



(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

Centeract 2.0

A Report for the Evaluation 3 of Project 2

Submitted by

Yash Garg

(1613102013 /

16SCSE1011896)

in partial fulfillment for the award of the degree

of

Bachelor of Technology

IN

Computer Science and Engineering

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

Under the Supervision of

Miss Swati Singh(Assistant Professor)

APRIL / MAY- 2020

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
1.	Abstract	3
2.	Introduction	4
3.	Existing System	6
4.	Proposed system	8
5.	Implementation or architecture diagrams	9
6.	Output / Result / Screenshot	29
7.	Conclusion/Future Enhancement	37
8.	References	38

1. Abstract

Android is the first complete, open and free mobile platform. It is developed by the *Open Handset Alliance*, a group of more than 30 technology and mobile companies. It is supported by Google and this project uses a Google Android Mobile SDK 1.0 for testing an application.

The aim of our project is to develop an integrated Web and mobile phone application prototype for connecting users with other users using Android. The Web interface is used by the users who has to download and upload their profiles and they use the web interface to post their activities through their respective accounts. The users must belong to an account. The mobile application is to be installed on the mobile phone.

This project focuses on developing a web application that is developed using java and Android where the users can have their profiles to know about each other. This web application increases the security allowing the service seeker to know the person who is going to connected by him. Both users will have an option to choose the person according to the ratings and reviews that are posted previously.

2.Introduction

Center act is an application which works on android development system. This is an application which help an individual to store the contacts which will be accessed only through this app only and can't be seen in mobile's phonebook and we can also store the details of the contact in this app.

Accessing doesn't mean only calling , u can also message or mail the contact through this application

Beside this this application also has access to the basic functionality of the mobile like u can use camera, gallery and can also play games in this application.

In this app user has to first register himself by providing the details like name , email , username , password etc. and after registering user can simply login through their username and password and can use the app according to their use

By registering the user we can also have track of the activities done by the user

Center act works on android which is a open and free mobile source and uses Google Android SDK 1.0 for testing application.

Two words describe what makes Center Act different: technology and speed. Our iPhone, Android and Blackberry applications allow you to track and interact with your efforts in real time. EnterAct will automatically route profiles of users and even contact you the instant users in contact. You can email, SMS or call your contacts with one tap of your touch screen. Whether you have 5 contacts or 1,000,Center Act will instantly create a short list of profiles and connect you with them.

3.Existing System

1.Waze-Available for both iOS and Android, the [Waze](#) navigation app is a beautiful little app with big ambitions. Not only do you get the latest traffic information through Waze, but that information gets provided to you by other drivers using the app. It's a small community of drivers who help each other out, with just about anything. Whether it's accidents or police traps, you can rest assured Waze has you covered — and it's free.

2. Maps.me-This app seems to be one of the more straightforward apps on this list. It doesn't have a lot of fancy features or any cool voice navigation. However, what it lacks in flashiness, it more than makes up for in content.

Plain and simple, [Maps.me](#) is one of the most detail-oriented navigation apps containing highly detailed maps of some of the most isolated and unknown places. And it works offline, which is a significant plus in those remote adventure travel destinations. It's free for both iOS and Android.

3.OSM-If you need to find your way offline, look no further than [OSM](#). The app delivers offline maps in a big way by hosting automatic rerouting. No Internet? No worries! Along with the use of offline maps, it even houses a speedometer, so the app alerts you when you're speeding.

It's available for Android for \$5.99 and free for iOS users.

4.MotionX GPS-Although it's one of the earlier entries in the life of navigational apps, it still packs quite a punch. The [MotionX GPS](#) features a slew of abilities that makes it hard to beat. For iOS users, you can also record your speed and even your max speed.

You get live altitude graphs, track personal waypoints, and can even share geotagged photos to make your experience on the road even more memorable. Overall, at \$1.99, the MotionX GPS is a great app for the iPhone users of the world.

4. Proposed System

CenterAct is a suite of tools designed specifically to login by user and use applications and use app to connect with data of the android devices.

With your input details, we allow user to make account or to login as a user. Users complete the CenterAct questionnaire through links in your application and can make a call or message or E-mail the people. The user can access other details like gallery, videos, images, calculator etc with the help of app.

This project uses the Google Android platform to build an application where users can upload their profiles, users can use and can see applications .The required application and mobile service providers can quickly identify the profiles and contact the persons. The overall concept is where users would use the applications through an innovative web application. A user using a smart phone running Google Android (for the project this will be an emulator) could either through GPS or by entering a specified code could request to show all applicants details needing for the profiles. The application will use SOAP API to retrieve database of users via web services the user could get extra information about the other users. Our Model Profiles take into account descriptions. The most important information used in developing our Model Profiles is your feedback on what you expect from your new user. This is where generic benchmarking may fall short.

5.Implementation

This section explains about the individual functionality and requirements of two major sections of this project

1- Mobile Application

2- Application

These requirements are identified from the sequence diagrams that are drawn to identify the flow of these two applications and the

communication between them.

5.1.1MOBILE APPLICATION:

Application requires service providers User Name and password. When a service provider starts this application a menu showing various options. If he/she selects the fun app list of all sessions will displayed , he/she can retrieve information of applications from these sessions and they can call, message or send mails to the userss . If he/she clicks on the user details then information of user will be displayed. When clicks on the user name then a window consists of various information or further fields where in you type name, last name, address and contact information will be displayed. When he/she clicks on Email then the user query the database and gets the information of the user in the area provided. Then a mail will send to the specified mail address.

5.1.2WEB APPLICATION:

This application starts with a Flash intro and it later redirects the user into the web page where he can only login and use the services according to his account type. It concentrates on the functionality and requirements for different type of

users. This application has 2 types of services

1.Fun App

2.User Details

These requirements are individually identified for each type of user and it is explained in detail as followed

ANDROID COMPONENTS

The followings gives a short overview of important Android components.

Activity

Activity represents the presentation layer of an Android application. A simplified (and slightly incorrect) description is that an Activity is a screen. This is slightly incorrect as Activities can be displayed as dialogs or transparent. An Android application can have several Activities.

and ViewGroups

Views are user interface widgets, e.g. buttons or text fields. The base class for all Views is `android.view.View`. Views often have attributes which can be used to change their appearance and behavior.

A ViewGroup is responsible for arranging other Views e.g. they are layout manager. The base class for a layout manager is `android.view.ViewGroups`. ViewGroup also extends View. ViewGroups can be nested to create complex layouts. You should not nestle ViewGroups too deeply as this has a negative impact on performance.

Intents

Intents are asynchronous messages which allow the application to request functionality from other components of the Android system, e.g. from Services or Activities. An application can call a component directly (explicit intent) or ask the Android system to evaluate registered components for a certain Intents (implicit intents). For example the application could implement sharing of data via an Intent and all components which allow sharing of data would be available for the user to select. Applications register themselves to an intent via

an IntentFilter. Intents allow to combine loosely coupled components to perform certain tasks.

Services

Services perform background tasks without providing an UI. They can notify the user via the notification framework in Android.

ContentProvider

ContentProvider provides an structured interface to data. Via a ContentProvider your application can share data with other applications. Android contains an SQLite database which is frequently used in conjunction with a ContentProvider to persists the data of the ContentProvider.

BroadcastReceiver

BroadcastReceiver can be registered to receives system messages and Intents. BroadcastReceiver will get notified by the Android system if the specified situation happens. For example a BroadcastReceiver could get called once the system completed its boot process or if a phone call is received.

Every Android application runs in its own process

(with its own instance of the Dalvik virtual machine).

Whenever there's a request that should be handled by a particular component,

- Android makes sure that the application process of the component is running,
- starting it if necessary, and
- That an appropriate instance of the component is available, creating the instance if necessary.

Application's Life Cycle

A Linux process encapsulating an Android application is created for the application when some of its code needs to be run, and will remain running until

1. it is no longer needed, **OR**
2. the system needs to reclaim its memory for use by other applications.

unusual and fundamental feature of Android is that an application process's lifetime is not directly controlled by the application itself.

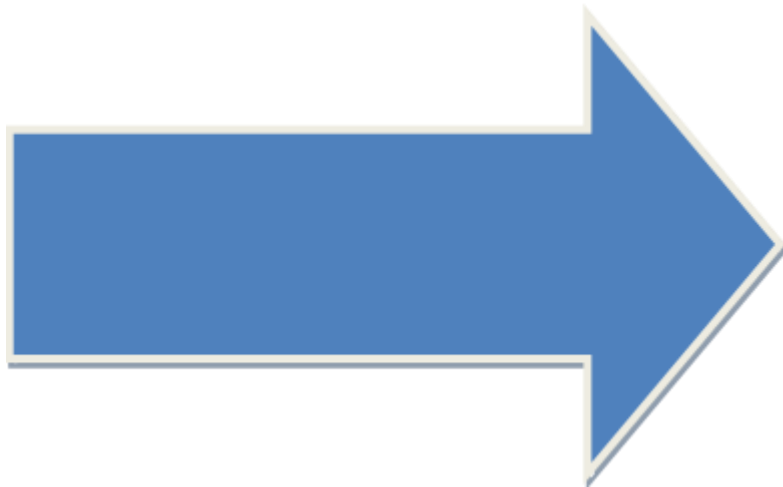
Instead, it is determined by the system through a combination of

1. the parts of the application that the system knows are running,
2. how important these things are to the user, and
3. how much overall memory is available in the system.

Component Lifecycles

Application components have a **lifecycle**

1. A **beginning** when Android instantiates them to respond to intents through to an **end** when the instances are destroyed.

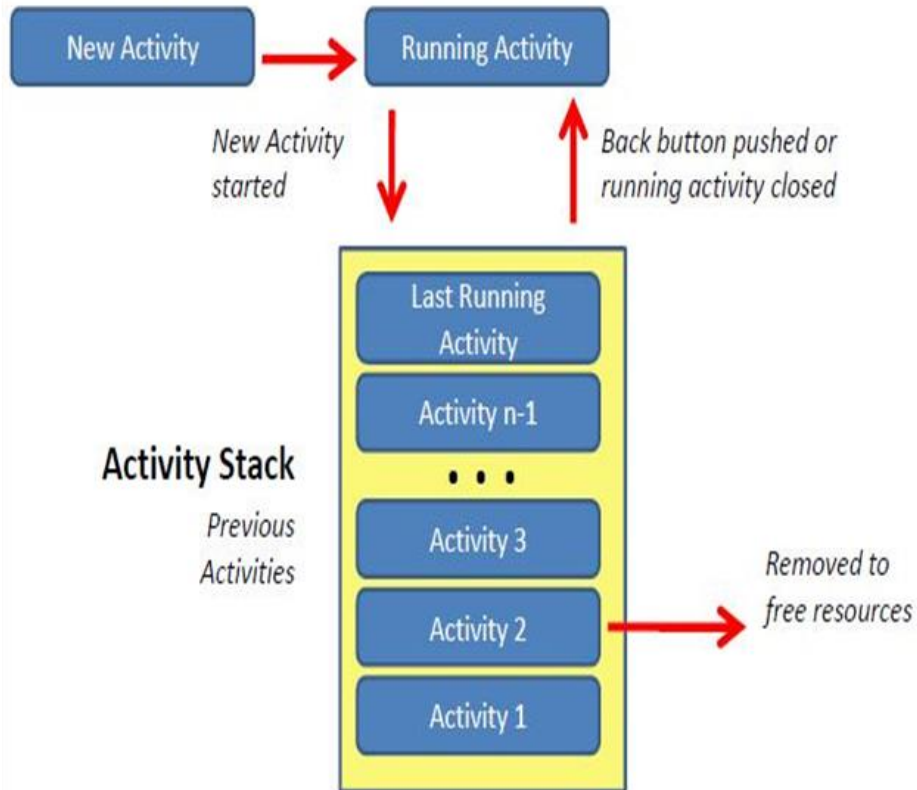


2. In **between**, they may sometimes be *active* or *inactive*, or -in the case of activities-*visible* to the user or *invisible*.

End

Activity Stack

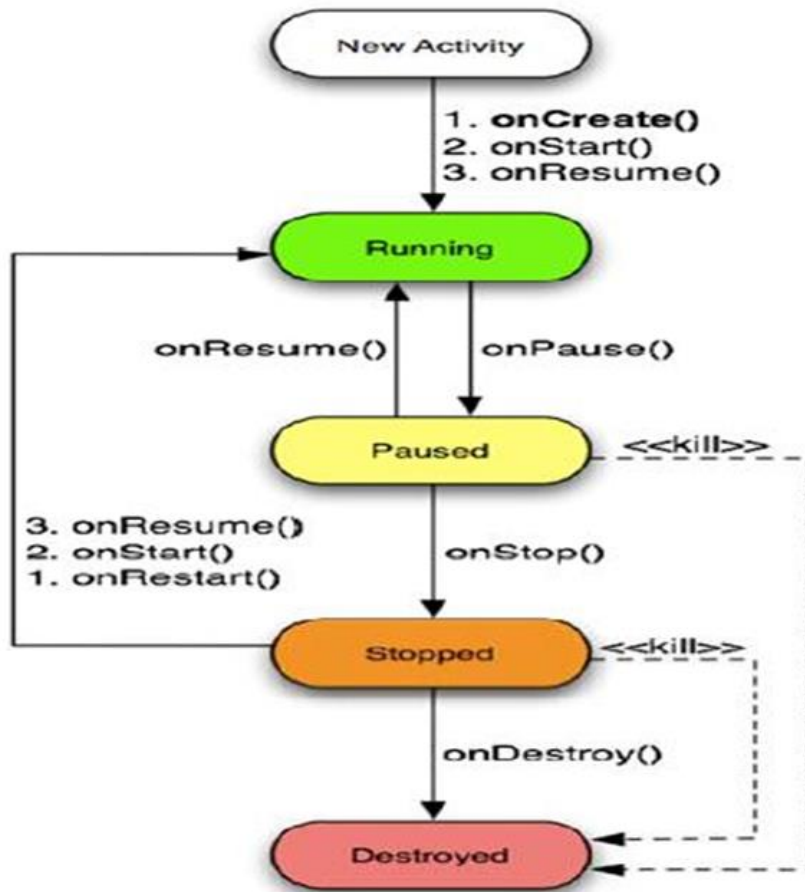
- Activities in the system are managed as an **activity stack**.
- When a new activity is *started*, it is placed on the *top* of the stack and becomes the running activity -- the previous activity always remains below it in the stack, and will not come to the foreground again until the new activity exits.
- If the user presses the *Back Button* the next activity on the stack moves up and becomes active.



Life Cycle States

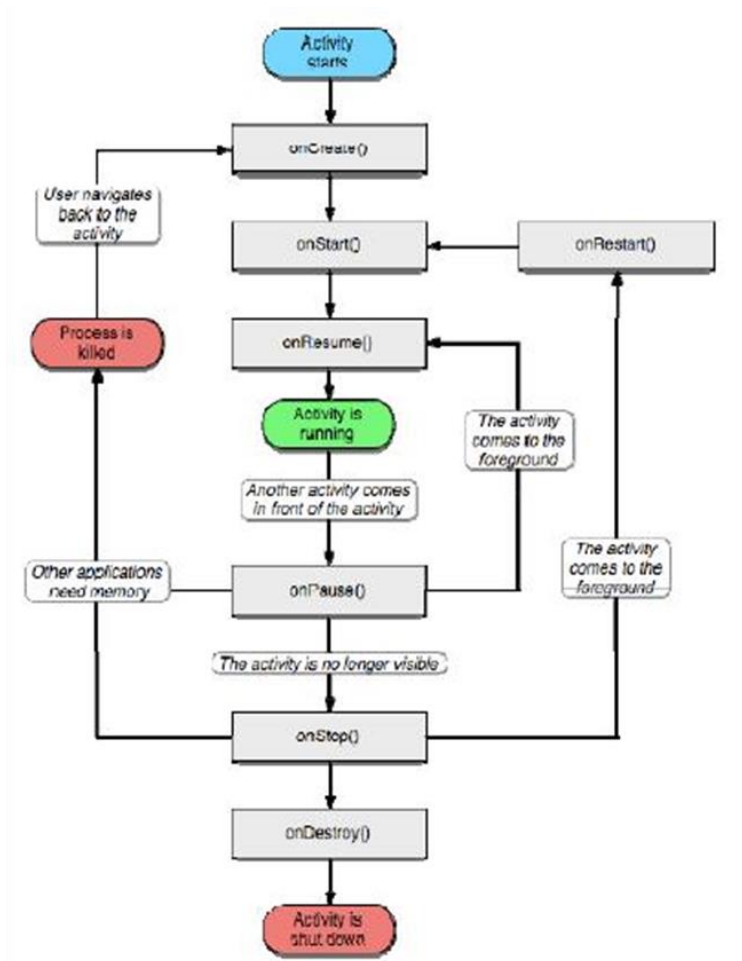
An activity has essentially three states:

1. It is *active or running*
2. It is *paused* or
3. It is *stopped* .



An activity has essentially three states:

1. It is **active or running** when it is in the *foreground* of the screen (at the top of the *activity stack* for the current task). This is the activity that is the focus for the user's actions.
2. It is **paused** if it has lost focus but is still visible to the user. That is, another activity lies on top of it and that new activity either is *transparent* or *doesn't cover the full screen*. A paused activity is completely *alive* (it maintains all state and member information and remains attached to the window manager), but can be killed by the system in extreme low memory situations.
3. It is **stopped** if it is completely *obscured* by another activity. It still retains all state and member information. However, *it is no longer visible* to the user so its window is hidden and it will often be killed by the system when memory is needed elsewhere.



Life Cycle Events

If an activity is paused or stopped, the system can drop it from memory either by asking it to finish (calling its **finish()** method), or simply killing its process. When it is displayed again to the user, it must be completely restarted and restored to its previous state. As an activity transitions from state to state, it is notified of the change by calls to the following protected *transition* methods:

All of these methods are **hooks** that you can override to do appropriate work when the state changes. All activities must implement **onCreate()** to do the initial setup when the object is first instantiated. Many activities will also implement **onPause()** to commit data changes and otherwise prepare to stop interacting with the user. The seven transition methods define the

entire lifecycle of an activity. The **entire lifetime** of an activity happens between the first call to `onCreate()` through to a single final call to `onDestroy()`. An activity does all its initial setup of "global" state in `onCreate()`, and releases all remaining resources in `onDestroy()`.

Visible Lifetime

The **visible lifetime** of an activity happens between a call to `onStart()` until a corresponding call to `onStop()`. During this time, the user can see the activity on-screen, though it may not be in the foreground and interacting with the user. The `onStart()` and `onStop()` methods can be called multiple times, as the activity alternates between being visible and hidden to the user. Between these two methods, you can maintain resources that are needed to show the activity to the user.

Foreground Lifetime

The **foreground lifetime** of an activity happens between a call to `onResume()` until a corresponding call to `onPause()`. During this time, the activity is in front of all other activities on screen and is interacting with the user. An activity can frequently transition between the *resumed* and *paused* states — for example,

- `onPause()` is called when the device goes to sleep or when a new activity is started,
- `onResume()` is called when an activity result or a new intent is delivered.

Killable States

- Activities on killable states can be terminated by the system *at anytime after the method returns, without executing another line of the activity's code.*
- Three methods (`onPause()`, `onStop()`, and `onDestroy()`) are *killable*.
- `onPause()` is the only one that is guaranteed to be called before the process is killed — `onStop()` and `onDestroy()` may not be.
- Therefore, you should use `onPause()` to write any persistent data (such as user edits) to storage

ANDROID DEVELOPMENT TOOLS

What are the Android Development Tools?

- Google provides the Android Development Tools (ADT) to develop Android applications with Eclipse. ADT is a set of plug-in which extended the Eclipse IDE with Android development capabilities.
- ADT contains all required functionality to create, compile, debug and deploy Android applications from the Eclipse IDE and from the command line. Other IDE's, e.g. IntelliJ, are also reusing components of ADT.
- ADT also provides an Android device emulator, so that Android applications can be tested without a real Android phone.

How to develop Android Applications

- Android applications are primary written in the Java programming language. The Java source files are converted to Java class files by the Java compiler.
- Android provides a tool dx which converts Java class files into a dex (Dalvik Executable) file. All class files of one application are placed in one compressed .dex file. During this conversion process redundant information in the class files are optimized in the .dex file. For example if the same String in different class file is found, the .dex file is stored only once and reference this String in the corresponding classes.
- .dex files are therefore much smaller in size then the corresponding class files.

The .dex file and the resources of an Android project, e.g. the images and XML files are packed into an .apk(Android Package) file. The program aapt (Android Asset Packaging Tool) perform this packaging.

- The resulting .apk file contains all necessary data to run the Android application and can be deployed to an Android device via the "adb" tool.
- The Android Development Tools (ADT) allows that all these steps are performed transparent to the user; either within Eclipse or via the command line.
- If you use the ADT tooling you press a button or run a script and the whole Android application (.apk file) will be created and deployed.

ANDROID VIRTUAL DEVICE - EMULATOR

What is the Android Emulator?

- The Android Development Tools (ADT) include an emulator to run an Android system. The emulator behaves like a real Android device (in most cases) and allows you to test your application without having a real device.

You can configure the version of the Android system you would like to run, the size of the SD card, the screen resolution and other relevant settings. You can define several devices with different configurations.

- Via the emulator you select which device should be started, you can also start several in parallel. These devices are called "Android Virtual Device" (AVD).
- The ADT allow to deploy and run your Android program on the AVD.

Emulator Shortcuts

Obviously you can use the emulator via the keyboard on the right side of the emulator. But there are also some nice shortcuts which are useful.

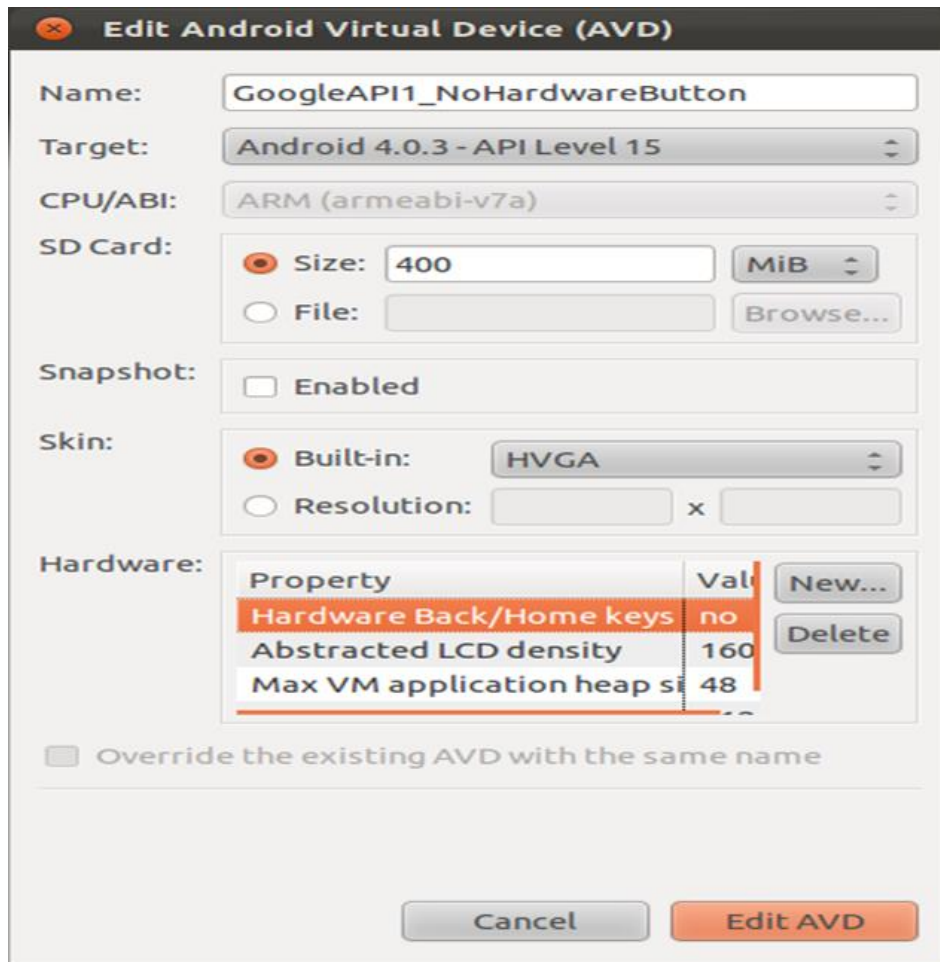
- **Alt+Enter** Maximizes the emulator. Nice for demos.
- **Ctrl+F11** changes the orientation of the emulator.
- **F8** Turns network on / off.

Performance

- Try to use a smaller resolution for your emulator as for example HVGA. The emulator gets slower the more pixels its needs to render as it is using software rendering.
- Also if you have sufficient memory on your computer, add at least 1 GB of memory to your emulator. This is the value "Device ram size" during the creation of the AVD.
- Also set the flag "Enabled" for Snapshots. This will save the state of the emulator and let it start much faster.

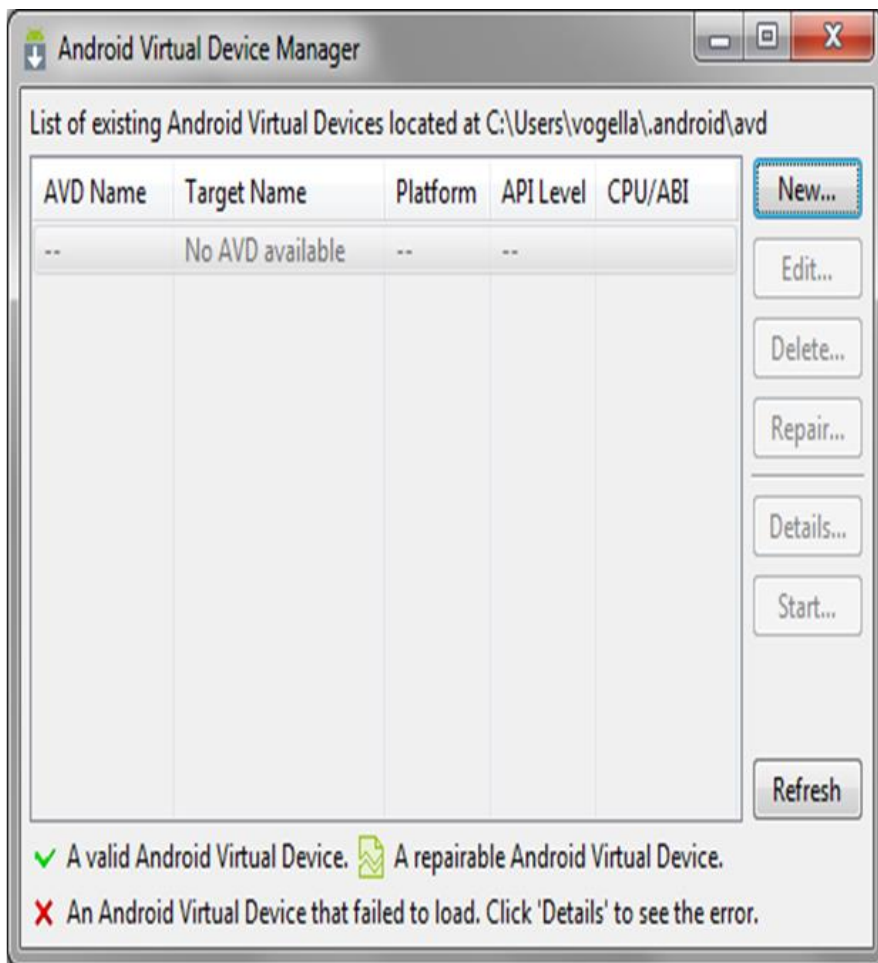
Hardware button

Android 4.0 introduced that devices do not have to have hardware button anymore. If you want to create such an AVD, add the "Hardware Back/Home keys" property to the device configuration and set it to "false".

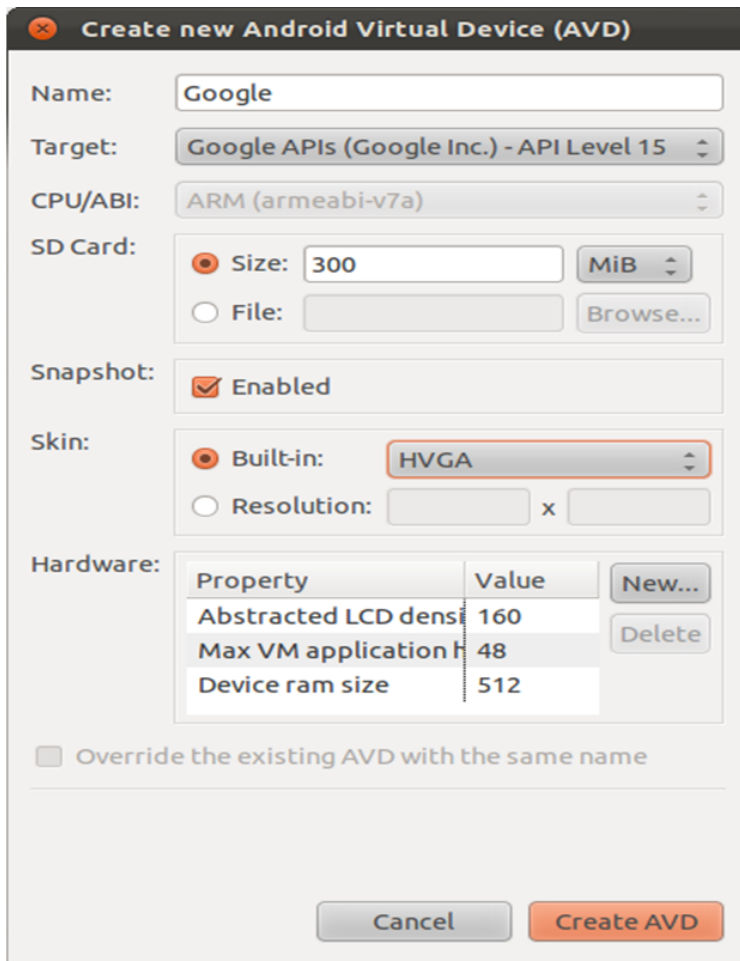


CREATE AND RUN ANDROID VIRTUAL DEVICE

To define an Android Virtual Device (ADV) open the "AVD Manager" via Windows → AVD Manager and press "New".



Enter the following.



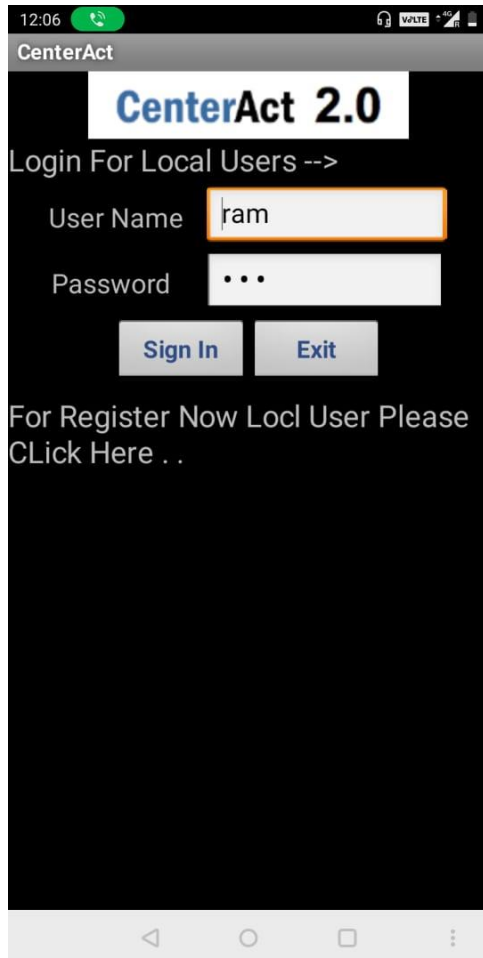
We can also select the box "Enabled" for Snapshots. This will make the second start of the virtual device much faster.






At the end press the button "Create AVD". This will create the AVD configuration and display it under the "Virtual devices".

To test if your setup is correct, select your device and press "Start".

After (a long time) your AVD starts. You are able to use it via the mouse and via the virtual keyboard of the emulator.

6.Output



12:06     

CenterAct

REGISTRATION FORM

Name

E-Mail


User-Name

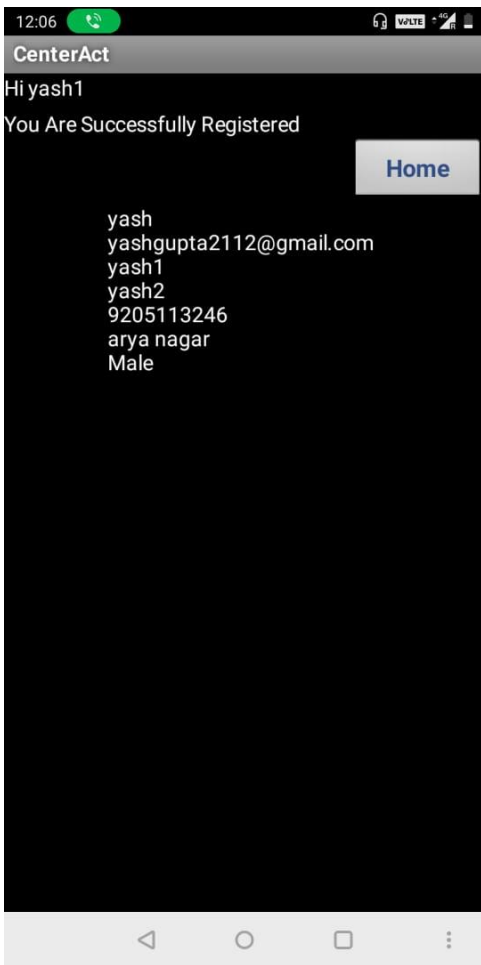
Password

Phone No.

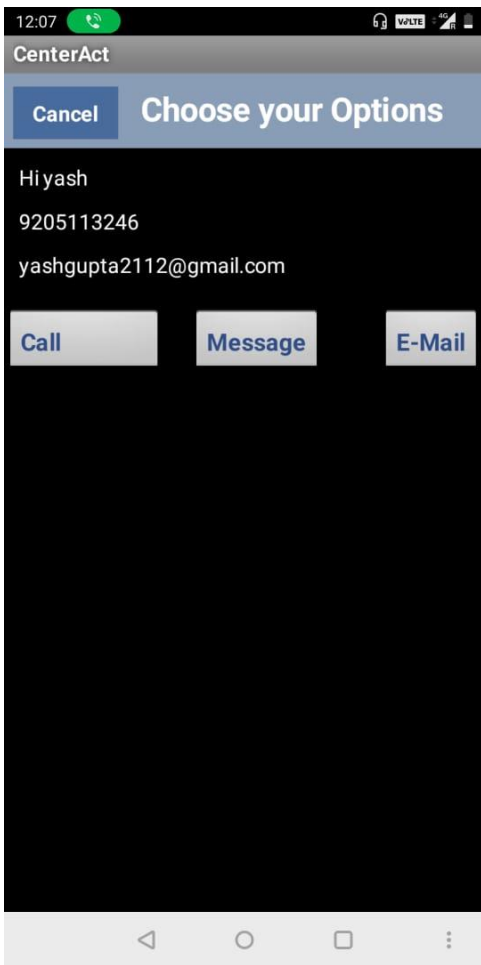
Address

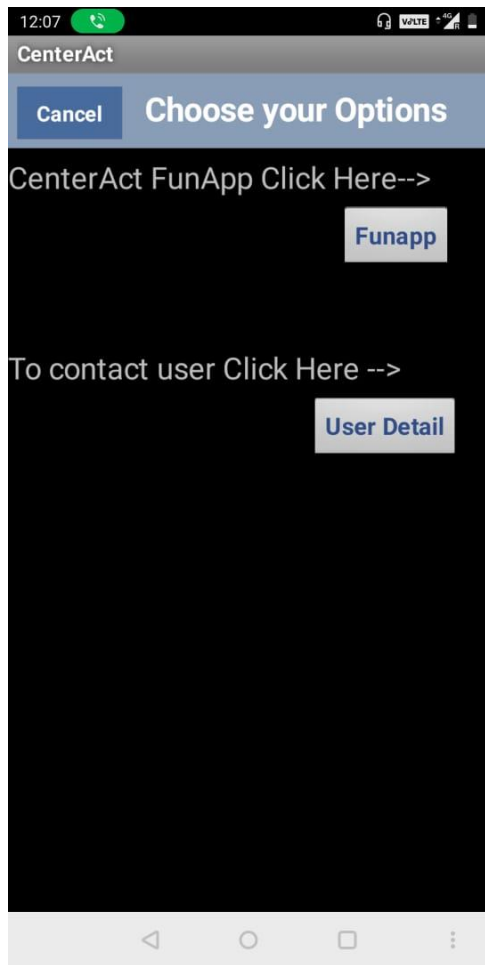
Gender Male Female

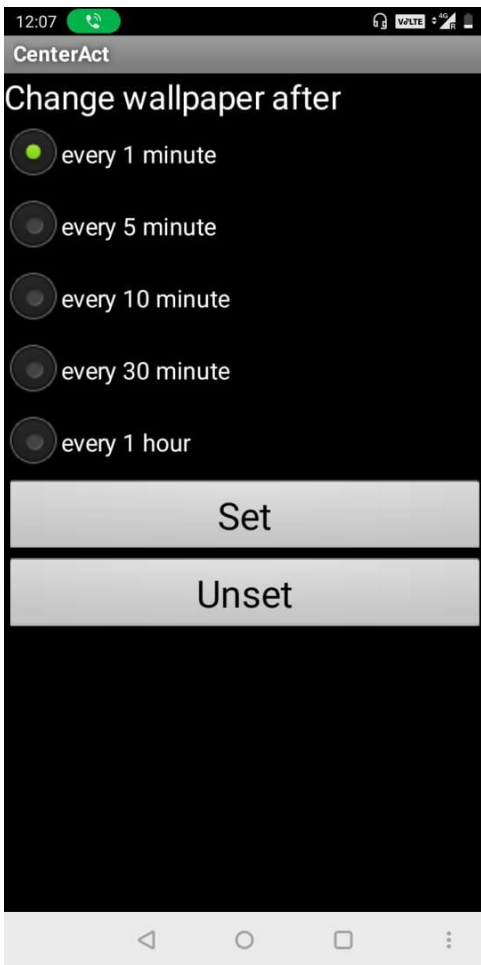


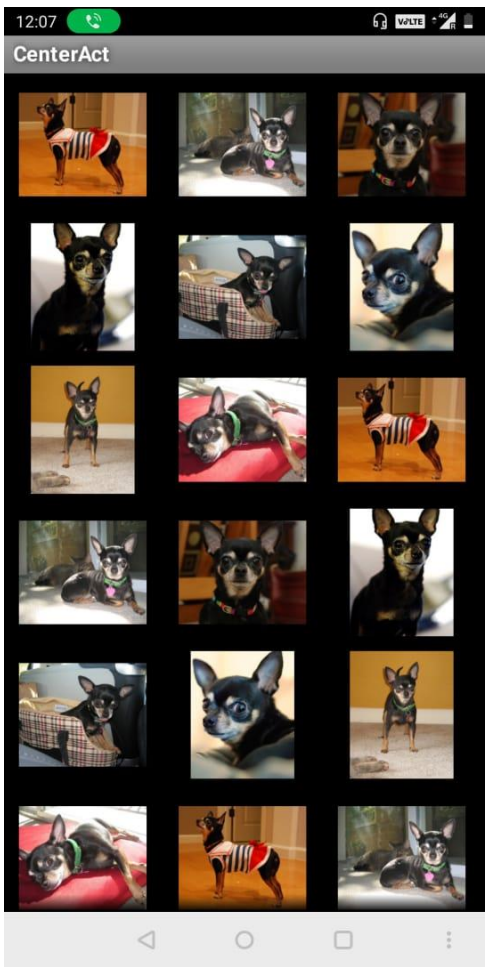












7.Conculsion

Now cell phone is the major part of everyone's life. Android making them more and more users interactive by providing lots of applications and services. Its large touch pad and sensor mode provide easy access to user. As all the mobile applications are build in java but android does not have any java virtual machine, there is a special virtual machine which is built for Android that is Dalvik virtual machine which compiles the code in .dev form. Android support for real time application, providing four different directions making it a real time system.

8.reference

1. —About the Android open source project||<http://source.android.com/about/index.html>.
2. Android Developers. 21 July 2009. <http://developer.android.com/guide/basics/what-isandroid.html>.
3. "Google Projects for Android". www.code.google.com. Google Inc. 2011. Archived from the original on 2011-02-23.
 1. <http://www.webcitation.org/5wiw1JXa2>.
 2. [http://en.wikipedia.org/wiki/Android_\(operating_system\)](http://en.wikipedia.org/wiki/Android_(operating_system))
 3. <http://www.lextrait.com/Vincent/implementations.html>
 4. <http://www.seeingwithsound.com/android.html>
5. Medical application for Android powered mobile systems by Gray S. Tyson and Ford Tyson in CSI communication 8 April 2011. Journal of the American Geriatrics Society, Vol. 58,No. 3, April 2010.
6. Open Handset Alliance, — Android overview,||
<http://www.openhandsetalliance.com/>
7. android_overview.html.
8. Research paper on —Evaluating Android OS for Embedded Real-Time Systems||. CISTER Research Centre, School of Engineering of the Polytechnic Institute of Porto in Euromicro Conference on Real- time systems (ECRTS- 2010) july 6, 2010.

[11] Technical trends – smart phones, smart people by Harikumar P Travancore Analytics, Technopark-Trivandrum, Kerala in CSI communication 11 april 2011.http://www.csi-india.org/c/document_library/

