bites
77.7277675	Gurudwar Road, Shastri Nagar, Meerut
28.9609555	Chanda Sweets And Fast Food
77.7254129	Shastri Nagar, Gurudwara Road, Meerut
28.9661069	Taste of indiana
77.7301252	Sector 3, Tejgarhi, Meerut
28.959576	The Food Box Restaurant
77.734216	D-351, Central Market Rd, D Block, Shastri Nagar, Beside Bank Of Baroda, Road,, Tejgarhi, Meerut, Uttar Pradesh 250004, Meerut
28.9668629	Samrat Sweets New
77.7305913	140c, Garh Road, Sector 3, Panchsheel Colony
28.9590468	Annpurna Fast Food
77.7348	Central Market Road, D Block, Sector 3, Tejgarhi, Meerut
28.9674477	Samrat Sweets & Fast Food
77.7304219	138/1, Garh Road, Nauchandi, Meerut

Figure 6.3. List Activity

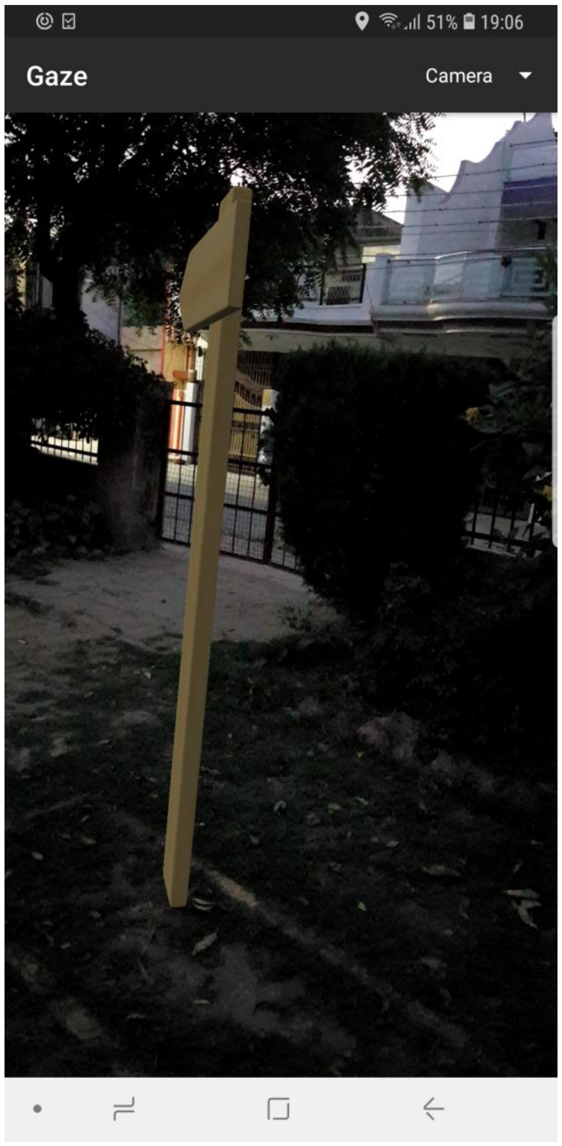


Figure 6.4. Camera View of Restaurant Section

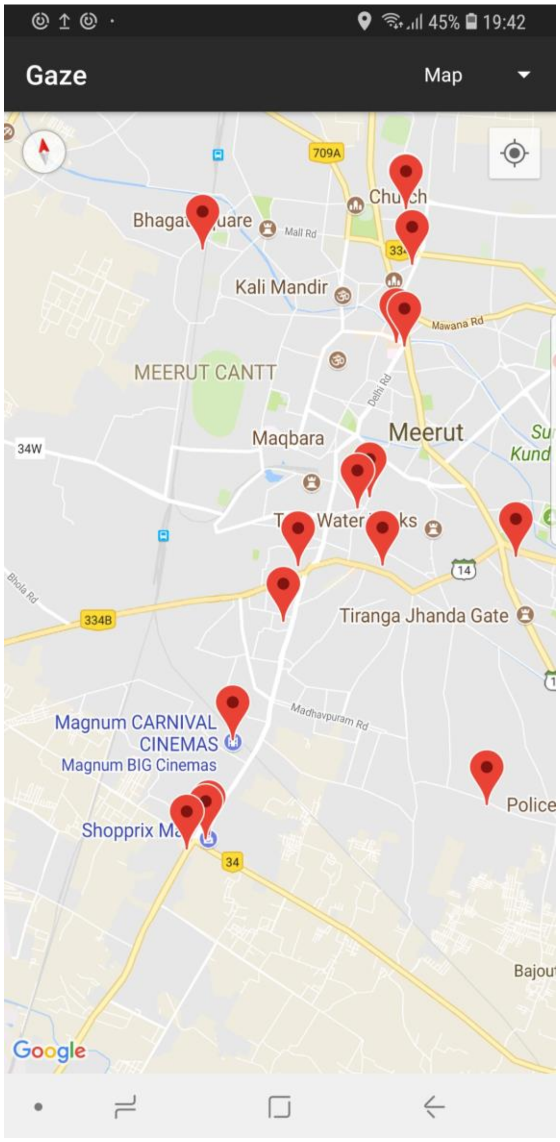
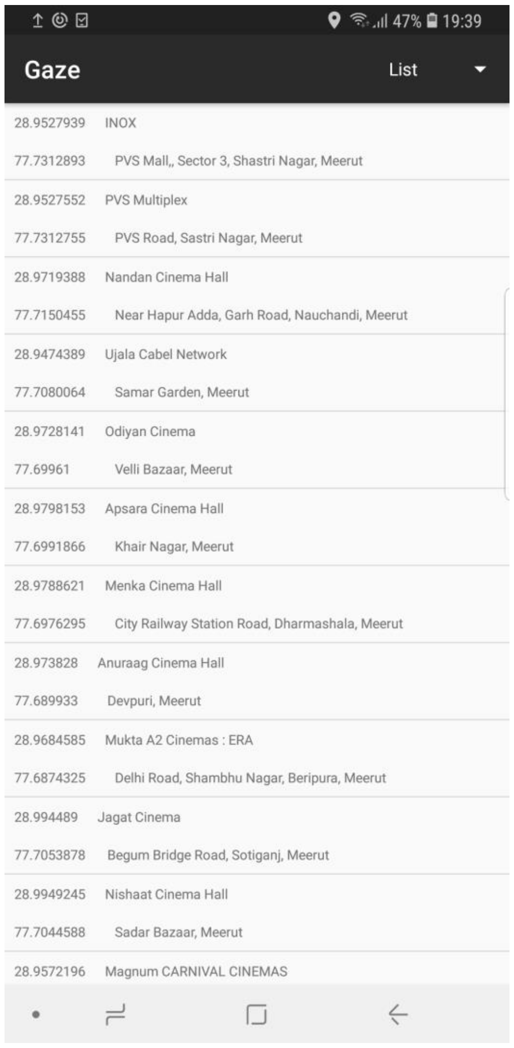


Figure 6.5. Map Fragment



28.9527939	INOX
77.7312893	PVS Mall,, Sector 3, Shastri Nagar, Meerut
28.9527552	PVS Multiplex
77.7312755	PVS Road, Sastri Nagar, Meerut
28.9719388	Nandan Cinema Hall
77.7150455	Near Hapur Adda, Garh Road, Nauchandi, Meerut
28.9474389	Ujala Cabel Network
77.7080064	Samar Garden, Meerut
28.9728141	Odiyan Cinema
77.69961	Velli Bazaar, Meerut
28.9798153	Apsara Cinema Hall
77.6991866	Khair Nagar, Meerut
28.9788621	Menka Cinema Hall
77.6976295	City Railway Station Road, Dharmashala, Meerut
28.973828	Anuraag Cinema Hall
77.689933	Devpuri, Meerut
28.9684585	Mukta A2 Cinemas : ERA
77.6874325	Delhi Road, Shambhu Nagar, Beripura, Meerut
28.994489	Jagat Cinema
77.7053878	Begum Bridge Road, Sotiganj, Meerut
28.9949245	Nishaat Cinema Hall
77.7044588	Sadar Bazaar, Meerut
28.9572196	Magnum CARNIVAL CINEMAS

Figure 6.6. List Fragment

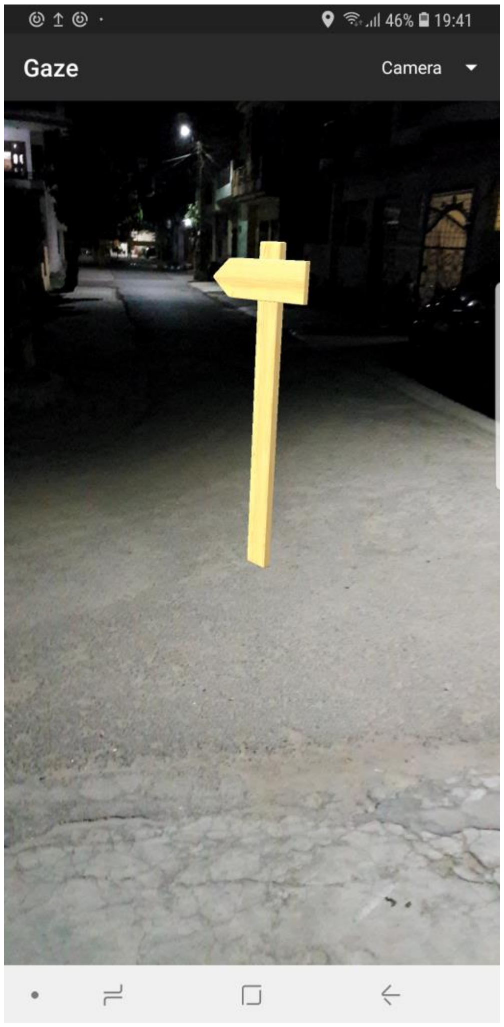


Figure 6.7. Camera Fragment

CHAPTER 7

TESTING

7.1. Test Plan

A test plan implies a series of desired course of action to be followed in accomplishing various testing methods. The Test Plan acts as a blue print for the action that is to be followed. The software engineers create a computer program, its documentation and related data structures. The software developers is always responsible for testing the individual units of the programs, ensuring that each performs the function for which it was designed. There is an independent test group (ITG) which is to remove the inherent problems associated with letting the builder to test the thing that has been built. The specific objectives of testing should be stated in measurable terms. So that the mean time to failure, the cost to find and fix the defects, remaining defect density or frequency of occurrence and test work-hours per regression test all should be stated within the test plan. The levels of testing include:

- Unit Testing
- Integration Testing
- Data Validation Testing
- Output Testing

7.1.1 Unit Testing

Unit testing focuses verification effort on the smallest unit of softwaredesign – the software component or module. Using the component level design description as a guide, important control paths are tested to uncover errors within the boundary of the module. The relative complexity of tests and uncovered scope established for unit testing. Unit testing for several modules in this project:

- **Algorithms**-Our algorithm was built in a waterfall model fashion, it’s functionality isincremented in a step by step fashion. If the previous added functionality is working fine, then another layer of functionality is added. After the complete build is established, it is tested separately through the command window, to test its complete functionality independent of other modules. As the unit testing progressed, the accuracy of the algorithm is increased, by identifying the areas which can be manipulated in order to do so. The whole unit testing process resulted in a better algorithmic accuracy then the parent module.

- **GUI-** The small GUI models of various tasks are separately built for testing purposes.Various windows and tools in the GUI are tested independent of each other in accordance with the rules of the unit testing model. Afterwards the complete GUI model is compiled as a wholesystem, connected vaguely to each other through empty functions, just to imitate the calling of various windows of GUI intertwined among each other. This process of
- unit testing resulted in the perfectly compiled GUI model of the project.

- **Numerous processing programs**- The project consists of various intermediary processeswhich help to establish interconnectivity and perform other important functions such as creating files and folders and managing directories. One of the most important process is that of image processing, that helps to convert various images of different dimensions to a standard size. Thus, these processes are tested separately to ensure proper independent working.

7.1.2. Integration Testing

Integration testing is systematic technique for constructing theprogram structure while at the same time conducting tests to uncover errors associated with interfacing. The objective is to take unit tested components and build a program structure that has been dictated by design. The entire program is tested as whole. Correction is difficult

because isolation of causes is complicated by vast expanse of entire program. Once these errors are corrected, new ones appear and the process continues in a seemingly endless loop.

The different modules are integrated together piece by piece to ensure they are working as they are intended to. Various mistakes had been found in the structure of the modules while doing so, and which in turn have been rectified to ensure that the system is successfully integrated and proper event calling and function returning was taking place.

7.1.3. Validation Testing or System Testing

This is the final step in testing. In this theentire system was tested as a whole with all forms, code, modules and class modules. This form of testing is popularly known as Black Box testing or System tests. Black Box testing method focuses on the functional requirements of the software. That is, Black Box testing enables the software engineer to derive sets of input conditions that will fully exercise all functional requirements for a program.

Black Box testing attempts to find errors in the following categories; incorrect or missing functions, interface errors, errors in data structures or external data access, performance errors and initialization errors and termination errors. The whole system is tested as a whole and single entity. Most of the system components are working properly, but gui had not been in synchronization properly. This problem is solved in this section, and the whole software has been successfully tested.

7.1.4. Acceptance Testing

The system considered is tested for user acceptance; here it should satisfy the firm's need. The software should keep in touch with perspective system; user at the time of developing and making changes whenever required. This done with respect to the following points:

- Input Screen Designs
- Output Screen Designs
- Online message to guide the user and the like.

The above testing is done taking various kinds of test data. Preparation of test data plays a vital role in the system testing. After preparing the test data, the system under study is tested using that test data. While testing the system by which test data errors are again uncovered and corrected by using above testing steps and corrections are also noted for future use.

7.1.5. Static and Dynamic Testing

There are many approaches to software testing.Reviews, walkthroughs, or inspections are considered as static testing, whereas actually executing programmed code with a given set of test cases is referred to as dynamic testing. Static testing can be (and unfortunately in practice often is) omitted. Dynamic testing takes place when the program itself is used for the first time (which is generally considered the beginning of the testing stage). Dynamic testing may begin

before the program is 100% complete in order to test particular sections of code (modules or discrete functions). Typical techniques for this are either using stubs/drivers or execution from a debugger environment. For example, spreadsheet programs are, by their very nature, tested to a large extent interactively, with results displayed immediately after each calculation or text manipulation.

7.1.6. Output Testing

The system considered is tested for user acceptance; here it shouldsatisfy the firm’s need. The software should keep in touch with perspective system; user at the time of developing and making changes whenever required.

The above testing is done taking various kinds of test data. Preparation of test data plays a vital role in the system testing. After preparing the test data, the system under study is tested using that test data. While testing the system by which test data errors are again uncovered and corrected by using above testing steps and corrections are also noted for future use. In this section the software has been put to test in such way that ensures that all the logical and representational entities of our software are in sync with each other and are generating properand expected output. The system is given some input values and datasets whose correct outputs is known to well in advance. Then the output generated by the system is cross-checked at every step, thus giving way to proper output checking and validation.

CHAPTER 8

CONCLUSION

This application GAZE is about ensuring that the user gets the real-time distance to a point of interest.

It is a helping hand, a guide, a friend that is always ready to help the userand to ensure that the user can find all kind of places around him. It provides all basic facilities to the user like find nearest places, choose between camera, list and map view, find places in different categories, check out user review and it has signboards that are interactive.
This application has a very friendly user interface and can be operated easily by anyone. This application is build with the intention of letting the user to find places in a way most suitable to him.

Using the application the user can navigate intuitively to places of his interest, using his phone's camera.All he has to do is to simply select one of the pre-defined categories, hold up his smart phone and watch as the world around him populates with digitally generated signposts. If he want more information, he simply click on the signpost to get full contact details, user reviews and directions.

The augmented reality feature combined with geo-tagging makes it easy to know which direction you should be going and the app makes it easy to quickly access information about the facilities around you.

Find places like

- Restaurants | Bars | Cafes
- Banks | ATMs
- Movie Theaters
- Museums | Art Galleries
- Parks
- Gas Stations
- Metro Stations | Train Stations | Bus Stops | Taxi Stands | Airports
- Hospitals | Doctors | Dental Clinics | Pharmacies
- Gyms | Spas
- Hotels
- Shopping Malls | Grocers | Clothing Stores | Book Stores | Shoe Stores

CHAPTER 9

FUTURE SCOPE AND LIMITATIONS

9.1. Future scope

- Gaze can be used by hotels and guesthouses to ease the stay of customers.
- When guest is close to location, a notification with directions on map, options to see arrival, parking, self-check-in info could be sent. If guest has no internet a SMS could help also. If guest doesn't give GPS permission it could be sent close to check-in time.
- Rather than just places, objects can be pinned to the AR map. Guests can find the objects in the rooms when needed just by using the AR feature of Gaze.
- We can also attach videos/images in augmented reality, instead of writing letters on paper to explain where to find/how to use things

9.2. Limitations

- The application GAZE has tight hardware and software requirement constraints below which it cannot be developed.
- With time when data will grow, it would become difficult to manage it and also the response time will increase that is the application will become slower.

CHAPTER 10

APPENDICES

10.1. APIs

The APIs used for the project are mentioned. Each file as mentioned below has a separate function that it performs for the application. The GetNearbyPlacesData file uses the Google Places API to access data about the nearby places mentioned by categories. This data is accessed in the form of a json file that can be downloaded from a URL formed using the category type. The DownloadUrl file accesses the URL obtained from GetNearbyPlacesData. It returns passes the json file obtained to the DataParser. The DataParser converts the json entries into usable data fields. This data is finally passed to the NearbyPlaces class that stores the data as a Hash map.

10.1.1. DataParser.java

```
package com.kbaquri.gaze.Api;
import android.util.Log;
import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;


public class DataParser {
   public List<HashMap<String, String>> parse(String jsonData) {
JSONArrayjsonArray = null;
JSONObjectjsonObject;

        try {
Log.d("Places", "parse");
jsonObject = new JSONObject((String) jsonData);
```

```
jsonArray = jsonObject.getJSONArray("results");
        } catch (JSONException e) {
Log.d("Places", "parse error");
e.printStackTrace();
        }
        return getPlaces(jsonArray);
    }

    private List<HashMap<String, String>>getPlaces(JSONArrayjsonArray) {
        int placesCount = jsonArray.length();
        List<HashMap<String, String>>placesList = new ArrayList<>();
        HashMap<String, String>placeMap = null;
Log.d("Places", "getPlaces");

        for (int i = 0; i<placesCount; i++) {
            try {
placeMap = getPlace((JSONObject) jsonArray.get(i));
placesList.add(placeMap);
Log.d("Places", "Adding places");

            } catch (JSONException e) {
Log.d("Places", "Error in Adding places");
e.printStackTrace();
            }
        }
        return placesList;
    }

    private HashMap<String, String>getPlace(JSONObjectgooglePlaceJson) {
        HashMap<String, String>googlePlaceMap = new HashMap<String, String>();
        String placeName = "-NA-";
        String vicinity = "-NA-";
        String latitude = "";
        String longitude = "";
        String reference = "";

Log.d("getPlace", "Entered");

        try {
            if (!googlePlaceJson.isNull("name")) {
placeName = googlePlaceJson.getString("name");
```

```
            }
            if (!googlePlaceJson.isNull("vicinity")) {
                vicinity = googlePlaceJson.getString("vicinity");
            }
            latitude =
googlePlaceJson.getJSONObject("geometry").getJSONObject("location").getString("lat");
            longitude =
googlePlaceJson.getJSONObject("geometry").getJSONObject("location").getString("lng")
;
            reference = googlePlaceJson.getString("reference");
googlePlaceMap.put("place_name", placeName);
googlePlaceMap.put("vicinity", vicinity);
googlePlaceMap.put("lat", latitude);
googlePlaceMap.put("lng", longitude);
googlePlaceMap.put("reference", reference);
Log.d("getPlace", "Putting Places");
        } catch (JSONException e) {
Log.d("getPlace", "Error");
e.printStackTrace();
        }
        return googlePlaceMap;
    }
}
```

10.1.2. DownloadUrl.java

```
package com.kbaquri.gaze.Api;
import android.util.Log;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.URL;

public class DownloadUrl {
   public String readUrl(String strUrl) throws IOException {
       String data = "";
InputStreamiStream = null;
HttpURLConnectionurlConnection = null;
       try {
           URL url = new URL(strUrl);

           // Creating an http connection to communicate with url
urlConnection = (HttpURLConnection) url.openConnection();

           // Connecting to url
urlConnection.connect();

           // Reading data from url
iStream = urlConnection.getInputStream();

BufferedReaderbr = new BufferedReader(new InputStreamReader(iStream));

StringBuffersb = new StringBuffer();

           String line = "";
           while ((line = br.readLine()) != null) {
sb.append(line);
           }

           data = sb.toString();
Log.d("downloadUrl", data.toString());
br.close();
```

```
        } catch (Exception e) {
Log.d("Exception", e.toString());
        } finally {
iStream.close();
urlConnection.disconnect();
        }
        return data;
    }
}
```

10.1.3. GetNearbyPlacesData.java

```
package com.kbaquri.gaze.Api;
import android.os.AsyncTask;
import android.util.Log;
import org.greenrobot.eventbus.EventBus;
import java.util.HashMap;
import java.util.List;
public class GetNearbyPlacesData extends AsyncTask<Object, String, String> {
   private String googlePlacesData;

   @Override
   protected String doInBackground(Object... params) {
      try {
Log.d("GetNearbyPlacesData", "doInBackground entered");
         String url = (String) params[0];
DownloadUrldownloadUrl = new DownloadUrl();
googlePlacesData = downloadUrl.readUrl(url);
Log.d("GooglePlacesReadTask", "doInBackground Exit");
      } catch (Exception e) {
Log.d("GooglePlacesReadTask", e.toString());
      }
      return googlePlacesData;
   }

   @Override
   protected void onPostExecute(String result) {
Log.d("GooglePlacesReadTask", "onPostExecute Entered");
      List<HashMap<String, String>>nearbyPlacesList = null;
DataParserdataParser = new DataParser();
nearbyPlacesList =  dataParser.parse(result);

EventBus.getDefault().postSticky(new NearbyPlaces(nearbyPlacesList));

Log.d("GooglePlacesReadTask", "onPostExecute Exit");
   }
}
```

10.1.4. NearbyPlaces.java

```
package com.kbaquri.gaze.Api;
import java.util.HashMap;
import java.util.List;

public class NearbyPlaces {

    List<HashMap<String, String>>nearbyPlacesList;

    public NearbyPlaces(List<HashMap<String, String>>nearbyPlacesList){
this.nearbyPlacesList = nearbyPlacesList;
    }

    public List<HashMap<String, String>>getNearbyPlacesList(){
        return nearbyPlacesList;
    }
}
```

10.2. Java files for first activity

These activities include java classes and their functions that define how the list activity functions. This group of files also includes the data to be passed to the main activity using intent. Recycler view is applied to the list activity to optimize it as well as make it faster to execute.

10.2.1. ListActivity.java

```
package com.kbaquri.gaze.First;
import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.RecyclerView;
import android.view.View;
import com.kbaquri.gaze.R;
import com.kbaquri.gaze.Second.MainActivity;

public class ListActivity extends AppCompatActivity {

    Category[] categories;

    private RecyclerViewmRecyclerView;
    private RecyclerView.AdaptermAdapter;
    private RecyclerView.LayoutManagermLayoutManager;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_list);

        categories = CategoryList.CATEGORIES;

mRecyclerView = findViewById(R.id.recycler_view);
mRecyclerView.setHasFixedSize(true);

mLayoutManager = new LinearLayoutManager(this);
mRecyclerView.setLayoutManager(mLayoutManager);
mAdapter = new MyRecyclerAdapter(categories);
mRecyclerView.setAdapter(mAdapter);

mRecyclerView.addOnItemTouchListener(new
RecyclerTouchListener(getApplicationContext(), mRecyclerView, new
RecyclerTouchListener.ClickListener() {
            @Override
            public void onClick(View view, int position) {
showMap(categories[position].getName());
            }

            @Override
            public void onLongClick(View view, int position) {

            }
        }));

    }

    void showMap(String nearby) {
        Intent intent = new Intent(this, MainActivity.class);
intent.putExtra("nearby", nearby);
startActivity(intent);
    }
}
```

10.2.2. MyRecyclerAdapter.java

```
package com.kbaquri.gaze.First;
import android.support.v7.widget.RecyclerView;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageView;
import android.widget.TextView;
import com.kbaquri.gaze.R;


public class MyRecyclerAdapter extends
RecyclerView.Adapter<MyRecyclerAdapter.ViewHolder> {

    private Category[] mCategories;

    public static class ViewHolder extends RecyclerView.ViewHolder {
        public ImageViewbackgroundIV;
        public TextViewtitleTV;

        public ViewHolder(View itemView) {
            super(itemView);

backgroundIV = itemView.findViewById(R.id.item_background);
titleTV = itemView.findViewById(R.id.item_title);
        }
    }

    public MyRecyclerAdapter(Category[] categories) {
this.mCategories = categories;
    }

    @Override
    public ViewHolderonCreateViewHolder(ViewGroup parent, int viewType) {

        View itemView =LayoutInflater.from(parent.getContext())
                .inflate(R.layout.activity_list_item, parent, false);

        return new ViewHolder(itemView);
    }

    @Override
    public void onBindViewHolder(ViewHolder holder, int position) {
        holder.backgroundIV.setImageResource(mCategories[position].getBackground());
holder.titleTV.setText(mCategories[position].getTitle());
    }

    @Override
    public int getItemCount() {
        return mCategories.length;
    }

}
```

10.2.3 RecyclerTouchListener.java

```
package com.kbaquri.gaze.First;
import android.content.Context;
import android.support.v7.widget.RecyclerView;
import android.view.GestureDetector;
import android.view.MotionEvent;
import android.view.View;

public class RecyclerTouchListener implements RecyclerView.OnItemTouchListener {
    private GestureDetectorgestureDetector;
    private ClickListenerclickListener;

    public RecyclerTouchListener(Context context, final RecyclerViewrecyclerView, final
ClickListenerclickListener) {
this.clickListener = clickListener;
gestureDetector = new GestureDetector(context, new
GestureDetector.SimpleOnGestureListener() {
            @Override
            public booleanonSingleTapUp(MotionEvent e) {
                return true;
            }

            @Override
            public void onLongPress(MotionEvent e) {
                View child = recyclerView.findChildViewUnder(e.getX(), e.getY());
                if (child != null &&clickListener != null) {
clickListener.onLongClick(child, recyclerView.getChildLayoutPosition(child));
                }
            }
        });
    }

    @Override
    public booleanonInterceptTouchEvent(RecyclerViewrv, MotionEvent e) {

        View child = rv.findChildViewUnder(e.getX(), e.getY());
        if (child != null &&clickListener != null &&gestureDetector.onTouchEvent(e)) {
clickListener.onClick(child, rv.getChildLayoutPosition(child));
        }
        return false; }


 @Override
    public void onTouchEvent(RecyclerViewrv, MotionEvent e) {
    }

    @Override
    public void onRequestDisallowInterceptTouchEvent(booleandisallowIntercept) {

    }

    public interface ClickListener {
        void onClick(View view, int position);

        void onLongClick(View view, int position);
    }
}
```

10.2.4. Category.java

```
package com.kbaquri.gaze.First;
import android.graphics.drawable.Drawable;
import java.util.List;

public class Category {

    private String mTitle;
    private String mName;
    private int mBackground;

    public Category(String title, String name, int background) {
this.mTitle = title;
this.mName = name;
this.mBackground = background;
    }

    public String getTitle() {
        return mTitle;
    }

    public String getName() {
        return mName;
    }

    public int getBackground() {
        return mBackground;
    }
}
```

10.2.5. CategoryList.java

```
package com.kbaquri.gaze.First;
import com.kbaquri.gaze.R;

public final class CategoryList {

   private CategoryList(){

   }

   public static final Category[] CATEGORIES = {

         new Category("Restaurant", "restaurant", R.drawable.resta),
         new Category("Movie Theater", "movie_theater", R.drawable.movc),
         new Category("Mall", "shopping_mall", R.drawable.shopc),
         new Category("Hospital", "hospital", R.drawable.hospa),
         new Category("Gas Station", "gas_station", R.drawable.petb),
         new Category("Mosque", "mosque", R.drawable.mosa),
         new Category("ATM", "atm", R.drawable.atmc)
   };

}
```

10.3. Java files for second activity

The second activity includes 3 fragments. These are map fragment, list fragment and camera fragment. It also includes a dropdown menu that can switch between the fragments. The fragments can also be switched using a slider. The MainActivity class is the main class for the project in the sense that it collects all the data and passes it to the separate fragments.

10.3.1. MainActivity.java

```
package com.kbaquri.gaze.Second;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.location.Location;
import android.location.LocationManager;
import android.support.annotation.NonNull;
import android.support.v4.app.ActivityCompat;
import android.support.v4.content.ContextCompat;
import android.support.v4.view.ViewPager;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.Spinner;
import com.google.android.gms.location.FusedLocationProviderClient;
import com.google.android.gms.location.LocationServices;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.kbaquri.gaze.Api.GetNearbyPlacesData;
import com.kbaquri.gaze.Api.NearbyPlaces;
import com.kbaquri.gaze.R;
import org.greenrobot.eventbus.EventBus;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;

public class MainActivity extends AppCompatActivity {

    private static final int PERMISSIONS_REQUEST_ACCESS_FINE_LOCATION = 1;

    String nearby = "restaurant";

    private ViewPagerAdaptermAdapter;
    private ViewPagermPager;

    private booleanmLocationPermissionGranted = false;

    // The entry point to the Fused Location Provider.
    private FusedLocationProviderClientmFusedLocationProviderClient;

    // The geographical location where the device is currently located. That is, the last-
known
    // location retrieved by the Fused Location Provider.
    private Location mLastKnownLocation;


    @Override
    public void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);

        nearby = getIntent().getStringExtra("nearby");

mAdapter = new ViewPagerAdapter(getSupportFragmentManager());

mPager = findViewById(R.id.pager);
mPager.setAdapter(mAdapter);

        // This is required to avoid a black flash when the map is loaded.  The flash is due
        // to the use of a SurfaceView as the underlying view of the map.
```

```
mPager.requestTransparentRegion(mPager);

        // Construct a FusedLocationProviderClient.
mFusedLocationProviderClient = LocationServices.getFusedLocationProviderClient(this);

getLocationPermission();
getDeviceLocation();
    }



    private void getDeviceLocation() {
        /*
         * Get the best and most recent location of the device, which may be null in rare
         * cases when a location is not available.
         */
        try {
            if (mLocationPermissionGranted&&statusCheckGPS()) {
                Task<Location>locationResult =
mFusedLocationProviderClient.getLastLocation();
locationResult.addOnCompleteListener(this, new OnCompleteListener<Location>() {
                    @Override
                    public void onComplete(@NonNull Task<Location> task) {
                        if (task.isSuccessful()) {
                            // Set the map's camera position to the current location of the device.
mLastKnownLocation = task.getResult();
                            if (mLastKnownLocation != null) {
getData();
EventBus.getDefault().postSticky(new Coordinate(
                                        new LatLng(mLastKnownLocation.getLatitude(),
mLastKnownLocation.getLongitude())));
                            }
                        } else {
Log.d("MYLOCATION", "Current location is null. Using defaults.");
Log.e("MYLOCATION", "Exception: %s", task.getException());
                        }
                    }
                });
            }
        } catch (SecurityException e) {
Log.e("Exception: %s", e.getMessage());
        }
    }

    public booleanstatusCheckGPS() {
        final LocationManager manager = (LocationManager)
getSystemService(Context.LOCATION_SERVICE);

        if (!manager.isProviderEnabled(LocationManager.GPS_PROVIDER)) {
buildAlertMessageNoGps();
            return false;
        }

        return true;
    }

    private void buildAlertMessageNoGps() {
        final AlertDialog.Builder builder = new AlertDialog.Builder(this);
builder.setMessage("Your GPS seems to be disabled. Enable?")
                .setCancelable(false)
                .setPositiveButton("Yes", new DialogInterface.OnClickListener() {
                    public void onClick(final DialogInterface dialog, final int id) {
startActivity(new
Intent(android.provider.Settings.ACTION_LOCATION_SOURCE_SETTINGS));
                    }
                })
                .setNegativeButton("No", new DialogInterface.OnClickListener() {
                    public void onClick(final DialogInterface dialog, final int id) {
dialog.cancel();
                    }
                });
        final AlertDialog alert = builder.create();
alert.show();
    }

    private void getLocationPermission() {
        /*
         * Request location permission, so that we can get the location of the
         * device. The result of the permission request is handled by a callback,
         * onRequestPermissionsResult.
         */
        if (ContextCompat.checkSelfPermission(this.getApplicationContext(),
android.Manifest.permission.ACCESS_FINE_LOCATION)
```

```
            == PackageManager.PERMISSION_GRANTED) {
mLocationPermissionGranted = true;
      } else {
ActivityCompat.requestPermissions(this,
               new String[]{android.Manifest.permission.ACCESS_FINE_LOCATION},
PERMISSIONS_REQUEST_ACCESS_FINE_LOCATION);
      }
   }

   @Override
   public void onRequestPermissionsResult(int requestCode,
                                @NonNull String permissions[],
                                @NonNull int[] grantResults) {
mLocationPermissionGranted = false;
      switch (requestCode) {
         case PERMISSIONS_REQUEST_ACCESS_FINE_LOCATION: {
            // If request is cancelled, the result arrays are empty.
            if (grantResults.length> 0
&&grantResults[0] == PackageManager.PERMISSION_GRANTED) {
mLocationPermissionGranted = true;
getDeviceLocation();
            }
         }
      }
   }

   @Override
   public booleanonCreateOptionsMenu(Menu menu) {
getMenuInflater().inflate(R.menu.menu, menu);

MenuItem item = menu.findItem(R.id.spinner);
      final Spinner spinner = (Spinner) item.getActionView();

ArrayAdapter<CharSequence> adapter =
ArrayAdapter.createFromResource(getApplicationContext(),
R.array.fragment_type, R.layout.spinner_item);

adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

spinner.setAdapter(adapter);
spinner.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {
```

```
            @Override
            public void onItemSelected(AdapterView<?> parent, View view, int position, long
id) {
                switch (spinner.getSelectedItem().toString()) {
                    case "Camera":
mPager.setCurrentItem(2);
                        break;
                    case "Map":
mPager.setCurrentItem(0);
                        break;
                    case "List":
mPager.setCurrentItem(1);
                        break;
                }
            }

            @Override
            public void onNothingSelected(AdapterView<?> parent) {

            }
        });

mPager.addOnPageChangeListener(new ViewPager.SimpleOnPageChangeListener() {
            @Override
            public void onPageSelected(int position) {
super.onPageSelected(position);
spinner.setSelection(position);
            }
        });


        return true;
    }



    public void getData() {
        String url = getUrl(mLastKnownLocation.getLatitude(),
mLastKnownLocation.getLongitude(), nearby);
GetNearbyPlacesDatagetNearbyPlacesData = new GetNearbyPlacesData();
getNearbyPlacesData.execute(url);
```

```
    }

    private String getUrl(double latitude, double longitude, String nearbyPlace) {

        StringBuilder googlePlacesUrl =
                new
StringBuilder("https://maps.googleapis.com/maps/api/place/nearbysearch/json?");
googlePlacesUrl.append("location=" + latitude + "," + longitude);
googlePlacesUrl.append("&rankby=" + "distance");
googlePlacesUrl.append("&type=" + nearbyPlace);
googlePlacesUrl.append("&key=" + getString(R.string.google_maps_key));
Log.d("getUrl", googlePlacesUrl.toString());
        return (googlePlacesUrl.toString());
    }

}
```

10.3.2. ViewPagerAdapter.java

```
package com.kbaquri.gaze.Second;
import android.support.v4.app.Fragment;
import android.support.v4.app.FragmentManager;
import android.support.v4.app.FragmentPagerAdapter;

public class ViewPagerAdapter extends FragmentPagerAdapter {

    public ViewPagerAdapter(FragmentManagerfm) {
        super(fm);
    }

    @Override
    public int getCount() {
        return 3;
    }

    @Override
    public Fragment getItem(int position) {
        switch (position) {

            case 0:
                return MyMapFragment.newInstance();
            case 1:
                return new MyPlacesListFragment();
            case 2:
                return new MyCameraFragment();
            default:
                return null;
        }
    }

}
```

10.3.3. Coordinate.java

```
package com.kbaquri.gaze.Second;
import com.google.android.gms.maps.model.LatLng;

public class Coordinate {
LatLnglatLng;

    public Coordinate(LatLnglatLng) {
this.latLng = latLng;
    }

    public LatLnggetLatLng() {
        return latLng;
    }
}
```

10.3.4. MyMapFragment.java

```
package com.kbaquri.gaze.Second;
import android.Manifest;
import android.annotation.SuppressLint;
import android.content.pm.PackageManager;
import android.os.Bundle;
import android.support.annotation.NonNull;
import android.support.v4.app.ActivityCompat;
import android.support.v4.content.ContextCompat;
import android.util.Log;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.BitmapDescriptorFactory;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.MarkerOptions;
import com.kbaquri.gaze.Api.NearbyPlaces;
import org.greenrobot.eventbus.EventBus;
import org.greenrobot.eventbus.Subscribe;
import org.greenrobot.eventbus.ThreadMode;
import java.util.HashMap;
import java.util.List;

public class MyMapFragment extends SupportMapFragment
        implements OnMapReadyCallback {

    private GoogleMapmMap;
LatLngtempLatLng = null;

    public static MyMapFragmentnewInstance() {
        return new MyMapFragment();
    }

    @Override
    public void onCreate(Bundle bundle) {
super.onCreate(bundle);

getMapAsync(this);
```

```
    }

    @SuppressLint("MissingPermission")
    @Override
    public void onMapReady(GoogleMapgoogleMap) {
mMap = googleMap;

mMap.setMyLocationEnabled(true);
    }

    private void showNearbyPlaces(List<HashMap<String, String>>nearbyPlacesList) {
        for (int i = 0; i<nearbyPlacesList.size(); i++) {
Log.d("onPostExecute", "Entered into showing locations");
MarkerOptionsmarkerOptions = new MarkerOptions();
            HashMap<String, String>googlePlace = nearbyPlacesList.get(i);
            double lat = Double.parseDouble(googlePlace.get("lat"));
            double lng = Double.parseDouble(googlePlace.get("lng"));
            String placeName = googlePlace.get("place_name");
            String vicinity = googlePlace.get("vicinity");
LatLnglatLng = new LatLng(lat, lng);
markerOptions.position(latLng);
markerOptions.title(placeName);
markerOptions.snippet(vicinity);
mMap.addMarker(markerOptions);

markerOptions.icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.H
UE_RED));
        }

        //move map camera
        if (tempLatLng != null) {
mMap.moveCamera(CameraUpdateFactory.newLatLng(tempLatLng));
mMap.animateCamera(CameraUpdateFactory.zoomTo(15));
        }
    }

    @Override
    public void onStart() {
super.onStart();
EventBus.getDefault().register(this);
    }
    @Override
    public void onStop() {
EventBus.getDefault().unregister(this);
super.onStop();
    }

    @Subscribe(threadMode = ThreadMode.MAIN, sticky = true)
    public void onNearbyPlaces(NearbyPlacesnearbyPlaces) {
showNearbyPlaces(nearbyPlaces.getNearbyPlacesList());
    }

    @Subscribe(threadMode = ThreadMode.MAIN, sticky = true)
    public void onCoordinate(Coordinate coordinate) {
tempLatLng = coordinate.getLatLng();
    }
}
```

10.3.5. MyPlacesListFragment.java

```
package com.kbaquri.gaze.Second;
import android.app.Activity;
import android.os.Bundle;
import android.support.annotation.Nullable;
import android.support.v4.app.Fragment;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;
import com.kbaquri.gaze.Api.NearbyPlaces;
import com.kbaquri.gaze.R;
import org.greenrobot.eventbus.EventBus;
import org.greenrobot.eventbus.Subscribe;
import org.greenrobot.eventbus.ThreadMode;
import java.math.BigDecimal;
import java.math.RoundingMode;
import java.util.HashMap;
import java.util.List;

public class MyPlacesListFragment extends Fragment {

ListViewlistView;
CustomAdapter adapter;

   @Override
   public View onCreateView(LayoutInflaterinflater, ViewGroup container, Bundle 
bundle) {
      View rootView = inflater.inflate(R.layout.fragment_places_list, container, false);

listView = rootView.findViewById(R.id.listView);

      return rootView;
   }

   public class CustomAdapter extends ArrayAdapter<HashMap<String, String>> {
      private final Activity _context;
```

```
        private List<HashMap<String, String>>nearbyPlacesList;

        public CustomAdapter(Activity context, List<HashMap<String,
String>>nearbyPlacesList) {
            super(context, R.layout.fragment_places_list_item, nearbyPlacesList);
this._context = context;
this.nearbyPlacesList = nearbyPlacesList;
        }

        @Override
        public View getView(int position, View convertView, ViewGroup parent) {

LayoutInflaterinflater = _context.getLayoutInflater();
            View rowView = inflater.inflate(R.layout.fragment_places_list_item, null, true);
TextViewtxtLat = rowView.findViewById(R.id.lat);
TextViewtxtLng = rowView.findViewById(R.id.lng);
TextViewtxtTitle = rowView.findViewById(R.id.title);
TextViewtxtSnippet = rowView.findViewById(R.id.snippet);

            HashMap<String, String>googlePlace = nearbyPlacesList.get(position);
            double lat = Double.parseDouble(googlePlace.get("lat"));
            double lng = Double.parseDouble(googlePlace.get("lng"));
            String placeName = googlePlace.get("place_name");
            String vicinity = googlePlace.get("vicinity");

txtLat.setText(round(lat, 7) + "");
txtLng.setText(round(lng, 7) + "");
txtTitle.setText(placeName);
txtSnippet.setText(vicinity);

            return rowView;
        }
    }

    public static double round(double value, int places) {
        if (places < 0) throw new IllegalArgumentException();

BigDecimal bd = new BigDecimal(value);
        bd = bd.setScale(places, RoundingMode.HALF_UP);
        return bd.doubleValue();
    }
```

```
   @Override
   public void onStart() {
super.onStart();
EventBus.getDefault().register(this);
   }

   @Override
   public void onStop() {
EventBus.getDefault().unregister(this);
super.onStop();
   }

   @Subscribe(threadMode = ThreadMode.MAIN, sticky = true)
   public void onNearbyPlaces(NearbyPlacesnearbyPlaces) {
      adapter = new CustomAdapter(getActivity(), nearbyPlaces.getNearbyPlacesList());
listView.setAdapter(adapter);
   }

}
```

10.4 XML Files

10.4.1. ActivityList.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.v7.widget.RecyclerView
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:id="@+id/recycler_view"
android:background="@color/colorPrimary"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:listitem="@layout/activity_list_item" />
```

10.4.2. ActivityListItem.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayoutxmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent"
android:layout_height="160dp">

<ImageView
android:id="@+id/item_background"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:background="@color/colorPrimaryDark"
android:scaleType="fitXY" />

<TextView
android:id="@+id/item_title"
android:layout_width="match_parent"
android:layout_height="wrap_content"
```

```
android:layout_margin="20dp"
android:fontFamily="monospace"
android:text="@string/sample_item_text"
android:textColor="#fff"
android:textSize="25sp"
android:textStyle="bold" />
</RelativeLayout>
```

CHAPTER 11

REFERENCES

1. Bill Philips & Brian Hardy. *Android Programming: The Big Nerd Ranch Guide (Big Nerd Ranch Guides).*

2. Greg Nudelman, *Developers. Android Design Patterns: Interaction Design Solutions fo*

3. Ian G. Clifton. *Android User Interface Design: Turning Ideas and Sketches into Beautifully Designed Apps.*

4. Dave Smith & Jeff Friesen. *Android Recipes: A Problem-Solution Approach.*

5. *Introduction to Android: http://developer.android.com/guide/index.html.*

6. *E*d Burnette. *Hello, Android: Introducing Google's Mobile Development Platform (Pragmatic Programmers).*

7. Mario Zechner. *Begining Android Games.*

8. ZigurdMednieks, Laird Dornin, G. Blake Meike& Masumi Nakamura. *ProgrammingAndroid.*