Project Report of Major Project

on

Android TO-DO Task Manger App

Submitted to

I.K. GUJRAL PUNJAB TECHNICAL UNIVERSITY, JALANDHAR

In partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology in

Computer Science Engineering

Submitted By

Anurag Kumar

PTU Roll No- 1326453

November 2016

Under the guidance of

Academic Tutor

Er. Harkomal Preet Kaur

Assistant Professor

CSE Department

To

Department of Computer Science and Engineering

Swami Vivekanand Institute of Engineering & Technology, Banur, Punjab

Declaration

I hereby declare that this project report entitled

Android TO-DO Task Manger App

is written by me and is my own effort and that no part has been plagiarized without citations.

STUDENT:______________________________ DATE:__________
(ANURAG KUMAR)

SUPERVISOR:___________________________ DATE:__________
(Er. HARKOMAL PREET KAUR)

Acknowledgment

This project is a hard piece of work. This project is been developed as a partial vision. This project involved a number of people sharing their views and ideas and mentoring my way through. I would like to acknowledge and thank them all for their precious time and ideas. I would like to thank Ms. Kawalpreet Kaur (HOD, CSE Department, SVIET, Banur, Punjab), Er. Vikas Zandu (Asst. Professor, SVIET, Banur, Punjab), Er. Harkomal Preet Kaur(Asst. Professor, SVIET, Banur, Punjab), I would also like to acknowledge my SVIET ACM Student Chapter under which I partially contributed in creating my first project. i would also like to thank my parents and friends who helped me a lot in finalizing this project within the limited time frame.

CONTENT INDEX

Introduction

The purpose of this project is to develop an Android Application that can be utilized to maintain a task manager or TO – DO list. Also an innovative idea added to this project is that this application creates a profile for the user and his/her tasks, to fulfill two purposes.

1. Synchronizing tasks list on the application cloud.
2. Making a users task ethics visible globally to all the application users.

This application helps the user to remember all tasks that he/she had left for any scheduled time. And also reminds the user regularly about the task until it is completed. The reminding feature is automated, i. e., it doesn't require any user interactions but it can be fine tuned according to the needs of individual users.

Introducing cloud synchronization to the picture helps in many ways. Any user can easily shift between there mobile phones without the need to copy all the tasks that they maintained on the application as it saves all the data on the cloud itself. And it also helps to quantify the users work ethics regarding those tasks. Other users globally can view your personal details with the noted ethics which can help the user to generate a positive character. This ethic measure is consistently upgraded with every new task that user provides to the application and amount of total tasks held by the application will perform a very important role in this measurement.

The term Ethics that have been used in many instances in this document refers to a simple quantification of the users willingness to completed tasks. This measure will easily define the occurrence of the associated user's completed tasks count ratio to the overall count of tasks. Many other factors will play role in measuring ethic quantity such as how much post the due date the task is finally completed or what percentage of task is completed under due date in case of multiple entities within a composite task. The initial ethic count for any user will be zero globally. And will increase with every task input by the associated user.

Android Applications is the paradigm used for this project. The reason behind this is that a task manager is supposed to be always near the intended user so as to remind the user on time. And in today's time, Android device is the most common hand-held device globally. It can help the application to reach the most amount of users worldwide efficiently.

What design choices did you have along the way, and why did you make the choices you made?

As of the design choices regarding this project, we were hell-bent on Android Platform due to it's immense popularity worldwide. Providing us with a vast society of users already available to accept the application. Also, we are fan of the Material Design UI idea generated by android.

What was the most difficult part of the project?

Developing a well established and efficient Web Service for the Application was the most difficult part of this project, As we wished to add API possibilities for future 3rd party applications embedded from the start, we tried to make web services as generic as possible which led to some implementation difficulties and complexities.

Why was it difficult?

Major difficulty was simply handling the individual threads of application wisely. As many API implementations were developed in generic format and this same service provider will be used by every application initiated by different users, the idea of multiple calling of service simultaneously needed to be addressed as a threat to overall functionality of services. The data is needed to be persistent throughout the application. Multiple Transactions needed to be handled wisely.

How did you overcome the difficulties?

To overcome such difficulties, we used a very powerful library known as Hibernate ORM, which is used commonly for maintaining persistent data in a database by being a middle man between the actual database and the service that is calling the database. It also handles multiple transactions very wisely without a sweat.

What did you learn?

By making this project, we learned how to build an efficient Android Application for any basic purpose or having advanced purpose in task management genre. We also learned to develop a Web Service to connect to that Android Application to form a Cloud application that can save user information and data at the cloud server only to manipulate at application level. We also understood how to handle multiple simultaneous threads over a cloud server for persistent data management.

So the basic problem statement of this project is to manage and manipulate TO-DO task list of several users unique by user-ID in an efficient and user friendly approach. Which is henceforth solved by implementing the problem in Android application for user friendly aspects and Web Services for cloud backup to solve management and manipulation at cloud level with data efficiency and persistence.

Background

This section of the document refers to all the pre-requisites for fellow reader so as to understand all about the project and further documentation of the project. The reader must have some knowledge regarding the Computer Science field and its basic commodities including the application life-cycle, database and persistence knowledge, understanding of ER diagrams for the database model implementation etc.

Reader must also have understanding towards the topic of Web Services and APIs as most of the data manipulation in this project will be done on cloud web services. Understanding of such is thereafter a basic necessity to journey further into the project documentation.

Also the reader should have relative basic knowledge regarding Android Application Development and Android Studio's role in this field as the development environment. Understanding Android UI layout and relative java building of that UI layout is important for the project as all the implementation of this project is useless if not called through an Android Application filled with implementations of the web service that we have build. Also Android Manifest plays an important role in building an Applications bone structure so its background is also needed for the reader to leap further in the project.

Some basic knowledge for Android development with the Help of Android Studio can be easily found at the below mentioned URL, this reference will help in further understanding of different technologies and structures used in this document to justify the Android part of project.

URL: https://developer.android.com/training/index.html

Below is a link for related open source Android Application development projects for gathering the overall scope and power of this knowledge. This is uploaded to Github which is a very common service where developers upload their open source projects for other developers to understand, help improve and maintain in future.

URL: https://github.com/pcqpcq/open-source-android-apps

Project Karma is also hosted on github as an open source project under Android development catalog for project maintenance and version control. Link is mentioned below.

URL: https://github.com/anuragkumarak95/Karma

There are many use case scenarios for this application, as being a universal TO-DO task list management application it is not bound to a certain user society. All groups of users can easily use it. Most common use under our opinion would be regular occupants that need to maintain there daily target jobs for the day. All company now-a days work on a daily target schedule and in such a manner this application can be very useful as it also encourages task completion under due time with the help of its rewarding system for task completion using ethics count.

Some irregular users may include home based task management like the grocery list and things to complete today and also student users can be encouraged to use this for maintaining upcoming assignment due dates and completion in modular fashion due to its mini-task nature involving many task in a task list. Will help them to be regular in there assignments and also gather work ethic from certain student scenarios. Which can help them to maintain a work ethic in future too. Generating a positive nature in people.

Also it can help remember tasks that where assigned quite a significant time earlier and the user may have forgot to complete it. Such like tender approvals that could be assigned for 6-8 months further in time can be maintained easily as the application will remind you as the due dates come near.

What is Android?

Developer	Google Open Handset Alliance
Written in	Java (UI), C (core), C++
OS family	Unix-like
Latest Version	Android 7.1 Nougat

Android is a mobile operating system developed by Google, based on the Linux kernel and designed primarily for touchscreen mobile devices such as smartphones and tablets. Android's user interface is mainly based on direct manipulation, using touch gestures that loosely correspond to real-world actions, such as swiping, tapping and pinching, to manipulate on-screen objects, along with a virtual keyboard for text input. In addition to touchscreen devices, Google has further developed Android TV for televisions, Android Auto for cars, and Android Wear for wrist watches, each with a specialized user interface. Variants of Android are also used on notebooks, game consoles, digital cameras, and other electronics.

Android has the largest installed base of all operating systems (OS) of any kind.[b] Android has been the best selling OS on tablets since 2013, and on smartphones it is dominant by any metric.

Initially developed by Android, Inc., which Google bought in 2005, Android was unveiled in 2007 along with the founding of the Open Handset Alliance – a consortium of hardware, software, and telecommunication companies devoted to advancing open standards for mobile devices. As of July 2013, the Google Play store has had over one million Android applications ("apps") published– including many "business-class apps" that rival competing mobile platforms– and over 50 billion applications downloaded. An April–May 2013 survey of mobile application developers found that 71% of developers create applications for Android, and a 2015 survey found that 40% of full-time professional developers see Android as their priority target platform, which is comparable to Apple's iOS on 37% with both platforms far above others. In September 2015, Android had 1.4 billion monthly active devices.

Android's source code is released by Google under open source licenses, although most Android devices ultimately ship with a combination of open source and proprietary software, including proprietary software required for accessing Google services. Android is popular with technology companies that require a ready-made, low-cost and customizable operating system for high-tech devices. Its open nature has encouraged a large community of developers and enthusiasts to use the open-source code as a foundation for community-driven projects, which deliver updates to older devices, add new features for advanced users or bring Android to devices originally shipped with other operating systems. The success of Android has made it a target for patent (and copyright) litigation as part of the so-called "smartphone wars" between technology companies.

What is Web Services and APIs?

A **Web service** is a service offered by an electronic device to another electronic device, communicating with each other via the World Wide Web. In a Web service, Web technology such as HTTP, originally designed for human-to-machine communication, is utilized for machine-to-machine communication, more specifically for transferring machine readable file formats such as XML and JSON. In practice, the Web service typically provides an object-oriented Web-based interface to a database server, utilized for example by another Web server, or by a mobile application, that provides a user interface to the end user. Another common application offered to the end user may be a mashup, where a Web server consumes several Web services at different machines, and compiles the content into one user interface.

We can identify two major classes of Web services:

- *REST-compliant Web services* , in which the primary purpose of the service is to manipulate XML representations of Web resources using a uniform set of "stateless" operations; and
- *arbitrary Web services* , in which the service may expose an arbitrary set of operations.
  — *W3C, Web Services Architecture*

The term "Web service" describes a standardized way of integrating Web-based applications using the XML, SOAP, WSDL and UDDI open standards over an Internet protocol backbone. XML is the data format used to contain the data and provide metadata around it, SOAP is used to transfer the data, WSDL is used for describing the services available and UDDI lists what services are available.

A Web service is a method of communication between two electronic devices over a network. It is a software function provided at a network address over the Web with the service *always on* as in the concept of utility computing.

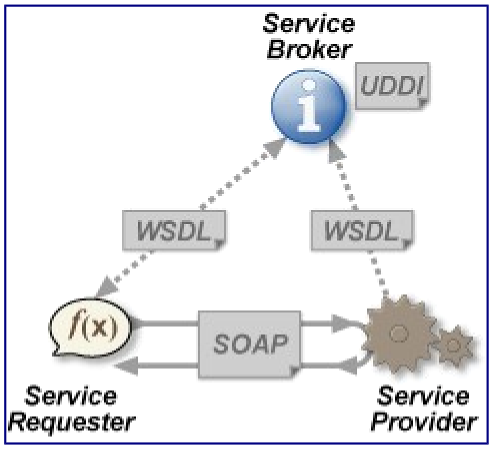
Many organizations use multiple software systems for management. Different software systems often need to exchange data with each other, and a Web service is a method of communication that allows two software systems to exchange this data over the internet. The software system that requests data is called a *service requester*, whereas the software system that would process the request and provide the data is called a *service provider.*

Different software may use different programming languages, and hence there is a need for a method of data exchange that doesn't depend upon a particular programming language. Most types of software can, however, interpret XML tags. Thus, Web services can use XML files for data exchange.

Rules for communication between different systems need to be defined, such as:

- How one system can request data from another system.
- Which specific parameters are needed in the data request
- What would be the structure of the data produced. (Normally, data is exchanged in XML files, and the structure of the XML file is validated against an .xsd file.)
- What error messages to display when a certain rule for communication is not observed, to make troubleshooting easier

All of these rules for communication are defined in a file called WSDL (Web Services Description Language), which has a .wsdl extension. (Proposals for **Autonomous Web Services** (**AWS**) seek to develop more flexible Web services which do not rely on strict rules.)



A directory called UDDI (Universal Description, Discovery and Integration) defines which software system should be contacted for which type of data. So when one software system needs one particular report/data, it would go to the UDDI and find out which other system it can contact for receiving that data. Once the software system finds out which other system it should contact, it would then contact that system using a special protocol called SOAP (Simple Object Access Protocol). The service provider system would first validate the data request by referring to the WSDL file, and then process the request and send the data under the SOAP protocol.

A Web API is a development in Web services where emphasis has been moving to simpler representational state transfer (REST) based communications. RESTful APIs do not require XML-based Web service protocols (SOAP and WSDL) to support their interfaces.

Feasibility Study

Technical Feasibility Study : This project contains cloud implementation and android application environment. Many open source cloud service providers are easily available for prototyping purpose, and Android development environment(Android Studio) is easily available. The project's technical feasibility is easily achievable.

Economical Feasibility Study : This project requires an cloud implementation and android development environment which both are available open source. Therefore, economic feasibility is easily achievable.

Operational Feasibility Study : This project requires computational algorithmic structures and basic connectivity establishment for the prototyping purpose which is easily achievable.

Schedule Feasibility Study : This project requires at most of 2 months for prototype completion.

Methodology

All the required computation will be performed on the cloud services developed in the project itself. Most commonly used language for the project will be JAVA in both android development and cloud services development. Other languages may include XML, groovy in case of android development.
Most part of the android application will include front-end development and user interface for the services provided by the project. All the user data will be stored in database implementations in the cloud interface itself for improving the resource gathering efficiency over the network. The data will be kept secure with authentication tokens for any CRUD application over the data.

There will be simple login/sign-up implementation involving a user-id for uniqueness and password for authentication, with some personal detail involving First Name, Last Name etc. this user-id will also further be used for creating collaborative task association and implementation.

The first most service provided by the application will be of storing and maintaining the tasks input by the associated users. And performing scheduled reminder services for the user regarding those tasks completed. Also a log will be maintained for all the task regarding their completion for measuring the ethic quantity for all the users. Ethics will be denoted in the application as an experience level format more commonly used in game genre of Android Applications to make the application more appealing.

There will also be a profile viewer implementation so as for any user to look for other users performance globally rated over the ethic count and the amount of total accomplished tasks by those users. It will perform in a way of leader-board feature for users to complete to.

Login/Sign-up process:

In this module, new users will sign up for the application and current users will login using there unique keys which will be user-ID and password of standard fashion. For new users, some essential extra data will be mandatory to fill like First Name, Last Name and contact number. After filling the data, application will forward the registration data to the web service where it will verify that there is no overlap of user-ID or contact data for maintaining uniqueness and persistence in data. Also an authentication key will be generated for every user-ID which will be used for further feature extractions in the application.

This same authentication key will be provided back by the web service to the application to authenticate the user in both sign-up and login processes. After which all the features of the web service will be called using that authentication key for maintaining user verification throughout the session in the application realm.

Task List Fetch Process:

This module can only be called after authentication is completed, otherwise the process will lead back to login process. In this, any user can fetch all the task lists that user have created within this application using the authentication key which will be managed by the application itself.

Application will ask for all task list that current active user has created from the web service using W3C machine to machine interaction of service. Web Service will gather all the task related to this particular user after it authenticates the user and then it will create a JSON bundle of all the data. This JSON bundle will be send back to application as a response body for the fetch request, which is easily readable by the application. Application will read this response and fill the interface with available task lists from the service.

Creating / Editing Task Lists:

These two actions can be easily implemented in similar process and for that reason we are merging them to a single process module. Here any task list generated by the relative user can be edited or any new task list can be generated by the user. Within application scope, if user focus on a certain task list, it will enter a task list editing panel, or if user clicks on new task list button the same panel will be opened but without any data, which is, blank task list.

In case of editing scenario, the task list will also be provided with the respective task id, but for new task list no task id is available so the task id variable will be empty for in negative fashion (-1).

When user will click on save button, all the task list data with task id variable will be send to web Service as a request body. Service will check for task ID, if it is not available (-1) it will generate a new task linked to the respective user-ID and response in data created status code. Otherwise with available task-ID, service will edit the content of that task list and response in data edited status code.

Data inside a task list include array of composite unit tasks, respective user-ID, task-ID, scheduled due date(time stamp). And a composite unit task include task content and Boolean for whether it is completed or not.

Leader-Board Process:

This module heavily relies on the work ethic calculation. For every user, this ethic calculation will be updated when user uploaded any data to the service. So as to maintain a fresh ethic value for all users. In application, a user will click on the leader-board button in the home activity of the application to access this activity. The application will request for the leader-board process from the web service. What service will be doing is arranging the list of users according to the descending order of ethic count value, creating an ethic leader-board list of users. This list will be forwarded back to the application n response body in format JSON, which will then be read by the application to fill a leader-board activity. This activity will show all the leading users according to ethic count and will uniquely also tell the current user’s rank in this leaderboard.

Deleting user / task-list process:

All remove process will be handled by similar web service, where application will provide the task id or user id to the service in request uri which is needed to be deleted, and all respective data will be removed from the database. In case of task-list, the task-list of provided task-id will be removed form the task list table from database, but in case of user-id, not just user will be removed from user table, but all the task respective to that user will also be deleted from the task list table of the database.

Facilities Required for Proposed Work

- Most of the facilities required for the accomplishment of this project is software based. Including Android Studio for android development environment and eclipse j2ee for cloud services development which can easily be gathered due to open source nature of these products.

- A Cloud Interface provider is required which can be provided for prototyping purposes by OpenShift Cloud Interface Providers by RedHat.

- And an efficient database implementation like MySQL for integrating to the cloud service and storing all the data logs for the project methodology.

- Also to upload this project to a work-space we need github, Github is an open source application/project uploading environment which help developers to share there projects easily within the team and also handle version control and software life-cycle efficiently. It also helps to track previous changes in a program and address certain issues in good manner.

Technologies Used

For implementing this project, many latest technologies where used by us, most of which are explained below, with purpose soled by these technologies, and some information regarding those technologies.

Environments:

Android Studio – Android studio is and IDE developed by google itself for helping android developers in producing quality android applications. We used this environment for the same purpose as it is very helpful in development of those specific applications.

Download link: https://developer.android.com/studio/index.html

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

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

Project Structure:

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

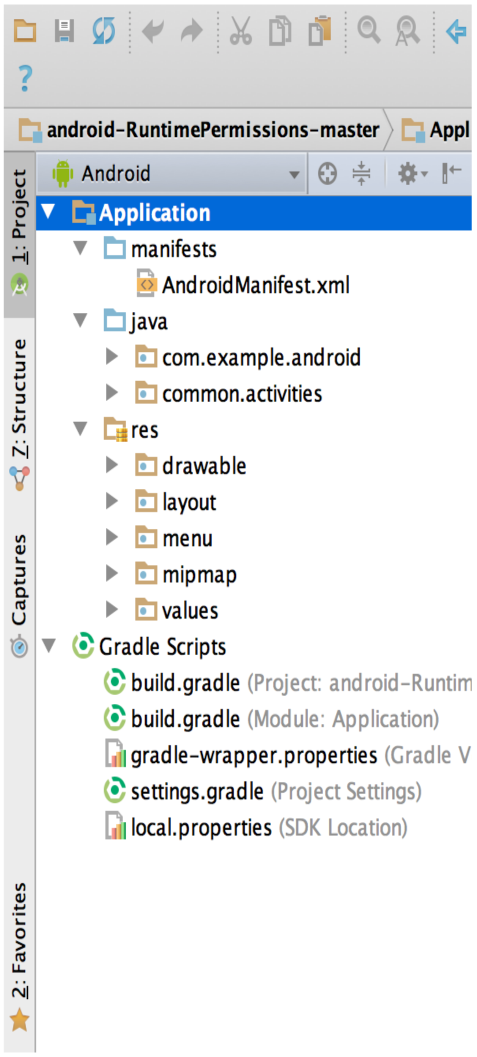
- Android app modules
- Library modules
- Google App Engine modules

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

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

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

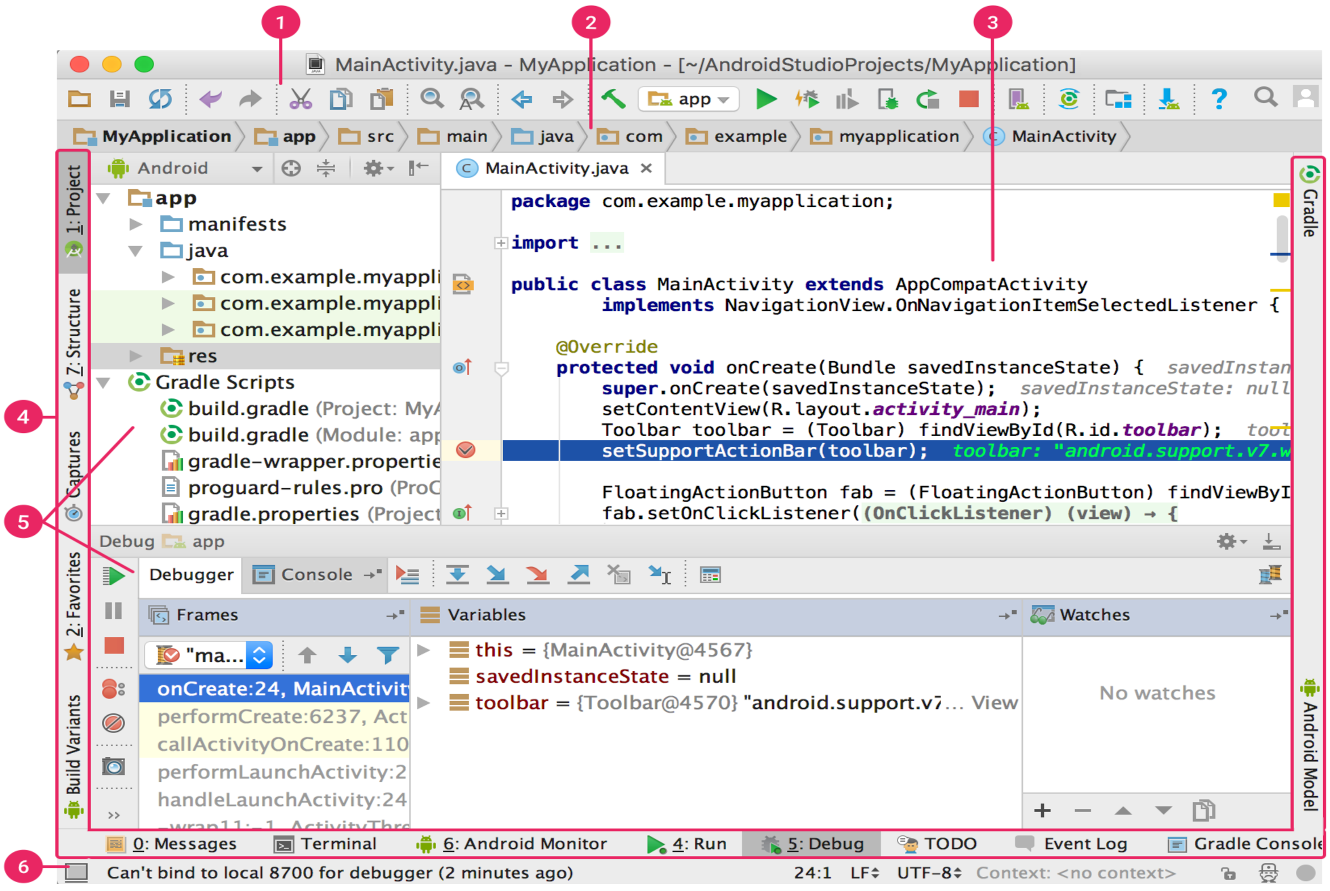
The Android project structure on disk differs from this flattened representation. To see the actual file structure of the project, select **Project** from the **Project** dropdown .



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

The User Interface:

The Android Studio main window is made up of several logical areas identified in figure below.



```
package com.example.myapplication;

import ...

public class MainActivity extends AppCompatActivity
        implements NavigationView.OnNavigationItemSelectedListener {

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

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

```
"ma...
onCreate:24, MainActivit
performCreate:6237, Act
callActivityOnCreate:110
performLaunchActivity:2
handleLaunchActivity:24
```

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

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

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

STS (Spring Tool Suite) – STS is an eclipse based J2EE IDE used for specific spring framework usage, and as we are using spring to create RestControllers in our web Services, so it is evident that STS is a better match for web service IDE.

Download link: https://spring.io/tools/sts

Spring Tool Suite™

The Spring Tool Suite is an Eclipse-based development environment that is customized for developing Spring applications. It provides a ready-to-use environment to implement, debug, run, and deploy your Spring applications, including integrations for Pivotal tc Server, Pivotal Cloud Foundry, Git, Maven, AspectJ, and comes on top of the latest Eclipse releases.

Included with the Spring Tool Suite is the developer edition of Pivotal tc Server, the drop-in replacement for Apache Tomcat that's optimized for Spring. With its Spring Insight console, tc Server Developer Edition provides a graphical real-time view of application performance metrics that lets developers identify and diagnose problems from their desktops.

The Spring Tool suite supports application targeting to local, virtual and cloud-based servers. It is freely available for development and internal business operations use with no time limits, fully open-source and licensed under the terms of the Eclipse Public License.

Feature Highlights -

Understands your Spring App:

The Spring Tool Suite understands your Spring projects. It parses your configuration files and displays detailed information about the beans that are being defined, their dependencies among each other, used namespaces, and extracts overviews for certain stereotypes like request controllers, aspects, services, and more.

Comprehensive Validations for your Spring Configuration:

Because the Spring Tool Suite understands your Spring projects, it provides a comprehensive set of validations that are being applied automatically. Those validations indicate errors in your configurations directly within the IDE, long before you actually run the app. Finding problems and misconfigurations gets a lot easier.

Refactoring Support for your Spring App:

Refactoring support is one of the most important parts of todays software engineering. Therefore the Spring Tool Suite provides advanced support for refactoring Spring applications. Not only the well-known Java refactorings are reflected in your Spring config files, the IDE adds new refactorings for Spring elements (like renaming of Spring beans, for example).

Code Assists All Over the Place:

It doesn't matter whether you are writing Spring XML configuration files or implement JavaConfig Spring apps, whether you are using the core Spring framework alone or together with all the various additional Spring projects, the Spring Tool Suite provides you with meaningful content-assist all over the place, together with quick-fixes for common errors and problems. You will never program with Spring without those code-assists anymore.

Graphical Viewers and Editors:

Want to get an overview of the bean dependencies in your Spring app? Or wanna visualize and edit Spring Integration, Spring Batch, or Spring Webflow definitions? Check out the graphical editors that come with the Spring Tool Suite, right in your IDE, just one click away from your configuration files.

The Best AOP Support Available

The Spring Tool Suite integrates with the AspectJ language tooling for Eclipse and provides the most comprehensive support for AOP that is available today. Aspects are being recognized, incrementally woven into your system, and visualized directly within the IDE. And see where pointcuts match immediately after saving a file.

OpenShift – OpenShift by RedHat is a open source cloud repository used for uploading any cloud application and maintaining its source code. We used it for same purpose for uploading a basic J2EE cloud application and hosting our web service using openshift. It provides with a unique uri for W3C interactions within the service and our application.

Link: https://developers.openshift.com/

Github – We used github to host our source code in open source society, which helps us in maintaining the project efficiently and also handles version control and back logs without any help. Openshift saves its cloud application source code in github hosting only which makes it a certain necessity.

Link: https://github.com/

GitHub is a web-based Git repository hosting service. It offers all of the distributed version control and source code management (SCM) functionality of Git as well as adding its own features. It provides access control and several collaboration features such as bug tracking, feature requests, task management, and wikis for every project.

GitHub offers both plans for private repositories, and free accounts which are commonly used to host open-source software projects. As of April 2016, GitHub reports having more than 14 million users and more than 35 million repositories, making it the largest host of source code in the world.

GitHub is mostly used for code.

In addition to source code, GitHub supports the following formats and features:

- Documentation, including automatically-rendered README files in a variety of Markdown-like file formats (see README files on GitHub)
- Issue tracking (including feature requests) with labels, milestones, assignees and a search engine.
- Wikis
- Pull requests with code review and comments.
- Commits history.
- Graphs: pulse, contributors, commits, code frequency, punch card, network, members.
- Integrations Directory
- Unified and split diffs.
- Email notifications.

Languages/Libraries:

JAVA 1.8 – JAVA is commonly used in this project for both web service and android application programming. Which makes it an essential part of this project.

XML – XML is used in android application for creating UI layouts in android. It is also used in android for maintaining static entries within the application such as primary or secondary color used by the application, basic static strings used in the applications. App theme or animation transactions.

SQLite – SQLite is the SQL version of android, used to maintain a database inside the android device in purpose of your application. We use this technology for maintaining some of our application data even if the application is called without network availability.

Spring Framework – It is used in this application for developing web services using RESTcontrollers which is easily applicable by the spring framework libraries. It is widely used now-a-days for similar purpose and also can be used to host the application all by itself on a spring boot server, but we needed to host it on openshift as a cloud application so we didn't opted that feature.

The Spring Framework is a Java platform that provides comprehensive infrastructure support for developing Java applications. Spring handles the infrastructure so you can focus on your application.

Spring enables you to build applications from "plain old Java objects" (POJOs) and to apply enterprise services non-invasively to POJOs. This capability applies to the Java SE programming model and to full and partial Java EE.

Examples of how you, as an application developer, can benefit from the Spring platform:

- Make a Java method execute in a database transaction without having to deal with transaction APIs.
- Make a local Java method a remote procedure without having to deal with remote APIs.
- Make a local Java method a management operation without having to deal with JMX APIs.
- Make a local Java method a message handler without having to deal with JMS APIs.

Hibernate ORM – Hibernate is a java library used to handle java persistence interactions between service program and database. It provides a neat implementation of simultaneous transaction handling and data persistence control without any actual SQL command to be written by the developer, increasing the productivity of the team.

Hibernate ORM (Hibernate in short) is an object-relational mapping framework for the Java language. It provides a framework for mapping an object-oriented domain model to a relational database. Hibernate solves object-relational impedance mismatch problems by replacing direct, persistent database accesses with high-level object handling functions.

Hibernate is free software that is distributed under the GNU Lesser General Public License 2.1.

Hibernate's primary feature is mapping from Java classes to database tables; and mapping from Java data types to SQL data types. Hibernate also provides data query and retrieval facilities. It generates SQL calls and relieves the developer from manual handling and object conversion of the result set.

Pivotal Virtual local-host server – as a web application is deployed on a server, rather than using a real server that could be very expensive, I used a virtual local-host generated server for deploying the application.

Pivotal Software, Inc. (Pivotal) is a software company based in Palo Alto, California that provides software and services for the development of custom applications for data and analytics based on cloud computing technology. Pivotal Software is a spin-out and joint venture of EMC Corporation and its subsidiary VMware that combined software products, employees, and lines of businesses from the two parent companies including Greenplum, Cloud Foundry, Spring, Pivotal Labs, GemFire and other products from the VMware vFabric Suite.

In July 2012 a GigaOM blog entry speculated on a possible spin out of VMware and EMC that would consolidate some of their cloud computing projects into a new division. The companies confirmed speculation in December of the same year, announcing the initiative with existing technology, people and programs from both companies focused on big data and cloud application platforms under one organization. On April 1, 2013, *The New York Times* reported that Pivotal was official and positioned as a competitor to Amazon Web Services.

At an official event held on April 24, 2013, the organization announced both a $105 million investment from General Electric and its PaaS offering, PivotalCF, a cloud-enabled application platform for private cloud initiatives and public cloud providers.

Oracle Database - (commonly referred to as **Oracle RDBMS** or simply as **Oracle**) It is abbrevated as **Oak Ridge Automatic Computer and Logical Engine** and It is an object-relational database management system produced and marketed by Oracle Corporation.
Larry Ellison and his two friends and former co-workers, Bob Miner and Ed Oates, started a consultancy called Software Development Laboratories (SDL) in 1977. SDL developed the original version of the Oracle software. The name *Oracle* comes from the code-name of a CIA-funded project Ellison had worked on while previously employed by Ampex.

An Oracle database system—identified by an alphanumeric system identifier or SID—comprises at least one instance of the application, along with data storage. An instance—identified persistently by an instantiation number (or activation id: SYS.V_$DATABASE.ACTIVATION#)—comprises a set of operating-system processes and memory-structures that interact with the storage. (Typical processes include PMON (the process monitor) and SMON (the system monitor).) Oracle documentation can refer to an active database instance as a "shared memory realm".

Users of Oracle databases refer to the server-side memory-structure as the SGA (System Global Area). The SGA typically holds cache information such as data-buffers, SQL commands, and user information. In addition to storage, the database consists of online redo logs (or logs), which hold transactional history. Processes can in turn archive the online redo logs into archive logs (offline redo logs), which provide the basis (if necessary) for data recovery and for the physical-standby forms of data replication using Oracle Data Guard.

If the Oracle database administrator has implemented Oracle RAC (Real Application Clusters), then multiple instances, usually on different servers, attach to a central storage array. This scenario offers advantages such as better performance, scalability and redundancy. However, support becomes more complex, and many sites do not use RAC. In version 10*g*, grid computing introduced shared resources where an instance can use (for example) CPU resources from another node (computer) in the grid.

The Oracle DBMS can store and execute stored procedures and functions within itself. PL/SQL (Oracle Corporation's proprietary procedural extension to SQL), or the object-oriented language Java can invoke such code objects and/or provide the programming structures for writing them.

The Oracle RDBMS stores data logically in the form of tablespaces and physically in the form of data files ("datafiles"). Tablespaces can contain various types of memory segments, such as Data Segments, Index Segments, etc. Segments in turn comprise one or more extents. Extents comprise groups of contiguous data blocks. Data blocks form the basic units of data storage.

Maven repository - maven is used for automatically generating a standard project directory structure for a vast amount of varying applications and automate the process of adding dependencies for those projects. As we provide the details for any dependency that we want to be added to the project, maven finds those dependencies online and import them to our projects.

Maven is a build automation tool used primarily for Java projects. The word *maven* means 'accumulator of knowledge' in Yiddish. Maven addresses two aspects of building software: First, it describes how software is built, and second, it describes its dependencies. Contrary to preceding tools like Apache Ant, it uses conventions for the build procedure, and only exceptions need to be written down. An XML file describes the software project being built, its dependencies on other external modules and components, the build order, directories, and required plug-ins.

It comes with pre-defined targets for performing certain well-defined tasks such as compilation of code and its packaging. Maven dynamically downloads Java libraries and Maven plug-ins from one or more repositories such as the Maven 2 Central Repository, and stores them in a local cache.This local cache of downloaded artifacts can also be updated with artifacts created by local projects. Public repositories can also be updated.
Maven can also be used to build and manage projects written in C#, Ruby, Scala, and other languages. The Maven project is hosted by the Apache Software Foundation, where it was formerly part of the Jakarta Project.

Maven is built using a plugin-based architecture that allows it to make use of any application controllable through standard input. Theoretically, this would allow anyone to write plugins to interface with build tools (compilers, unit test tools, etc.) for any other language. In reality, support and use for languages other than Java has been minimal. Currently a plugin for the .NET framework exists and is maintained, and a C/C++ native plugin is maintained for Maven 2.

Alternative technologies like gradle and sbt as build tools do not rely on XML, but keep the key concepts Maven introduced. With Apache Ivy, a dedicated dependency manager was developed as well that also supports Maven repositories.

A central feature in Maven is dependency management. Maven's dependency-handling mechanism is organized around a coordinate system identifying individual artifacts such as software libraries or modules. The POM example above references the JUnit coordinates as a direct dependency of the project. A project that needs, say, the Hibernate library simply has to declare Hibernate's project coordinates in its POM. Maven will automatically download the dependency and the dependencies that Hibernate itself needs (called transitive dependencies) and store them in the user's local repository. Maven 2 Central Repository is used by default to search for libraries, but one can configure the repositories to be used (e.g., company-private repositories) within the POM.

There are search engines such as The Central Repository Search Engine which can be used to find out coordinates for different open-source libraries and frameworks.

Projects developed on a single machine can depend on each other through the local repository. The local repository is a simple folder structure that acts both as a cache for downloaded dependencies and as a centralized storage place for locally built artifacts. The Maven command mvn install builds a project and places its binaries in the local repository. Then other projects can utilize this project by specifying its coordinates in their POMs.

Analysis and Design

In Project Karma, two different systems are designed by us, which can easily work independent from each other, but for the purpose of this project, one system is heavily relied on the other. That system being Android Application and which relies on Web Service system. Here web service system is anyhow independent as it can be used by other applications in further implementations as a back-end API which leads to the problem of application authentication. This problem will be handled in this project in near future as other application attempt to embed this service to their systems.

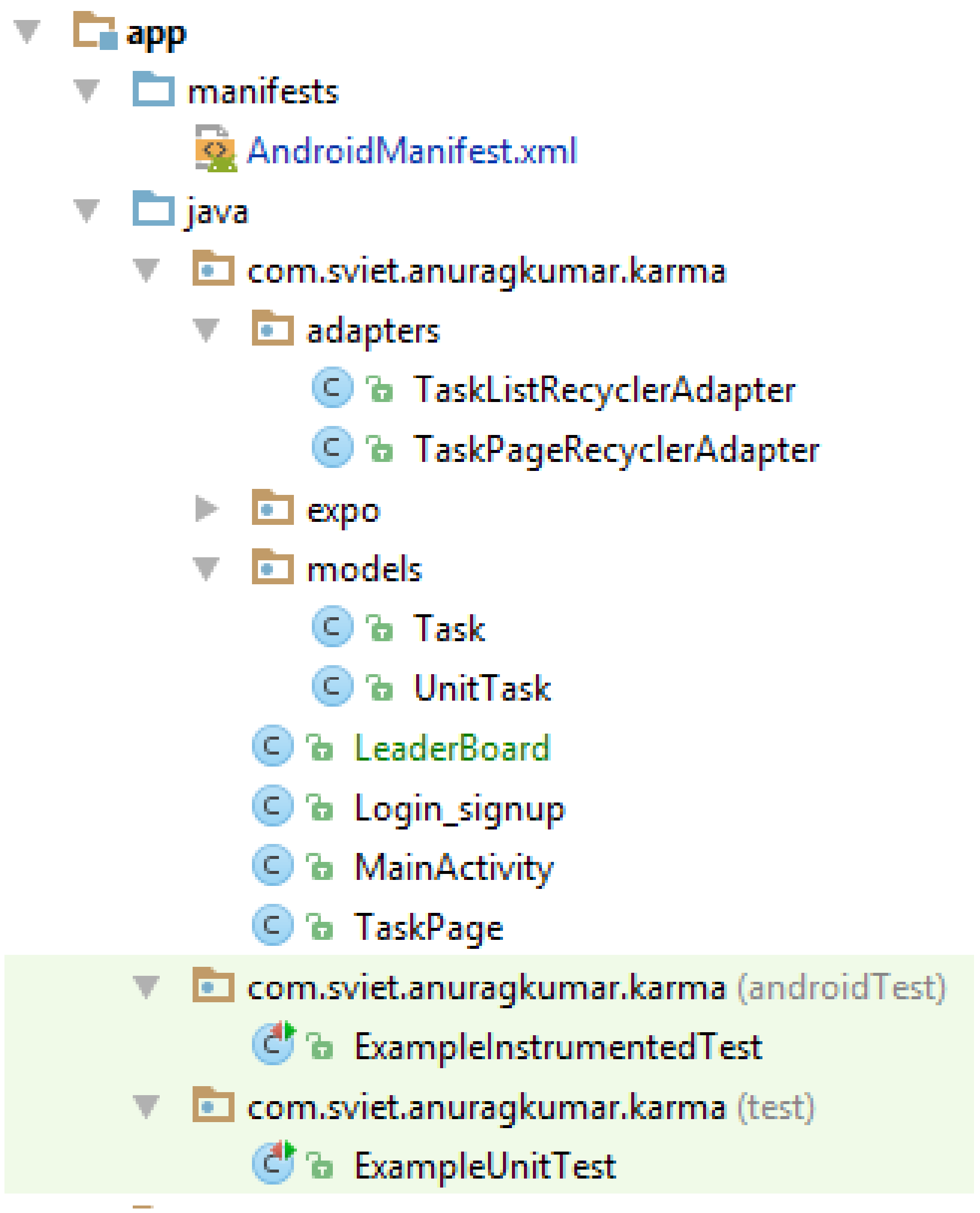
Both of the systems are developed independently, android application in android studio IDE and Web Service using STS toolkit. They are merely connected to each other in uri links. As application tries to create connection to the cloud service, it uses the standard uri format to request a service in our web services, only this term adds them as a connected system otherwise they are very independent from each other.

There is no face or UI for the web service as it is just a machine to machine interaction process including many unit services designed just to create different operational interactions between application and its cloud data, in this case, for task list manipulation, user data altering, task authentication and user authentication and removal of all sorts.

In Android application design, emphasis is given towards material design UI, as it is commonly implemented these days for maintaining design continuity. Most efficient views are implemented including standard RecyclerViews, TextViews, EditTexts, CardViews etc. the above is programed using xml format files for individual layouts.

All background compilations are done in android using java like array generation in ui for fetched data, and dynamic altering in that adaptation of data array. Gathering data from the screen elements and implementing different codes for different scenarios. All web service connectivity is handles asynchronously in this module.

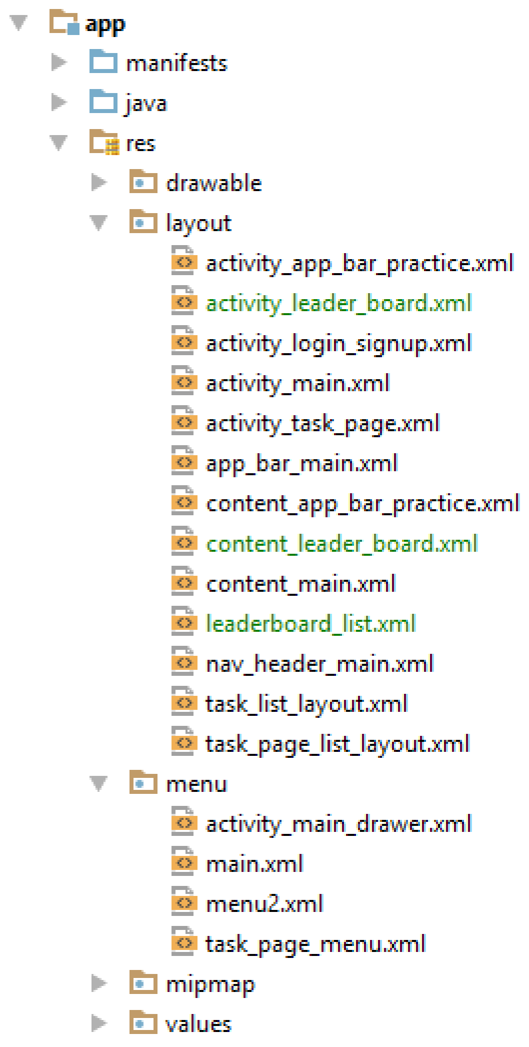
Android Application JAVA and Manifest hierarchical structure:



```
app
  manifests
    AndroidManifest.xml
  java
    com.sviet.anuragkumar.karma
      adapters
        TaskListRecyclerAdapter
        TaskPageRecyclerAdapter
      expo
      models
        Task
        UnitTask
      LeaderBoard
      Login_signup
      MainActivity
      TaskPage
    com.sviet.anuragkumar.karma (androidTest)
      ExampleInstrumentedTest
    com.sviet.anuragkumar.karma (test)
      ExampleUnitTest
```

The java and manifest directory also consist of Unit test classes that are commonly used for modular testing of different parts of the program in customized fashion.

Android Application Resources hierarchical structure:



```
app
  manifests
  java
  res
    drawable
    layout
      activity_app_bar_practice.xml
      activity_leader_board.xml
      activity_login_signup.xml
      activity_main.xml
      activity_task_page.xml
      app_bar_main.xml
      content_app_bar_practice.xml
      content_leader_board.xml
      content_main.xml
      leaderboard_list.xml
      nav_header_main.xml
      task_list_layout.xml
      task_page_list_layout.xml
    menu
      activity_main_drawer.xml
      main.xml
      menu2.xml
      task_page_menu.xml
    mipmap
    values
```

As you can see above, Resource directory of an Android Application consist not just of layouts, but also menu layout for different layouts and values like string, color scheme, animation xml style, app theme etc.

Mentioned below is the **AndroidManifest.xml** content for this project, helping the reader to gain better prespective of how things are arranged in this Applications, or how activities are awakened and what permissions are available.

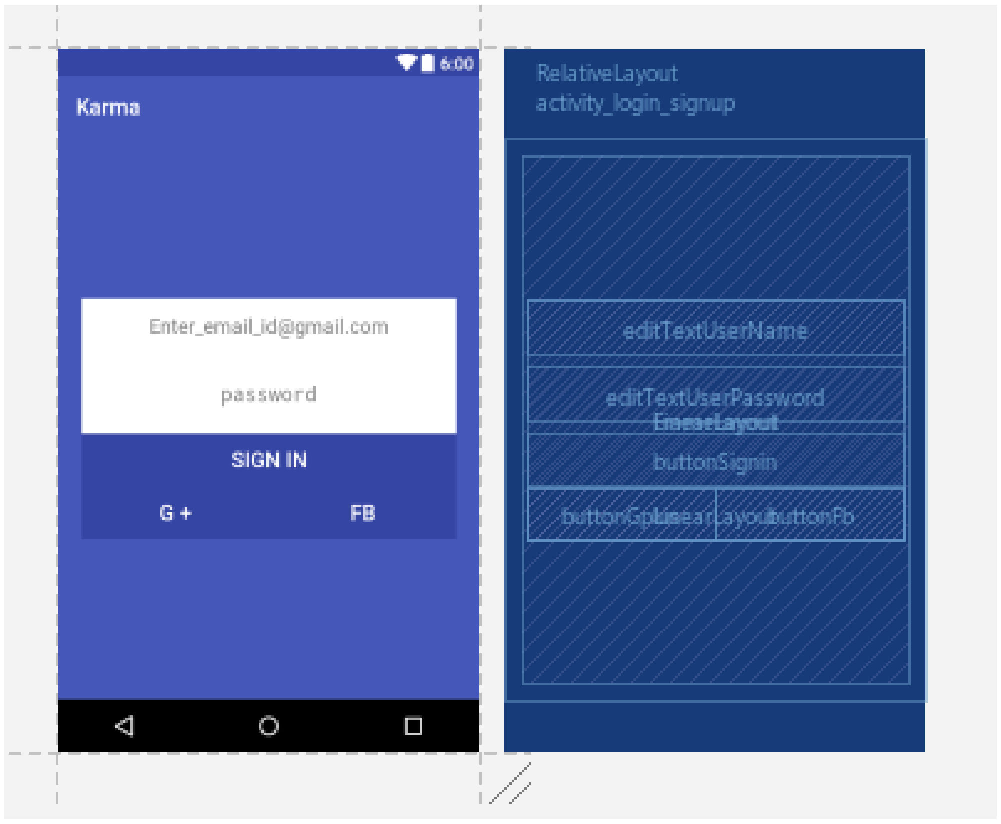
```xml
<?xml version="1.0" encoding="utf-8"?>

<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.sviet.anuragkumar.karma">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">

        <activity
            android:name=".MainActivity"
            android:label="@string/app_name"
            android:theme="@style/AppTheme.NoActionBar">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

        <activity android:name=".Login_signup" />

        <activity
            android:name=".expo.AppBarPractice"
            android:label="@string/title_activity_app_bar_practice"
            android:theme="@style/AppTheme.NoActionBar" />

        <activity android:name=".TaskPage" />

        <activity
            android:name=".LeaderBoard"
            android:label="@string/title_activity_leader_board"
            android:theme="@style/AppTheme.NoActionBar"></activity>
    </application>

</manifest>
```

Below part has the **gradle.build** document of this app, showing what pre-defined libraries are required for this application and what active compilation will be done further for this application. Gradle is a similar tool like Maven, and is used in Android Development for similar purpose as compared to Maven for web Services. Maven can also be used in Android for the Same but Android Studio comes with in-build orientation regarding gradle which is very time efficient and easy to implement.

```
apply plugin: 'com.android.application'
android {
    compileSdkVersion 24
    buildToolsVersion "24.0.3"
    defaultConfig {
        applicationId "com.sviet.anuragkumar.karma"
        minSdkVersion 21
        targetSdkVersion 24
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-
rules.pro'
        }
    }
}
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    androidTestCompile('com.android.support.test.espresso:espresso-core:2.2.2', {
        exclude group: 'com.android.support', module: 'support-annotations'
    })
    compile 'com.android.support:appcompat-v7:24.2.1'
    compile 'com.android.support:design:24.2.1'
    compile 'com.google.code.gson:gson:2.4'
    compile 'com.android.support:cardview-v7:24.2.+'
    compile 'com.android.support:recyclerview-v7:24.2.+'
    testCompile 'junit:junit:4.12'
}
```

Layouts Design & XML codes:

activity_login_signup.xml

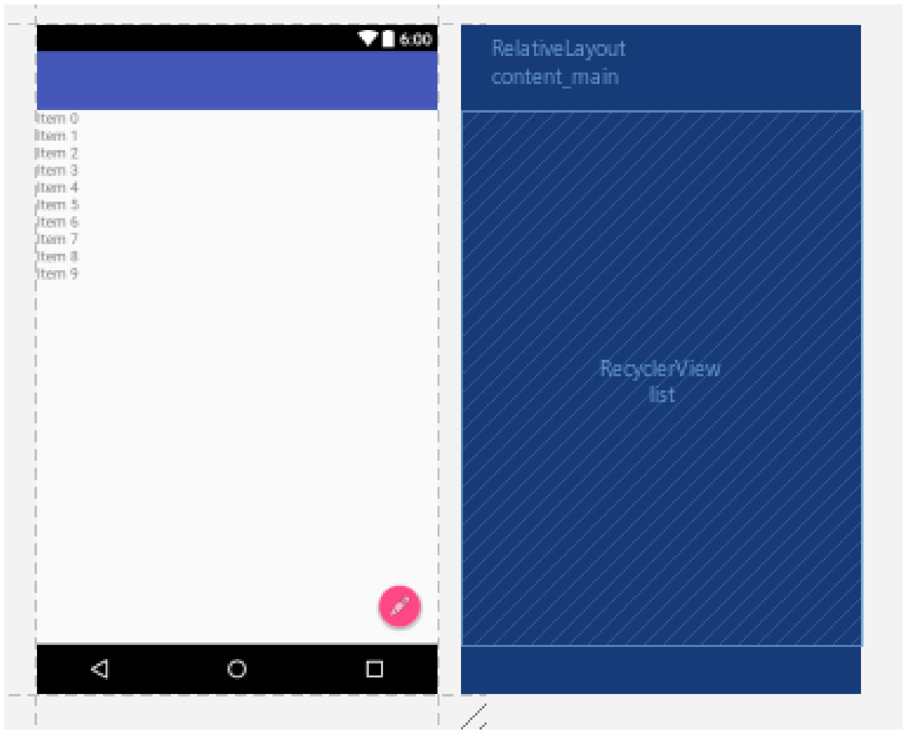


```
<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_login_signup"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:background="@color/colorPrimary"
    tools:context="com.sviet.anuragkumar.karma.Login_signup">
    <FrameLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent">
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
```

```
android:orientation="vertical"
android:background="#ffffff"
android:layout_margin="5dp"
android:layout_gravity="center">

<EditText
     android:layout_width="match_parent"
     android:layout_height="wrap_content"
     android:inputType="textEmailAddress"
     android:id="@+id/editTextUserName"
     android:layout_weight="1"
     android:gravity="center"
     android:padding="15dp"
     android:maxLines="1"
     android:hint="Enter_email_id@gmail.com"/>
<EditText
     android:layout_width="match_parent"
     android:layout_height="wrap_content"
     android:inputType="textPassword"
     android:ems="10"
     android:gravity="center"
     android:padding="15dp"
     android:layout_marginTop="10dp"
     android:id="@+id/editTextUserPassword"
     android:hint="password"
     android:maxLines="1"
     android:layout_weight="1" />


<Button
     android:layout_width="match_parent"
     android:layout_height="wrap_content"
     android:layout_marginTop="10dp"
     android:textColor="#ffff"
     android:textSize="20sp"
     android:id="@+id/buttonSignin"
     android:background="@color/colorPrimaryDark"
     android:text="Sign in" />
<LinearLayout
     android:layout_width="match_parent"
     android:layout_height="wrap_content"
     android:orientation="horizontal">
     <Button
          android:layout_width="match_parent"
          android:layout_height="wrap_content"
          android:textColor="#ffff"
          android:textSize="20sp"
          android:layout_weight="1"
          android:id="@+id/buttonGplus"
          android:background="@color/colorPrimaryDark"
          android:text="G +"/>
     <Button
          android:layout_width="match_parent"
          android:layout_height="wrap_content"
          android:textColor="#ffff"
          android:textSize="20sp"
          android:id="@+id/buttonFb"
          android:layout_weight="1"
          android:background="@color/colorPrimaryDark"
          android:text="Fb"/>
```

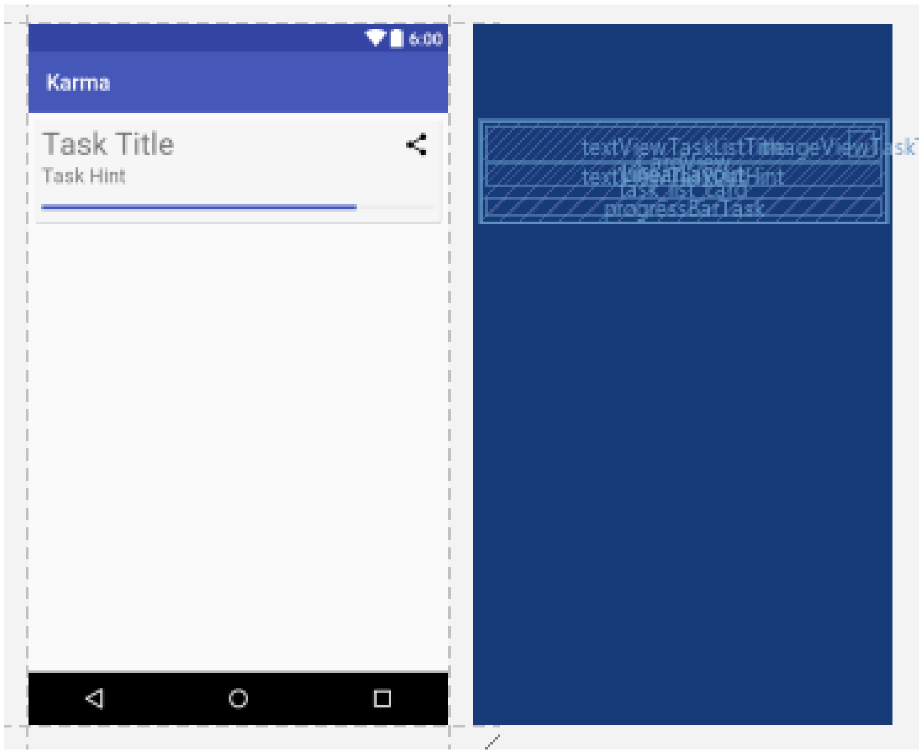
```
                </LinearLayout>
            </LinearLayout>
        </FrameLayout>
</RelativeLayout>
```

content_main.xml



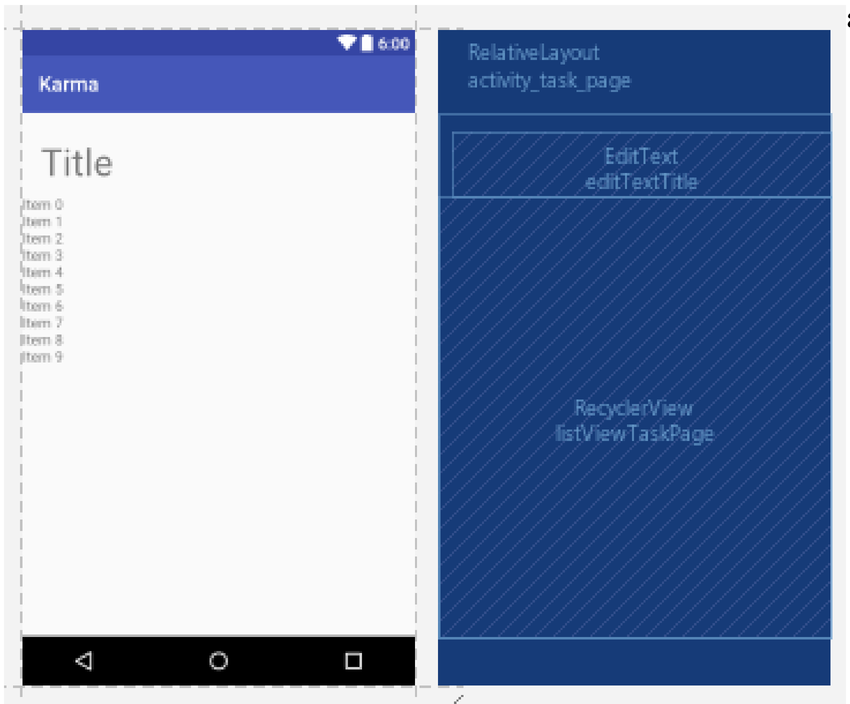
```
<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/content_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    app:layout_behavior="@string/appbar_scrolling_view_behavior"
    tools:context="com.sviet.anuragkumar.karma.MainActivity"
    tools:showIn="@layout/app_bar_main">
    <android.support.v7.widget.RecyclerView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/list">
    </android.support.v7.widget.RecyclerView>
</RelativeLayout>
```

task_list_layout.xml



```
<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
     xmlns:app="http://schemas.android.com/apk/res-auto"
     android:orientation="horizontal" android:layout_width="match_parent"
     android:layout_height="wrap_content"
     android:clickable="true"
     android:focusable="true"
     android:foreground="?android:attr/selectableItemBackground"
     android:background="?android:selectableItemBackground"
     android:layout_margin="5dp">
<android.support.v7.widget.CardView xmlns:card_view="http://schemas.android.com/apk/res-
auto"
     android:layout_width="match_parent"
     android:layout_height="wrap_content"
     android:transitionName="taskCard"
     card_view:cardBackgroundColor="#F5F5F5"
     card_view:cardCornerRadius="1dp"
     android:layout_margin="2dp"
     android:id="@+id/task_list_card"
     card_view:cardElevation="2dp"
     >
          <LinearLayout
               android:layout_width="match_parent"
```

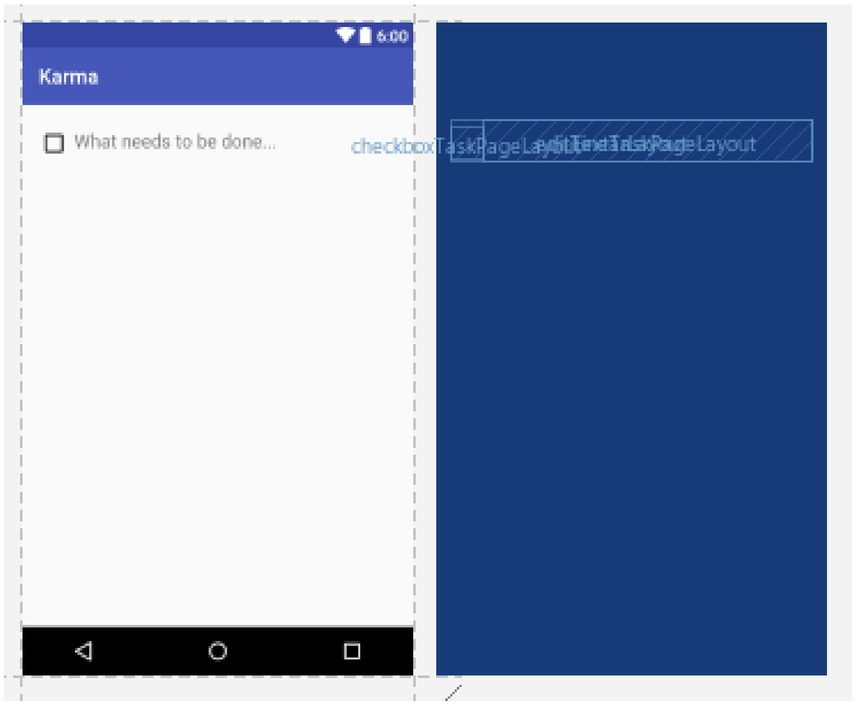
```
                android:layout_height="wrap_content"
                android:padding="5dp"
                android:orientation="vertical">


                <TextView
                    android:text="Task Title"
                    android:transitionName="titleTrans"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:id="@+id/textViewTaskListTitle"
                    android:textSize="28sp" />
                <TextView
                    android:text="Task Hint"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_marginTop="2dp"
                    android:id="@+id/textViewTaskListHint"
                    android:textSize="18sp" />
                <ProgressBar
                    style="?android:attr/progressBarStyleHorizontal"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_marginTop="10dp"
                    android:progressTint="@color/colorPrimary"
                    android:id="@+id/progressBarTask"
                    android:progress="80" />
            </LinearLayout>


        <ImageView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:src="@drawable/ic_menu_share"
            android:layout_gravity="end"
            android:id="@+id/imageViewTaskType"
            android:layout_margin="10dp" />
    </android.support.v7.widget.CardView>
    </LinearLayout>
```



activity_task_page.xml

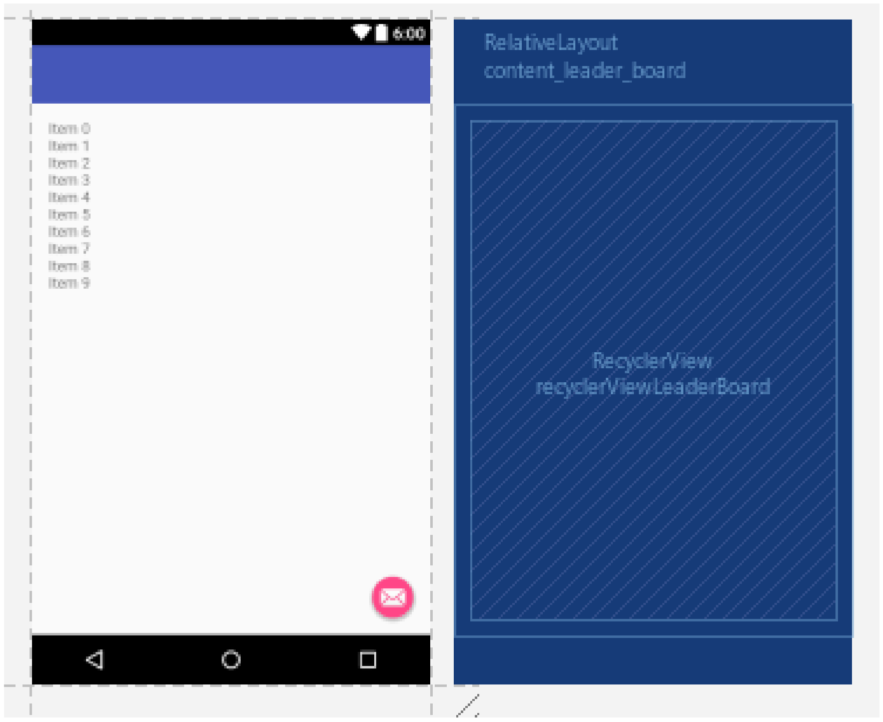
```
<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_task_page"
    android:layout_width="match_parent"
    android:transitionName="taskCard"
    android:layout_height="match_parent"
    tools:context="com.sviet.anuragkumar.karma.TaskPage">
    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:inputType="textCapCharacters"
        android:transitionName="titleTrans"
        android:ems="10"
        android:layout_alignParentStart="true"
        android:layout_marginStart="14dp"
        android:layout_marginTop="18dp"
        android:id="@+id/editTextTitle"
        android:hint="Title"
        android:fontFamily="roboto"
        android:textSize="36sp"
        android:maxLines="1" />
    <android.support.v7.widget.RecyclerView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/listViewTaskPage"
        android:layout_below="@+id/editTextTitle" />
</RelativeLayout>
```

task_page_list_layout.xml



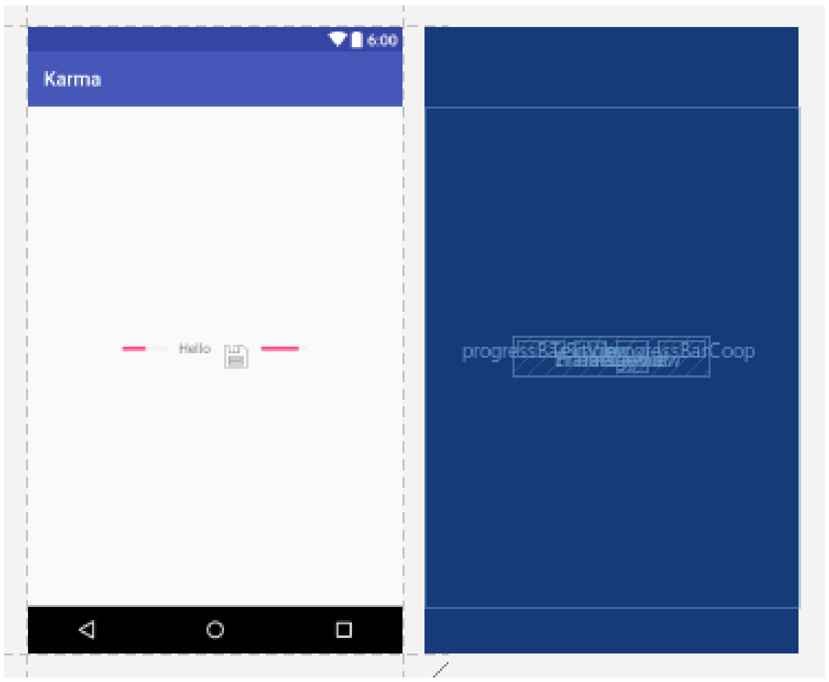
```
<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="horizontal" android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="15dp">
    <CheckBox
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/checkboxTaskPageLayout"/>
    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/editTextTaskPageLayout"
        android:hint="What needs to be done..."
        />
</LinearLayout>
```

activity_leader_board.xml



```
<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/content_leader_board"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    app:layout_behavior="@string/appbar_scrolling_view_behavior"
    tools:context="com.sviet.anuragkumar.karma.LeaderBoard"
    tools:showIn="@layout/activity_leader_board">
    <android.support.v7.widget.RecyclerView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/recyclerViewLeaderBoard">
    </android.support.v7.widget.RecyclerView>
</RelativeLayout>
```

leaderboard_list.xml



```
<?xml version="1.0" encoding="utf-8"?>

<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
     android:layout_width="match_parent"
    android:layout_height="match_parent">
    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:orientation="horizontal">
        <ProgressBar
            style="?android:attr/progressBarStyleHorizontal"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:progress="50"
            android:layout_margin="5dp"
            android:id="@+id/progressBarSingle"  />
```

```
         <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:gravity="center"
            android:layout_margin="5dp"
            android:text="Hello"/>
        <ImageView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_margin="5dp"
            android:src="@android:drawable/ic_menu_save"/>
        <ProgressBar
            style="?android:attr/progressBarStyleHorizontal"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:progress="80"
            android:layout_margin="5dp"
            android:id="@+id/progressBarCoop" />
    </LinearLayout>
</FrameLayout>
```

Above mentioned are the major layout components of this application starting from Login_Signup page. Many more resource files are involved in this application for other secondary purposes but they are not significantly relevant to be explained purposely. These layouts are written in xml format which is understandable by now to the reader.

All the java codes are not mandatory to be explained as they follow the respective methodology of those layouts and elements involved in those layouts. The similar name convention help developers to discriminate between java and layout modules easily so as to know which java class constitutes to which layout xml file.

task_list_layout.xml, leaderboard_list.xml and task_page_list_layout.xml are basically component layout files that corresponds to different composite components that can be universally used within the application for different kind of composite data visualization. For example, task_list_layout.xml is used in content_main.xml file to fill the users task list data as a composite component array in the available RecyclerView. RecyclerViews are commonly used to create linear or grid list layouts within a layout dynamically.

Testing for Android Application:

Testing and Debugging are very important phases of any application software life-cycle. It helps in further development ventures and maintenance of the software. We have implemented testing for different parts of this application in modular fashion.

All individual layout and java class links of this application are provided with prototype dummy data sets to showcase there abilities and current functional features in action even without the connectivity from the web service itself. So as to understand how data is flowing through the application and where it has to be prevented or implemented wisely.

For example, main content activity, is provided with dummy task list data sets so it can create a prototype look of the page without actual data flow from the web service. This process helps developer to understand the actual look and feel of the activity and how it has to be scaled for excessive data or under-looked data.

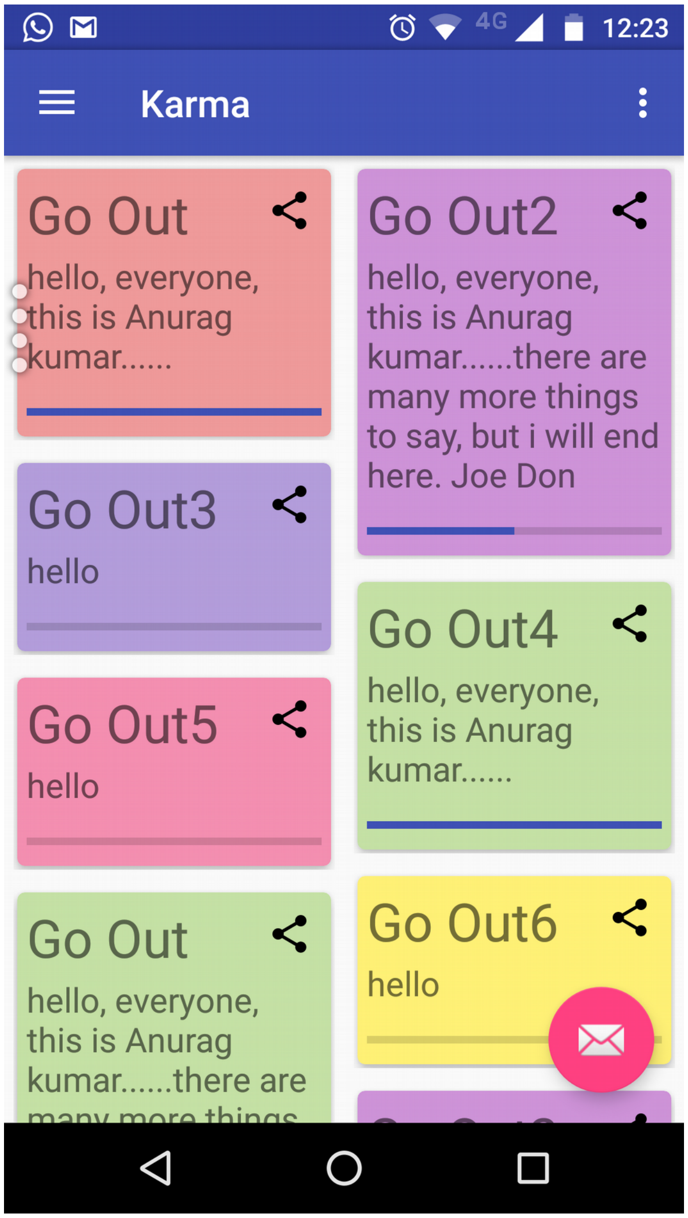
You will further see prototype dummy data provided to all activities in the screen-shot section of the documentation below.

As no ongoing calculus is implemented in the application itself, so no need for modular algorithm testing and debugging on this side of the project. While SQLite content provider is used by the application for better performance during low network or data visibility during network-less sessions, it has to be thoroughly tested before usage. Such test for persistence in SQLite regarding the whole CRUD application of content is done in ExampleUnitTest class of the project under java directory.

Resulting Application:

Undergoing all the content mentioned above in the document, the application structure looks very soothing to the eyes using material design standards and efficient code to handle those transactions. Some early glimpse of the application is provided below as screen-shot by an android device.

All data visible in the below screen-shot are prototyping dummy data sets for documentation purpose only.



Go Out

hello, everyone, this is Anurag kumar......

Go Out2

hello, everyone, this is Anurag kumar......there are many more things to say, but i will end here. Joe Don

Go Out3

hello

Go Out4

hello, everyone, this is Anurag kumar......

Go Out5

hello

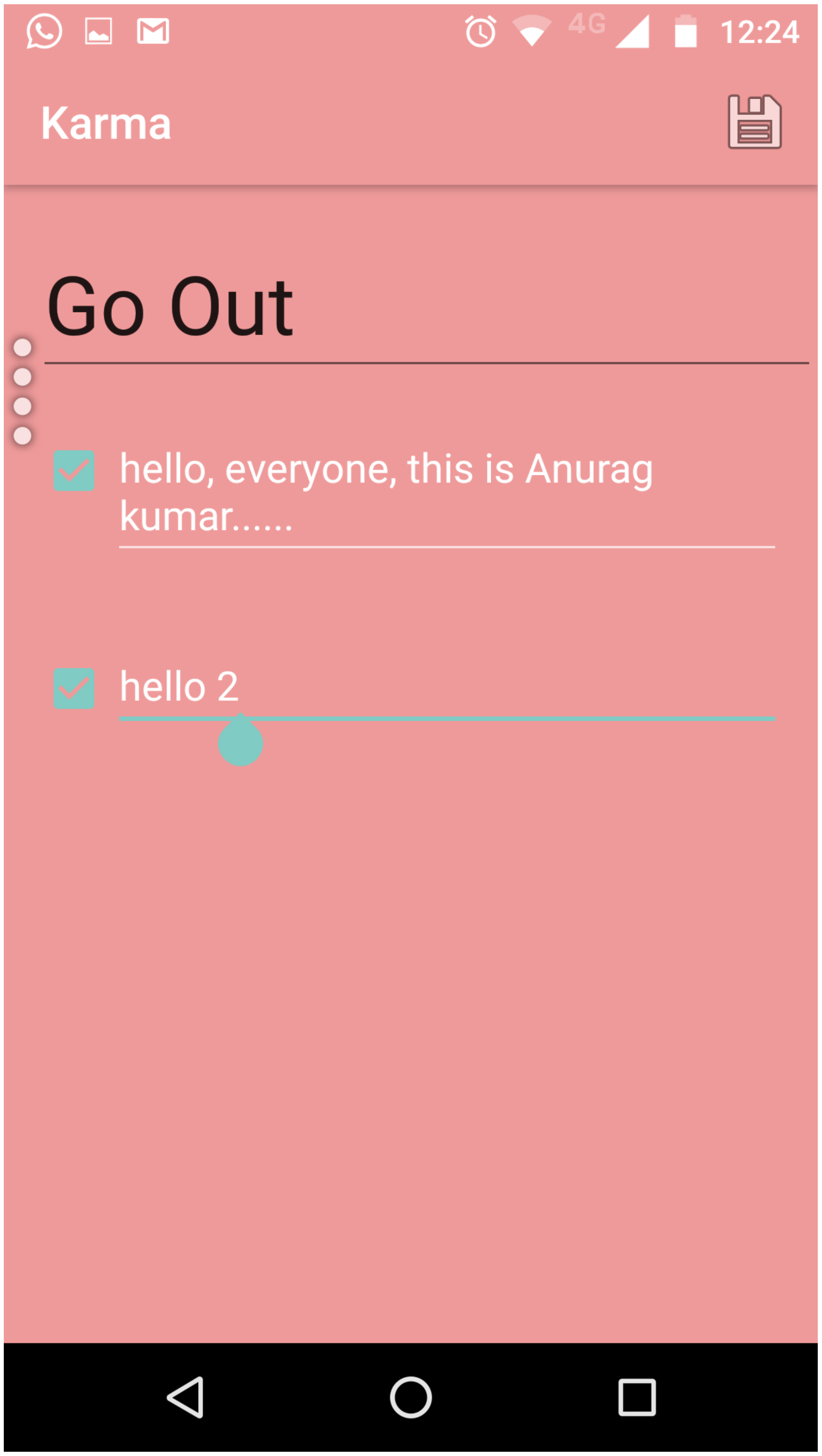
Go Out

hello, everyone, this is Anurag kumar......there are

Go Out6

hello

content_main.xml

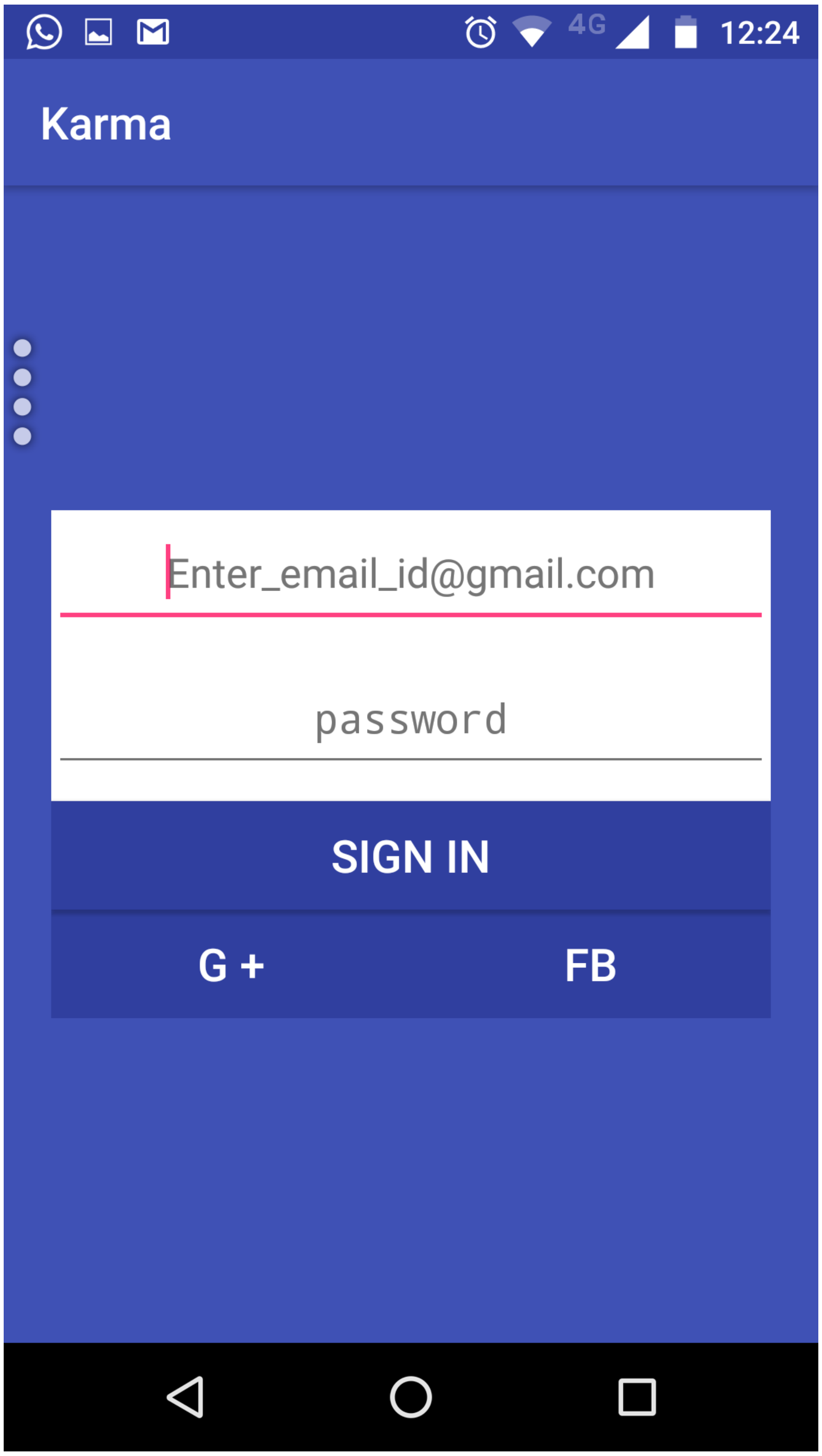


Go Out

hello, everyone, this is Anurag kumar......

hello 2

activity_task_page.xml



activity_login_signup.xml

Further Ventures:

Some of the further feature additions that we are focusing on are mentioned below.

- Co-collaborative task list access and manipulation.
- Better implementation of leader board feature with chat-rooms and co-op work ethic involvement.
- Real-time task list update streams between respective users under same task list.
- Better persistence efficiency overall in the project.
- Better performance under low grade network scenarios with understanding technologies for such purpose.

Bibliography

Developer Examples and Training, developer.android.com,Android,Google.

Case Study and implementation for REST API, various Youtube Channels,Youtube.

Department Faculties for Advise and mentor, CSE Dept., SVIET.

Definitions and explanations for major topics, Wikipedia.

Project Hosting and maintainence, Github.(www.githhub.com)

GitLink for this project hosting:

https://github.com/anuragkumarak95/Karma