1. Introduction

In the Indian society, where money is one of the most valued things, commuting to and back from workplace is one of the most expensive things. After Delhi coming at the top of the list of the most polluted cities in the world and start of the odd-even rule, an application that helps find co travelers was very needed.

A lot of times we use transport means such as auto rickshaws, cabs etc. and since we don't find any co-travelers, we end up paying much more than we actually should have, while all we have to do is to find co-travelers and reduce the cost.

1.1 Why we need a Ride/Cab/Auto Sharing app?

Suppose we find "n" co travelers (n being the number of seats), and the total bill for the journey is X. then on a shared ride that works on a pay per km basis, everyone has to pay only "X/n," which is a significant saving compared to paying X by one person.

Since most people use android phone, thus adding to the accessibility, the app is just a tap away. "**A tap to Save**".

This application helps you track other people nearby who wish to go to nearly the same destination, so that all of them can split the fare and save money.

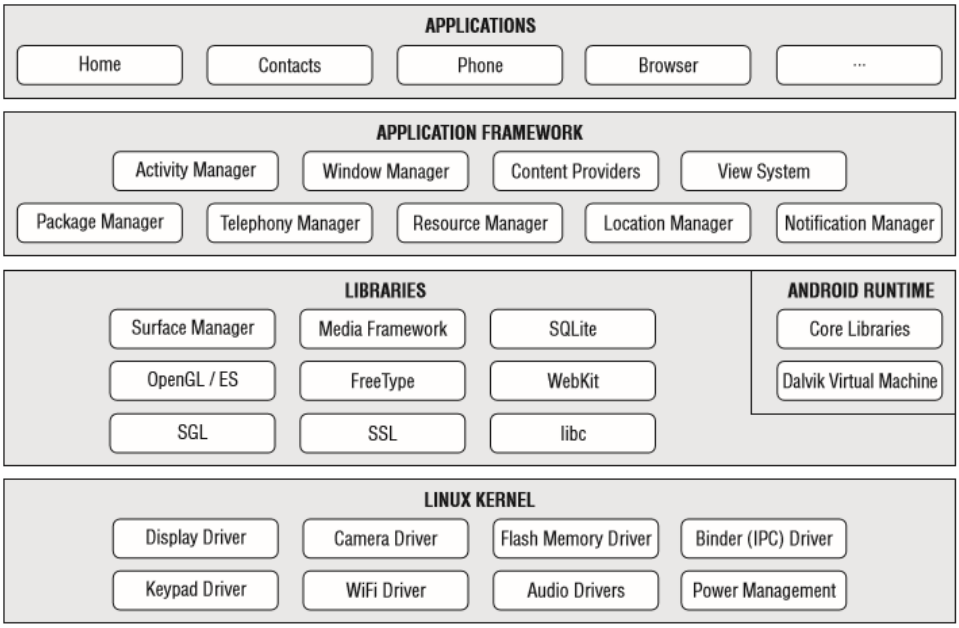
1.2 Why an Android app and not on some other platform?

Android has the largest base of users, thus making an android app means targeting the largest available user base at once. This puts our application in the hands of a big population without the need of much changes. Since this app targets mainly commute, therefore, the application must be available to user whenever they need. That mainly leaves us with only mobile operating systems, and since the number of android users exceed the number of iOS users any given d ay, therefore android is the smart choice.

2. Android

The android operating system is built and developed by Google. The system runs on Linux kernels. Apps are programmed in java and xml is used for layout. Google uses Yaml (or yml) to translate this code into machine language.

The basic structure of an android file system is as below



Applications: These are the system or user applications that run on top of the OS. This is the layer of the operating system that an end user sees.

Application Framework: Android comes with a SDK (Software Development Kit) which provides us tools and API libraries for developing and building applications for the android operating environment using java as the primary programming language. This is called the android application framework.

Libraries: These are stores of source codes and resources which are used by several other android applications and projects. The ADT (Android Development Tools) create a JAR file to compile the contents of these libraries into the project. Libraires help us structure our application code. A lot of Open Source libraries are available for android, therefore, an understanding of the library projects is very important for every programmer.

Android Runtime: Android runtime is an application runtime enviornment used by the android OS. The application's bytecode is translated into native

instructions that are executed by the runtime enviornment of the device. [1]

Linux Kernel: Linux makes the core part of android. Linux is used by android under the hood because it is open source and the developers at google couldhence modify it according to their needs. It also gives them a pre-built and maintained system kernel to start with so they don't have to work from scratch writing their own kernel.

This version can be checked under the kernel header in the about menu in android

The detailed kernel information is given in the image above.

2.1 Evolution of Android

2.1.1 Android 1.0

It is the place where the journey of Android began. It was originally released in the year 2008 and was chiefly centered around giving the clients a whole new experience which noone thought of before. HTC's Dream was the first phone featuring the Android Operating System, it showed the world how capable Android was and what it could do and it was. With onboard hardawre and software supporting camera, Bluetooth, WiFi, organizers, a web browser, Youtube, warnings, wake up timer, texting, media player, touchscreen and the entire google applications cluster and Google Play Store. The primary variant already had satisfied a ton of whta cell phones ought to have the capacity to do by that time. There were just a couple android applications accessible in its exceptionally starting stages and there were just not many developers who considered this OS important.

2.1.2 Android 1.1 (2009)

In 2009, the next upgrade of Android was launched, this was nearly 4 months after the 1.0 v launch. There was no drastic change or big update loaded with new things over the older version. After this version, Android was still a newborn, trying to kick its legs but on the other hand, it showcased the ability of android to give seamless new updates and also the ease by which the end-users could install these updates. At that time, no other smartphone platform could do this, this ability was a big game changer for Android.

2.1.3 Android 1.5 – Cupcake: (2009)

It was with the launch of this version that the Android operating system started to be seen as a stable platform for smartphones. The smart-phone Industry had been established well by this time and therefore this launch was a big step. It wasn't just that it had added a lot of new things that were important and temting to keep the android kicking in the market. It was with this version that they started using candies and eatable assortments for giving names to the newer versions. Cupcake displayed great improvements in the platform like decrease in the time taken by gps to lock position, better camera support both in terms of hardware and software, auto-loading of videos on YouTube, launch of virtual keyboard etc.

2.1.4 Android Donut: 1.6 (2009)

It showed no major update over Cupcake, It made simple refinements all through the stage. It's adaptation may be associated with two reasons, first one being that CDMA backing came to be initially offered in this version, which cleared paths, opening the way for goliath networks and conceivably countless supporters crosswise over Asia. Also, It was with this version that AndroidOS demonstrated it was fit for running on an assortment of angle proportions and screen resolutions, which opened the entryway for telephones that highlighted presentations of various resolutns with this rendition, Android included a modest bunch of new and redesigned highlights like the case for snappy hunt and voice seek, Indicator for battery usage, gathering of camera and exhibition applications, video mode expansion, content to discourse dialects and so on.

2.1.7 Android Gingerbread: 2.4 (2010)

It was about six months after the launch of Froyo that Google thought of a new form named Gingerbread. Contrasted and the past rendition, this was moderately a small launch. In any case, the small changes altogether acquired a genuinely vast change on stage, the progressions brought into the UI components made it right away look cleaner and more fresh. This launch additionally gave a lot of

changes like UI reconsidered for less demanding and speedier performance, Fresh console for quicker content info, selecting content elements, duplicate/glue, Internet calling options. This release of Gingerbread, Google launched yet another new round of the Nexus program, it had chosen Samsung to create the Nexus, which cleared path for the organization's uncontrollably fruitful Galaxy lineup.

2.1.9 Android Ice Cream Sandwich: 4.0 (2011)

Ice Cream Sandwich was, without inquiry, the greatest change for Android on Smartphone till the time. The UI was totally upgraded, and it gave an all new affair for the clients. This variant acquainted the capacity with using delicate catches on the touch screen rather than hard catches for route to places like Home, Back, and Menu. Execution was fundamentally enhanced, and numerous new elements like fresh text style (Roboto), Possibility of clicking of a photo with a grin, including usefulness, for example, overseeing organizers, bookmarks and catching screenshots, Swipe expansion to conceal notices, closed site pages and that's just the beginning, Supporting Wi-Fi Direct, Bluetooth HDP and Android Beam and so on were brought into the stage which made it gigantically well known and increased a great many clients for the Android stage.

2.1.10 Android Jelly Bean: 4.1 (2012)

After IceCreamSandwich, Google concocted the redesign in 2012 naming it as JellyBean. The release planned enhancing the UI and execution to a totally different level, utilizing elements, for example, touch expectation, triple buffering in the design pipelines, upgraded vsync timing, improvements to Android Beam, expansion of Google Now, multichannel sound, Resizable gadgets, USB sound and Updates to the application enhanced and speedier. With this rendition, general client experience had turned out to be exceptionally responsive, smooth and brisk so that the client gets the feeling that the gadget is reacting to inputs progressively.

2.1.11 Android 4.4 – KitKat (2013)

In under a year, Google thought of the upgraded adaptation 4.4 KitKat, named after the well known KitKat chocolate. This version further enhanced execution, by enhancing livelinesss and also with a far and away superior shrewd and down to earth outline. Kitkat accompanied another control board available from warning screen. Miracast was the one of the greatest expansion over previous variant, this component accompanied capacity to remotely stream media and sound from one gadget to other showcase gadget like TV. This was considered as a brief answer from Google to Apple's AirPlay and this increased market footing since now, there were greater nomber of equipment gadgets that supported this element. Kitkat variant joined different augmentations additionally like supporting Bluetooth MAP, new structure for moves in the UI, Support for remote printing, Optimization of memory and touch screen for speedier multitasking and so forth.

2.1.12 Android 5.0 Lollipop (2014)

Lolipop accompanied a modest bunch of newer components, a visuall upgrade plus numerous changes in engine improvements for making assuming quicker, more effective and less heavy on your battery. The configuration, movements, symbol size everything is strong, straightforward, less demanding, and to a great degree smooth which out and out provides a totally new affair to the clients. It accompanied al new lockscreen, warnings, popups, pull-down bar, settings menu and other multitasking choices. Lollipop offered much all the new elements like Device sharing, battery Saver, Smart Lock, ART runtime rather than Dalvik, slick movements for catching taps or symbol determinations and so on.

2.1.13 Android 6.0 Marshmallow (2015)

This focuses mainly on enhancing the overall experience of the users of Lolipop by providing a new and better managed architecture for permissions, new APIs for contextual assistants (a utility marketed by the name "Now On Tap"— which a new ability of the OS searching Google by taking a screenshot of the open app and running all data on the screen through its system and returning useful results on top of the app), a new power management system that reduces background activity when a device is not being used, added software support for hardware

features like fingerprint recognition and USB C type connectors and the ability to move app data and even complete apps to a SD card and use it like internal storage or primary storage, and also a lot of other modifications internally.

2.1.14 Android 7.0 Nougat (2016) (developer preview 4)

Android 7.0 "Nougat" is an up and coming release of the Android working framework. Initially released as an alpha build on March 9, 2016 and last beta release is booked for mid-July. Android 7.0 acquaints eminent changes with the working framework and its advancement stage, including the capacity to show different applications on-screen on the in a split-screen view, support for inline answers to warnings, and an OpenJDK-based Javaenvironment and support for the Vulkan Graphics platform, and "seamless" framework reports on bolstered gadgets.

Android has effectively secured far and it's been an exceptionally fruitful trip till now. A little idea has developed so much enormous that it impacting a huge number of individuals around the world. It is the vision and reinforcement of google that helped AndroidOS to end up the most effective portable working framework like it is currently. From cell phones to tablets and to wearable gadgets. Android made a monster jump in the course of the most recent decade and has become the victor in portable OS's, winning over the iOS from Apple that is paths behind. It will be really energizing for any innovation fan to consider other things Google can invoke in the next AndroidOS releases.

In any case, Android is just an starting, Google is walking ahead. We shall expect Android OS's that deals with Car dashboard, TV's and numerous different gadgets that obviously will, in the future make our lives much less demanding, smoother and gainful.

A little idea by Andy Rubin, is turned into a basic piece of our life.

3. Creating an android Application

Creating anndroid application is a tedious task. Android applications have a lot of advantages over applications of other platforms. We don't need to design the app for every single device or OS version, and when we need to patch some bugs or provide updates, the existing code can be used.

Creating an Android Application is not an easy task. We require a lot of knowledge of JAVA programming and other thigs like using XML for layout, accessing database, SQL, JSON objects. Designing a user friendly UI that users of all age and type can easily access ad navigate through is also very important.

Unlike other android applications, I wont be using a web serverto connect to an online database. The service I'll use here is called Firebase, which is currently owned by google.

Firebase uses a NoSQL database that syncs in real time across devices. It also provides options for storingother data in its cloud such as data and media files. Since I'm using the free plans here, therefore the resources provoded are limited.

Of course my application wont be filled by features and other high end secuity features, but it will do the basic function like accessing database, modifying and deleting content.

Here is what my application will be able to do.

1. Authenticating users.

2. Signing up new users.

3. Signing up or logging in users using their facebook accounts.

4. Posting to the NoSQL database of firebase.

5. Retrieving content from the database.

6. Modifying content in the database.

7. Keeping a log of the content.

8. Managing and monitoring app crashes through the firebase console.

9. Helping users contact the developer.

10. Logging out of the application.

4. Tools and Languages required

4.1 Java is the primary programming language used for coding. All the code is written in java and the JDK. The JDK version used is 1.8. A good working knowledge of java comes in handy while programming android apps as all the code logic and data handling is done in java only. We can design the web backend in any language, but the core coding and the handlingof data sent and received from the server is handled through java code only.

4.2 Android Studio: The program used for coding, compiling and debugging is android studio. It is powered by the IntelliJ platform.

Android studio is the officially suported application by google for all kinds of programming across all android devices such as android TV, android wear, Android auto, Android tablets and smartphones etc.

The layout of the application is designed in XML files, and android provides an easy drag and drop screen that makes it easy for anyone to design the layout without much hassle.

4.3 Firebase: here, I use the firebase platform by google as a backend for storing data, authenticating users and managing the database as admin.

Firebase uses a NoSQL database for storing data and this syncs in realtime across all devices.

The free plan allows upto 50 users to signup and 500 simultaneous connections from devices at a given time, upto 5 GB of storage and upto 1 GB size of website for hosting on their server.

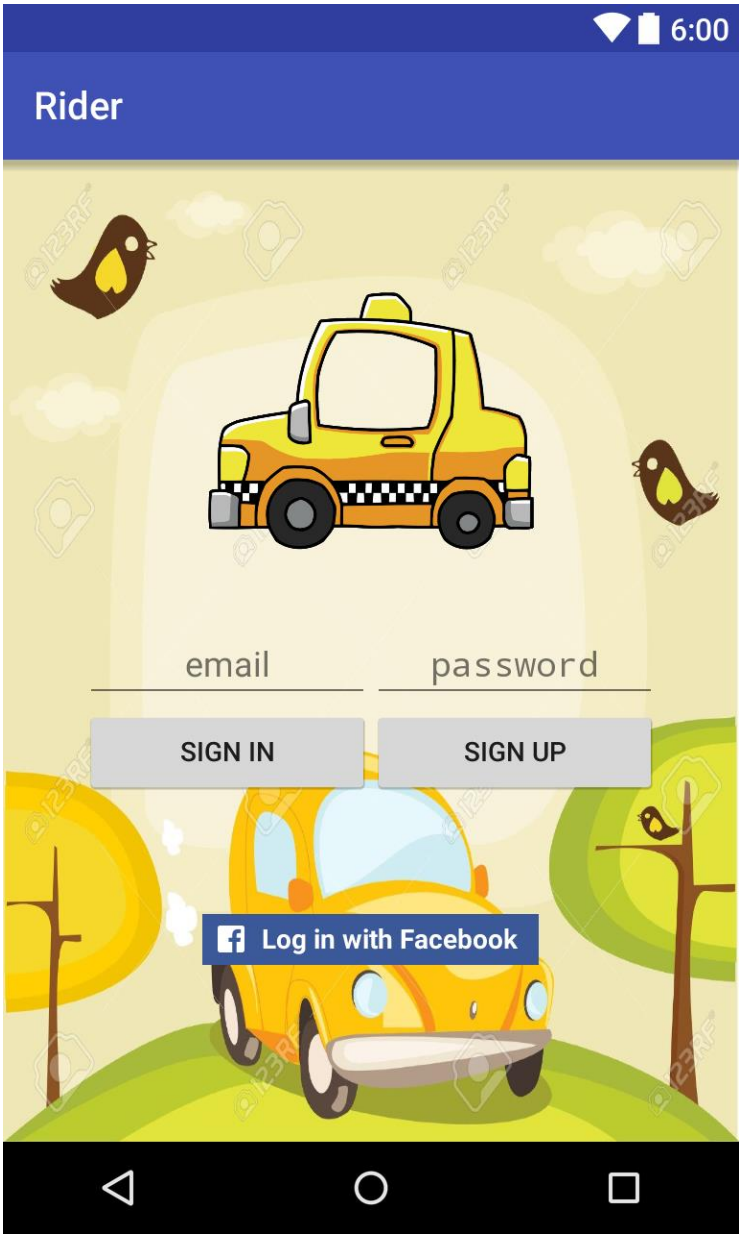
4.4 XML: It stands for Extensible Markup Language, and here in android studio, it is used to design all the layout of the application. However we do not eed to explicitly learn to code in XML as android Studio also provides a deag and drop interface with other easy features and options to design the layout.

5. How The App Works

Log in

The user can login using either email id or password or using Facebook. Once logged in, an auth token is created and that handles all the tasks until the app is closed or the user logs out.

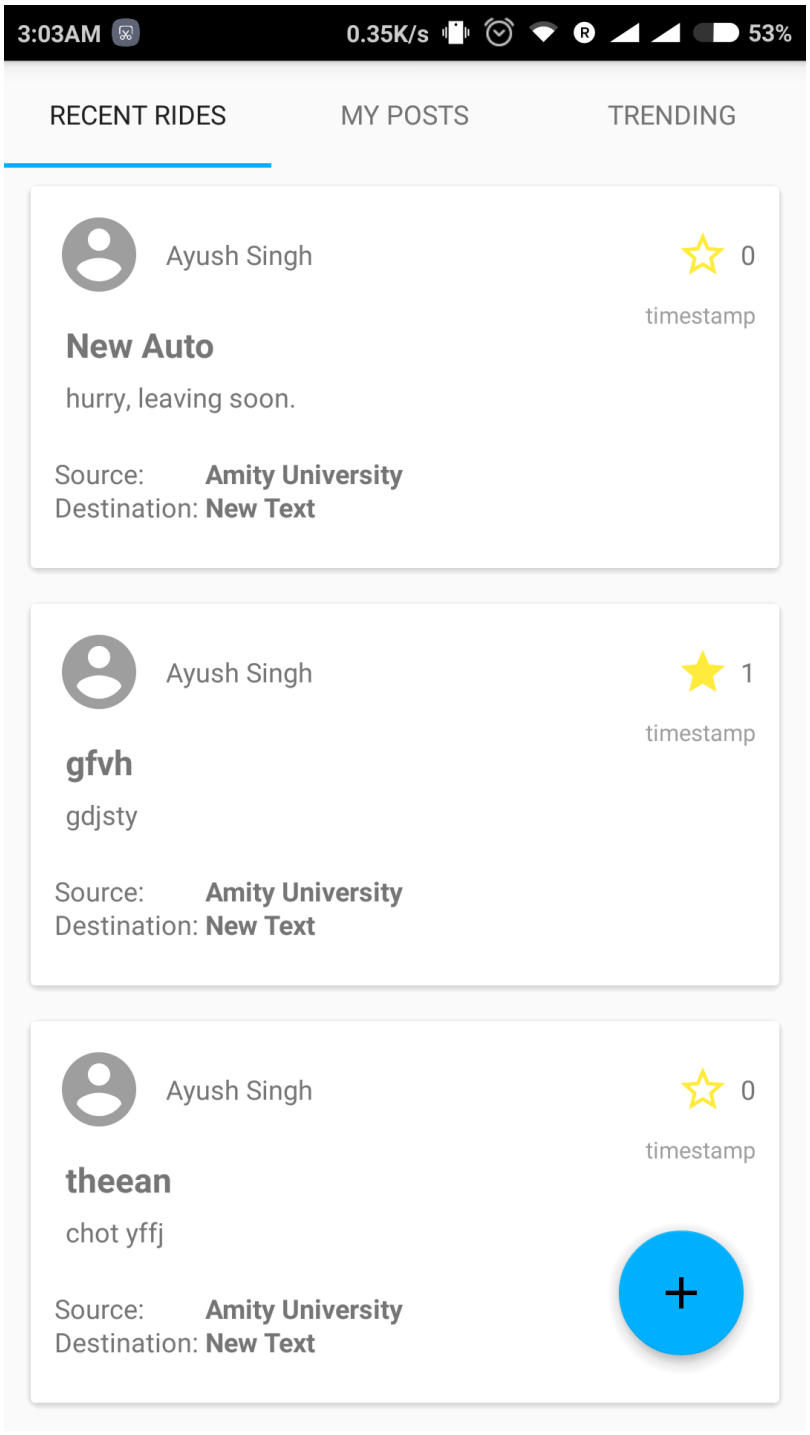
The login screen of the app is as below.

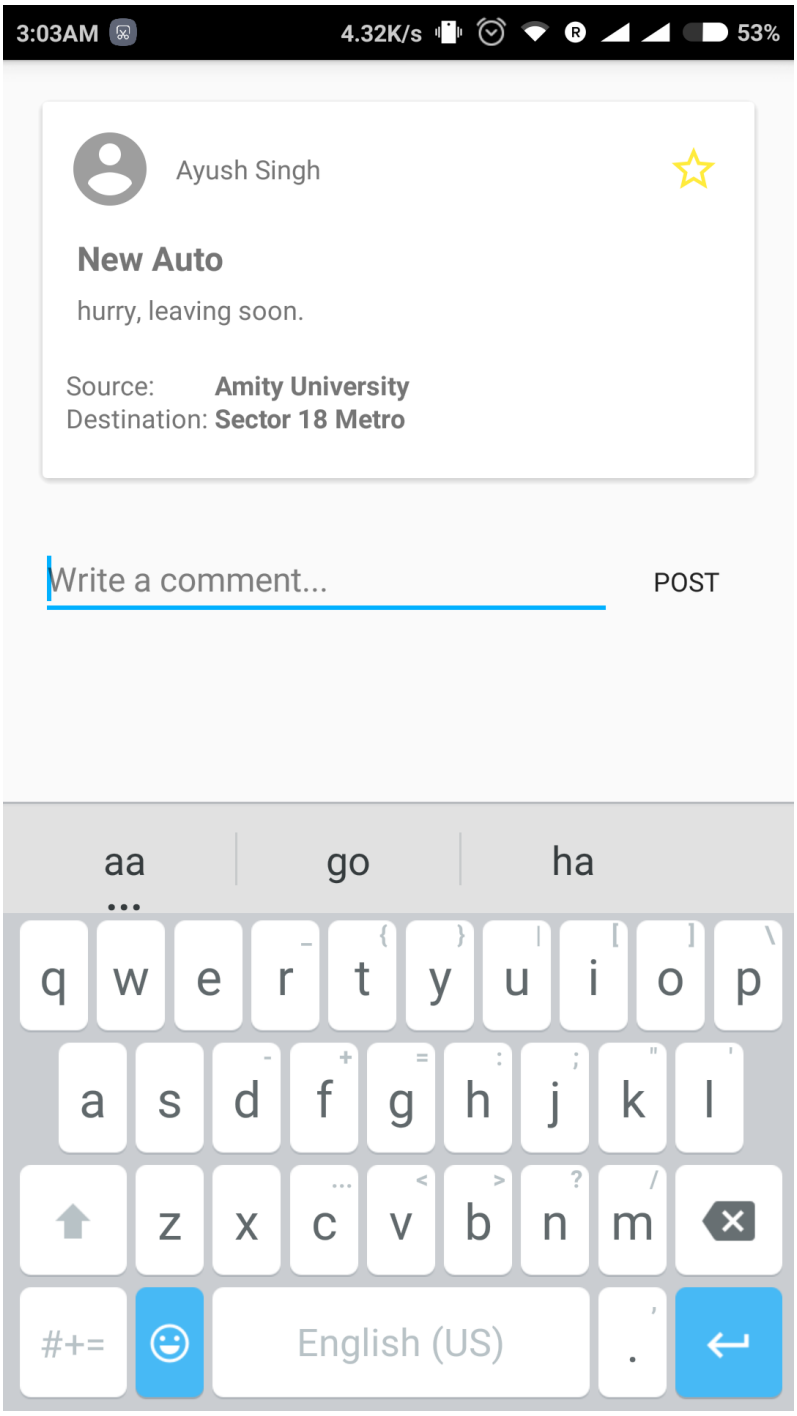


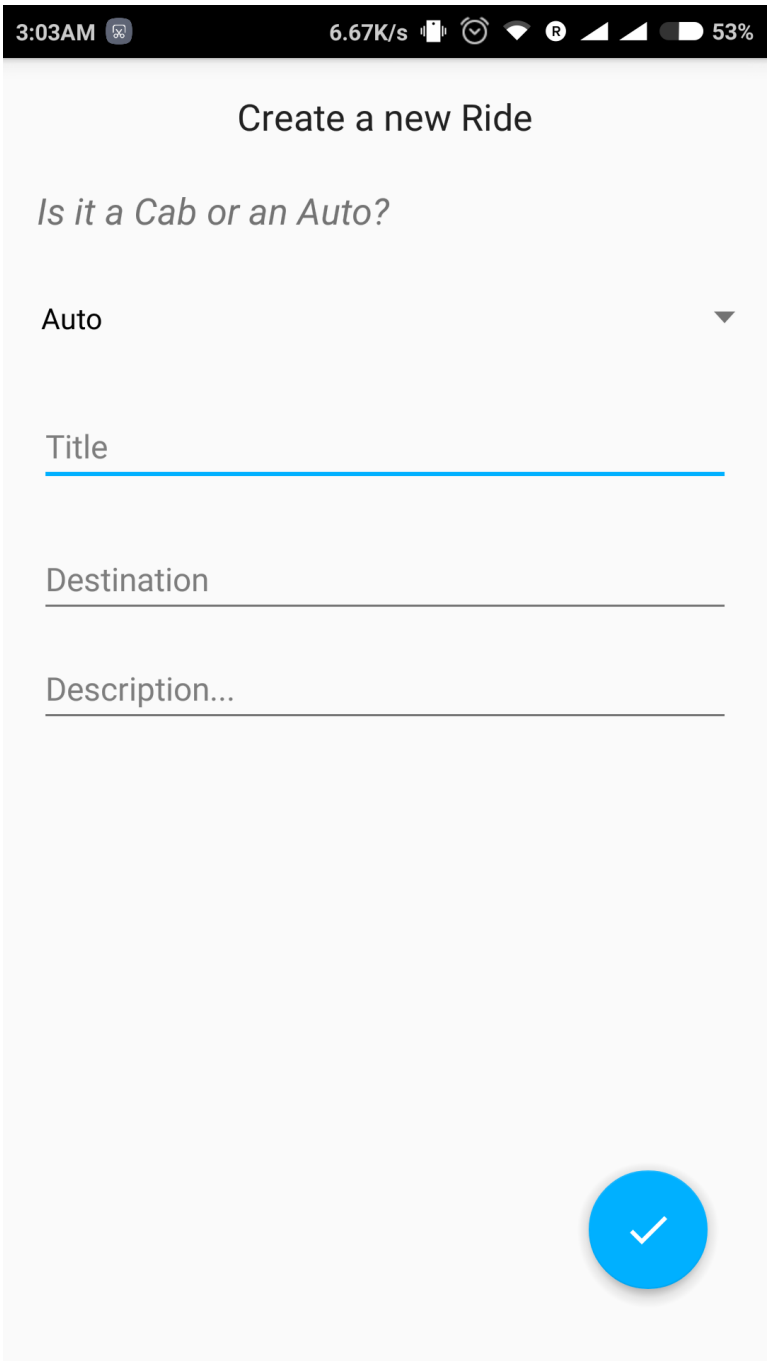
Create a ride: After login, when a user chooses to create a ride, an entry is made into the database for other users that are searching for rides.

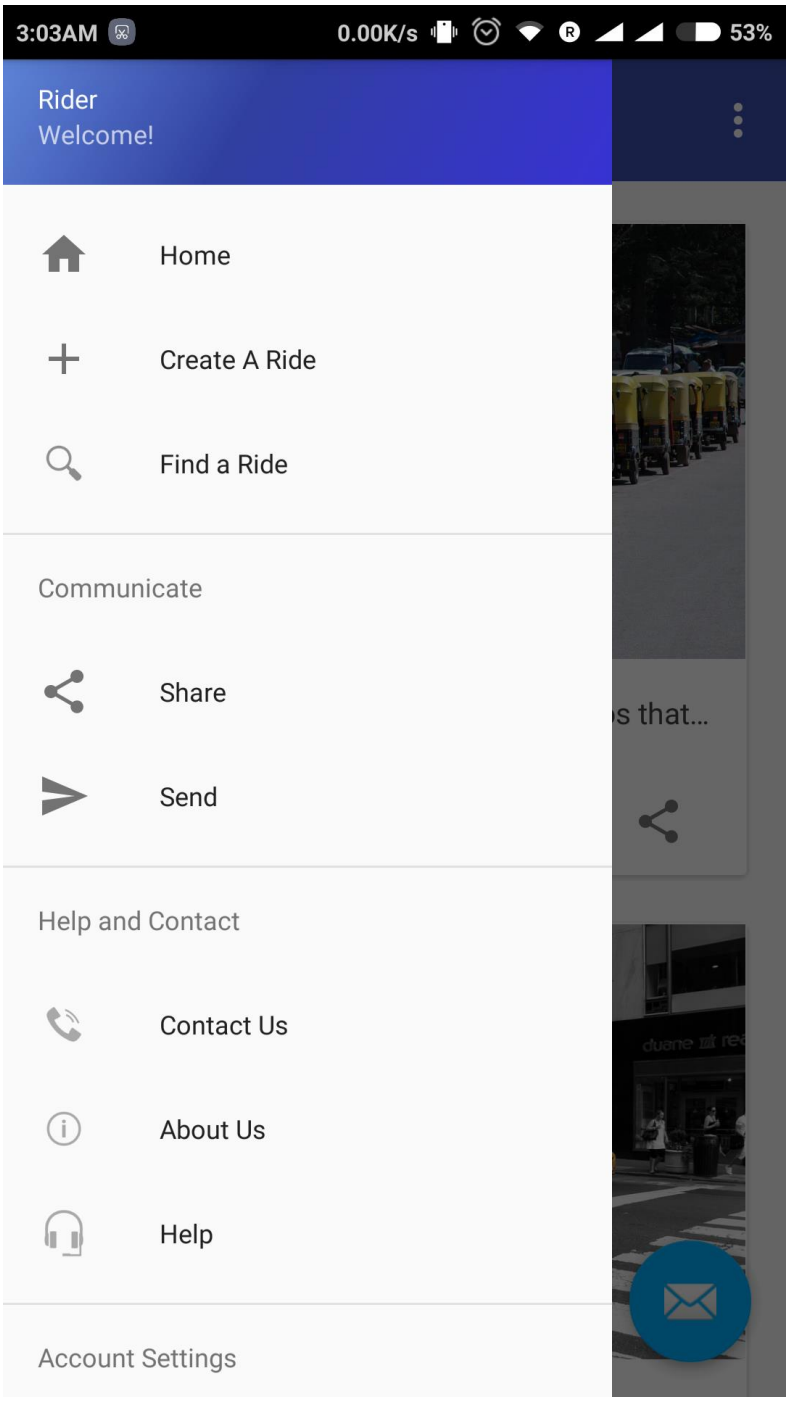
Find co-travelers: when a user searches for a co-traveler, a query is sent to the database and the results matching the criteria are displayed on screen.

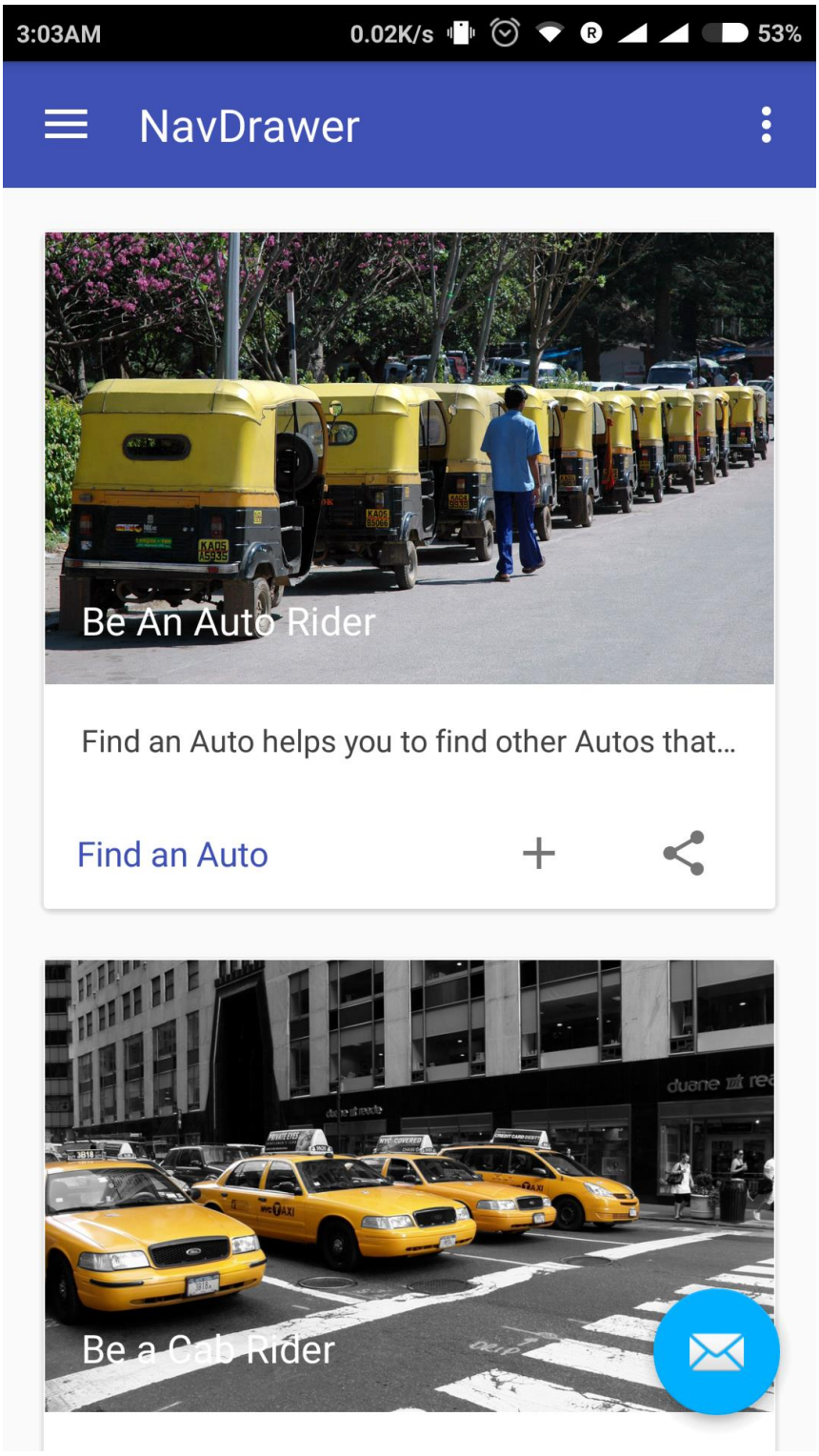
Once the users find co travelers, they can start the ride and this will delete the entry from database.


3:03AM
0.35K/s
53%
RECENT RIDES
MY POSTS
TRENDING
Ayush Singh
0
timestamp
New Auto
hurry, leaving soon.
Source: Amity University
Destination: New Text
Ayush Singh
1
timestamp
gfvh
gdjsty
Source: Amity University
Destination: New Text
Ayush Singh
0
timestamp
theean
chot yffj
Source: Amity University
Destination: New Text


3:03AM
4.32K/s
53%
Ayush Singh
New Auto
hurry, leaving soon.
Source: Amity University
Destination: Sector 18 Metro
Write a comment...
POST
aa
go
ha
q w e r t y u i o p
a s d f g h j k l
z x c v b n m
#+=
English (US)
.


3:03AM
6.67K/s
53%
Create a new Ride
Is it a Cab or an Auto?
Auto
Title
Destination
Description...

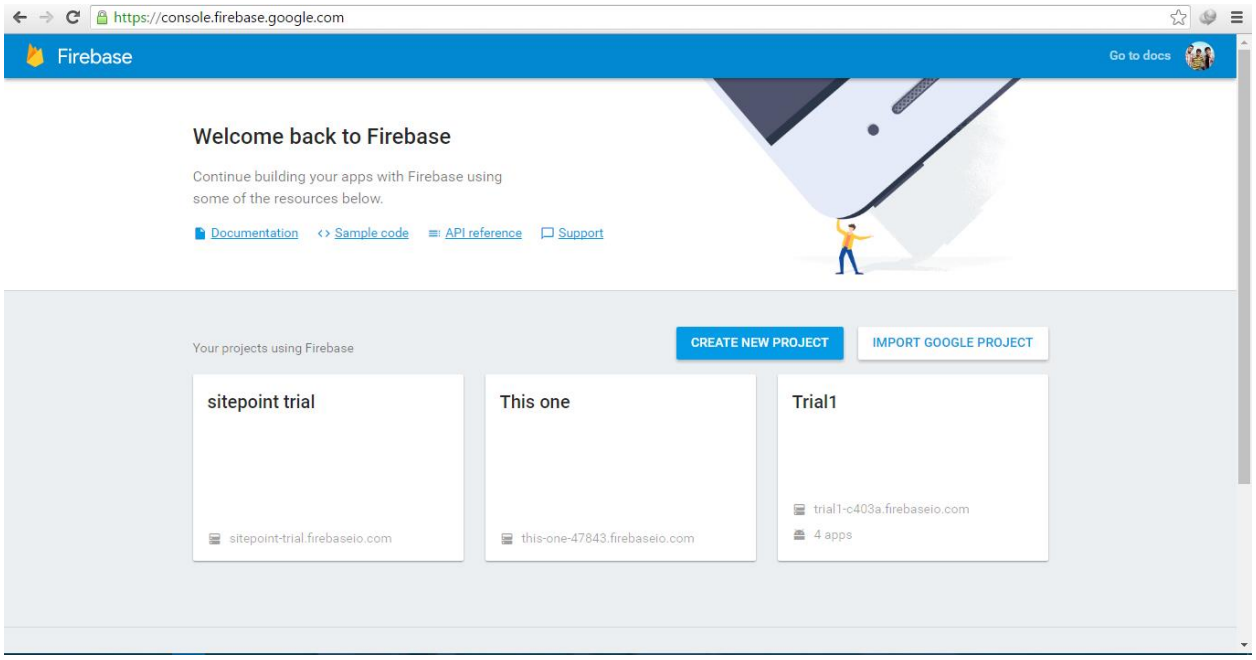

3:03AM
0.00K/s
53%
Rider
Welcome!
Home
Create A Ride
Find a Ride
Communicate
Share
Send
Help and Contact
Contact Us
About Us
Help
Account Settings


3:03AM
0.02K/s
53%
NavDrawer
Be An Auto Rider
Find an Auto helps you to find other Autos that...
Find an Auto
Be a Cab Rider

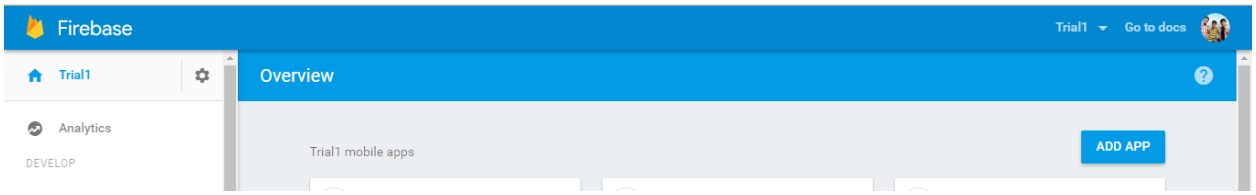
6. Methodology

6.1 We are using firebase as backend. The following steps show how to set up firebase as a backend for the application.

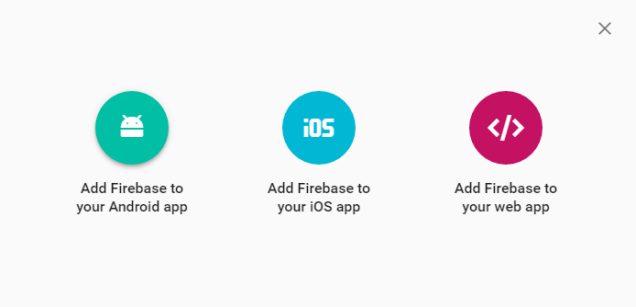
1. Open the firebase website. If you are already a user of google, then you don't even need to signup or signin as you automatically get signed into firebase. Open console and then follow the next steps to set up your firebase.



2. On your console page, select create a new project and enter the details asked for the application. After that ypu will be taken to the application page as shown in the image below. Select the **ADD APP** option to add your application.



3. On selecting the **ADD APP** button, you will be asked to select the kind of device or platform you want to add your app to. Firebase currently suports Android, iOS, and Web Apps. Here we are making an android app, so we'll go with android for now.



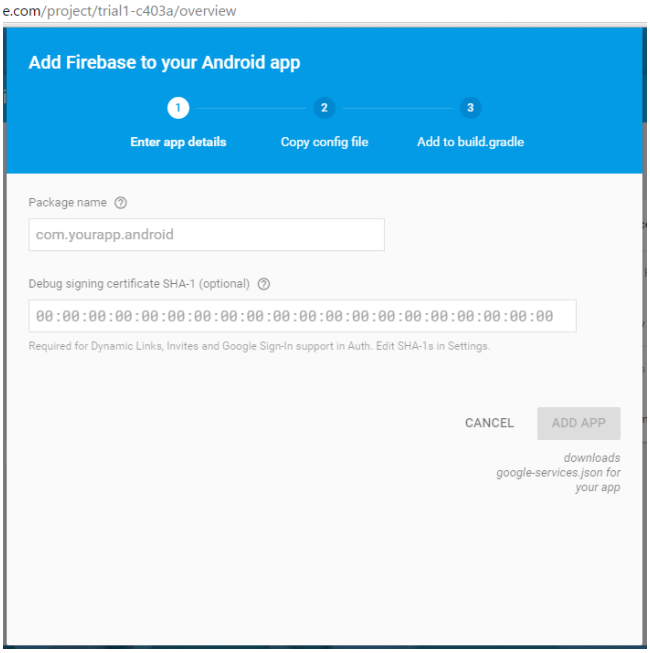
4. After selecting android as an option, you will be asked to enter the package name of the application that you are making. This can be any name and there are no restrictions on this, but make sure you use your own name and not some copyright name of some company brand or product.

   Below the package name of the application, you will be asked to enter the SHA-1 code of your application. This is also known as debug signing code here. Although this is shown optional, but we still need to enter this if we want to use features like authenticating our users to the firebase cloud. The process to obtain the key can be seen by tapping the small question mark above the field.

   On a windows device, the user can input the following line of code in the **Android Studio** condole to get the SHA-1 key.

```
keytool -exportcert -list -v \
-alias androiddebugkey -keystore %USERPROFILE%\.android\debug.keystore
```

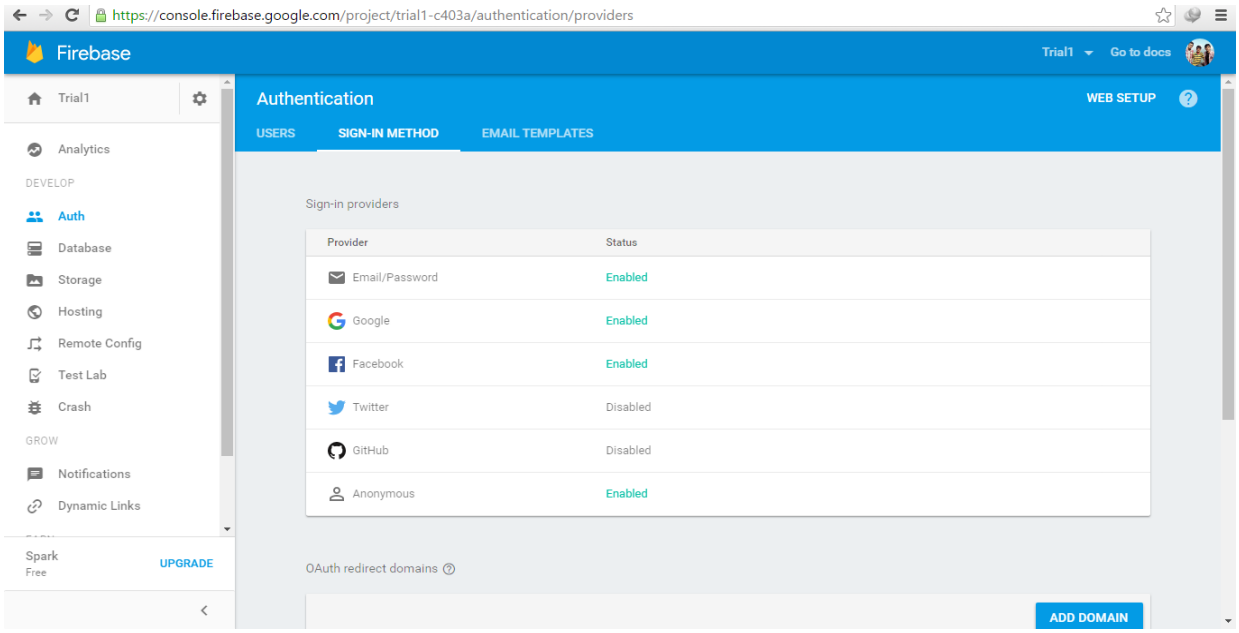
After entering the key obtained from the above code into the image below, a **JSON** file is created, and downloaded. This JSON file helps connect our app to the firebase server. This file must be pu in the “app” folder in the location where the project is stored in the Hard Disk. Without this file, the firebase code doesn’t build in the android studio.

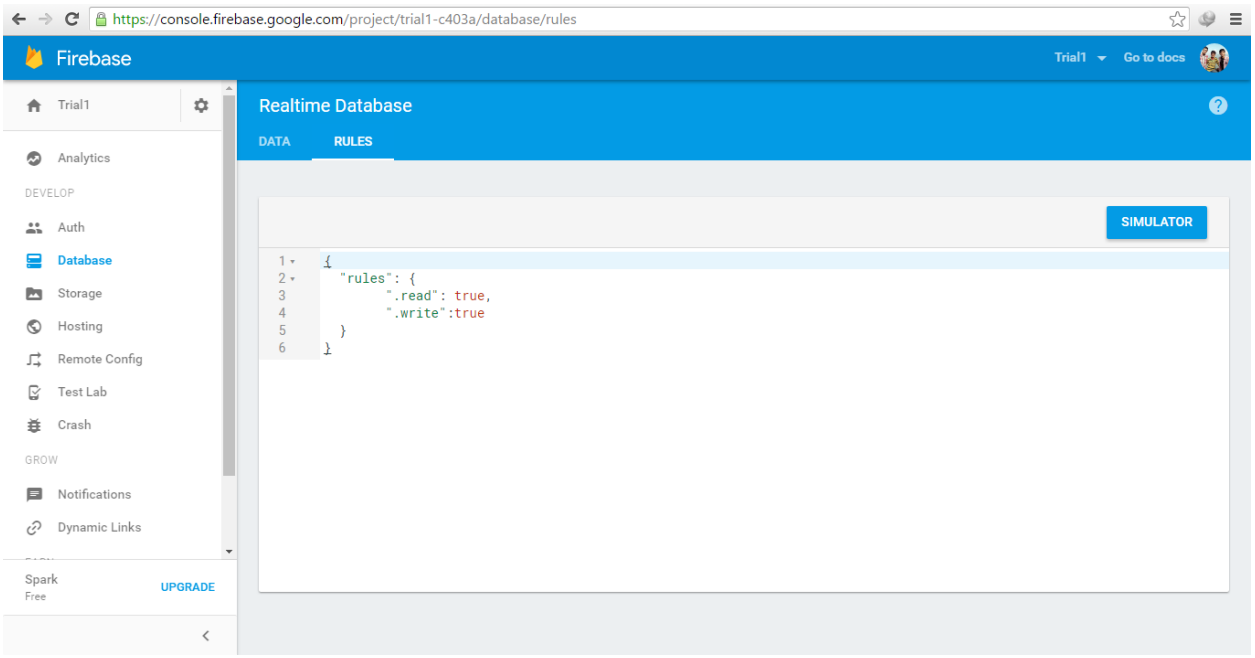


5. After the above steps, you are almost all set up and you can start coding your app in the Android Studio, However, there are a few more things that need to be taken care of before we switch over to android studio. These include setting up the signin methods, defining database rules etc.

   To set up the login methods, select **Auth** in the navigation pane and then select **SIGN-IN METHOD** tab from the three tabs shown. Under this, select the methods using which you want to allow the user to log in to the app and access database.

6. Setting up the database rules is a very important task as we don't want unauthorised people to access the database. So we need to change the default rules that give anyone access to the database to stricter rules. The following image shows the default rules of the database. These rules will allow anyone to get access to the database.





These rules will allow only the authenticated users with a valid auth token from firebase to access and modify the database.

```
// These rules grant access to a node matching the authenticated
// user's ID from the Firebase auth token
{
  "rules": {
    "users": {
      "$uid": {
        ".read": "$uid === auth.uid",
        ".write": "$uid === auth.uid"
      }
    }
  }
}
```

It is essential that you configure these rules correctly before launching your app to ensure that your users can only access the data that they are supposed to.

These rules will allow any authenticated user to access and modify the database.

```
// These rules require authentication
{
  "rules": {
    ".read": "auth != null",
    ".write": "auth != null"
  }
}
```

It is essential that you configure these rules correctly before launching your app to ensure that your users can only access the data that they are supposed to.

```
// These rules give anyone, even people who are not users of your app,
// read and write access to your database
{
  "rules": {
    ".read": true,
    ".write": true
  }
}
```

It is essential that you configure these rules correctly before launching your app to ensure that your users can only access the data that they are supposed to.

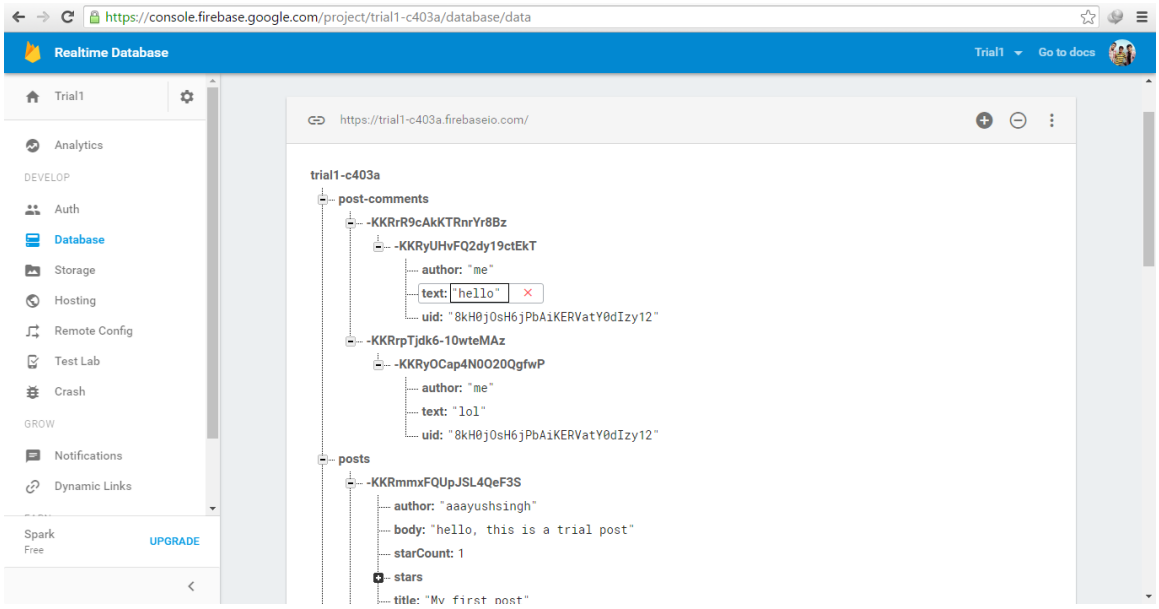
These allow the people, even those who are not the users of our app access to the database.

```
// These rules don't allow anyone read or write access to your database
{
  "rules": {
    ".read": false,
    ".write": false
  }
}
```

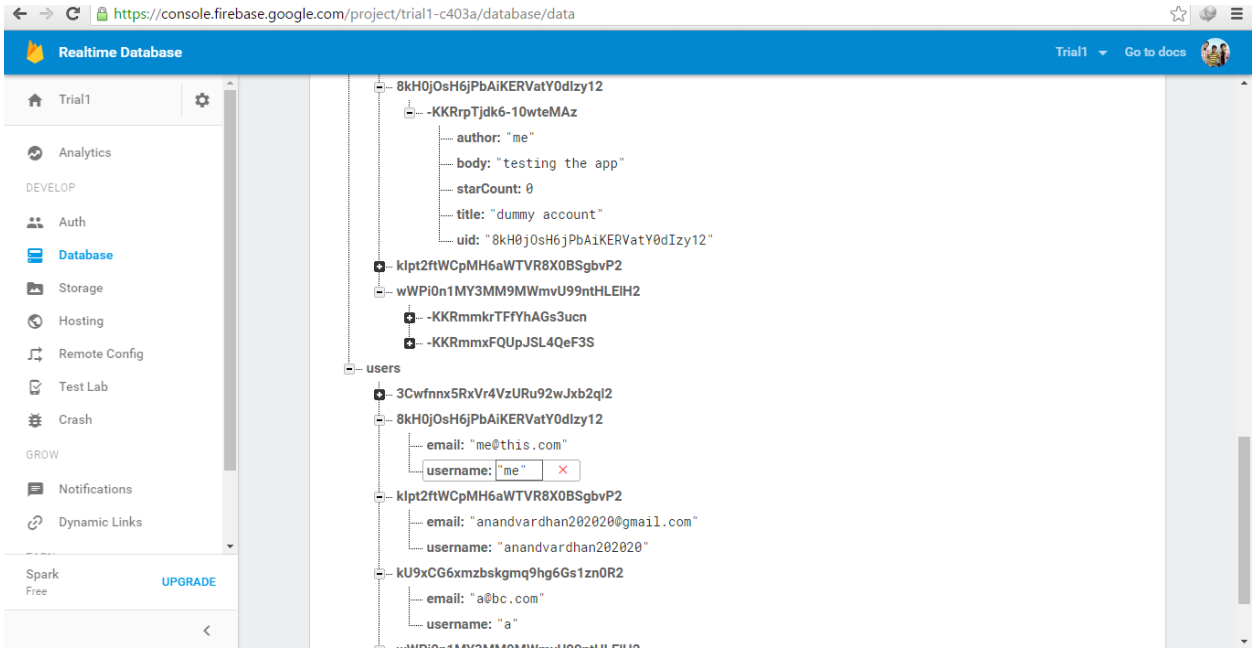
It is essential that you configure these rules correctly before launching your app to ensure that your users can only access the data that they are supposed to.

These rules make the database completely private. Noone is allowed to access or modify the contents of the database until these rules are changed by the admin.

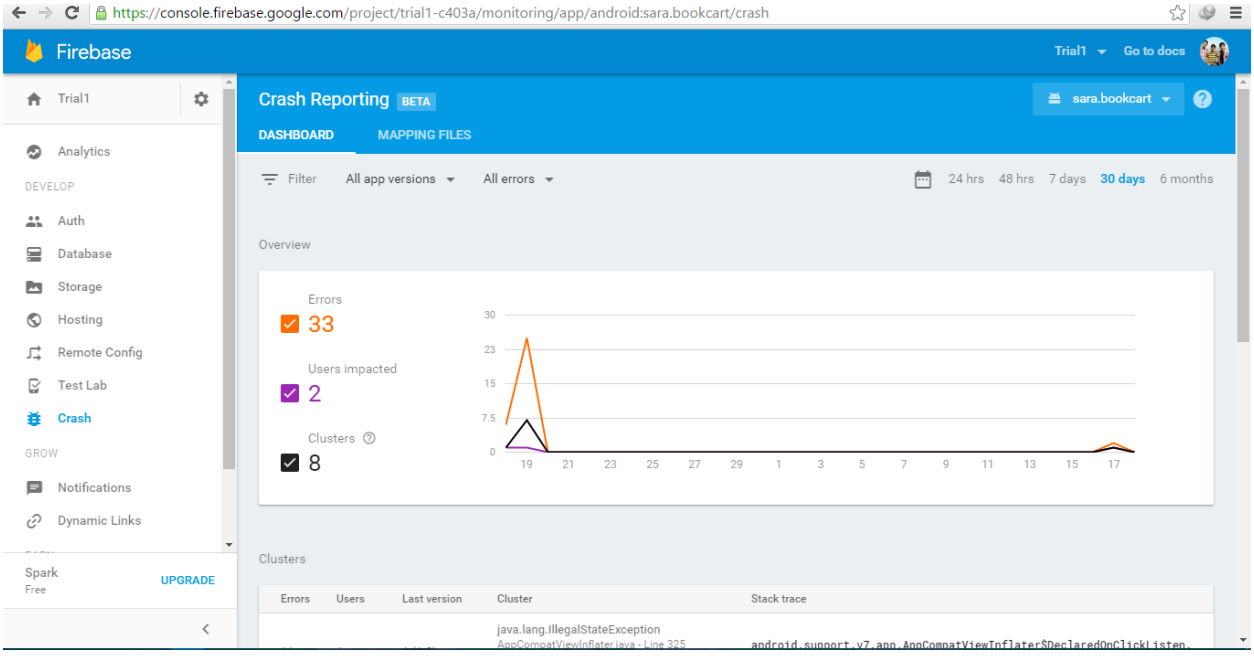
7. Once we are done with the above things and start populating the database, it starts to look like this. All the data in the database is stored under user token which are generated when they signup. These tokens are generated autmatically by the database. We can also manually add data to this database, but incorrect format will cause app to crash when trying to access the database looking for certain data.



This image is another screenshot of the database that shows the users that have signed up into our app.



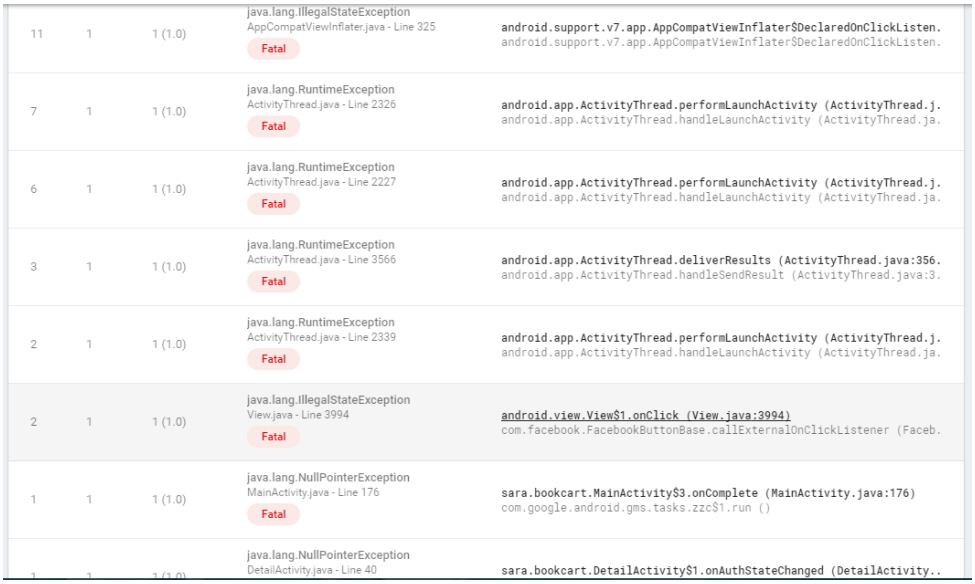
8. Firebase offers the developers an option to monitor app crashes. Each time an app crashes, the report is sent to the firebase server that keeps all the details of the crash that can be seen and monitored by the developer or the admin so that they can fix the problems without the need to ask user to



upload the crash report themselves. This is a very useful feature and gives in depth and technical details of the cause of crash.

It also keeps list of the number of app crashes, the devices in which the app crashed, and the number of users affected by the crash.

The following image is a screenshot of the crash reasons or bugs that couldn't be handled by default and caused the application to be terminated unexpectedly.



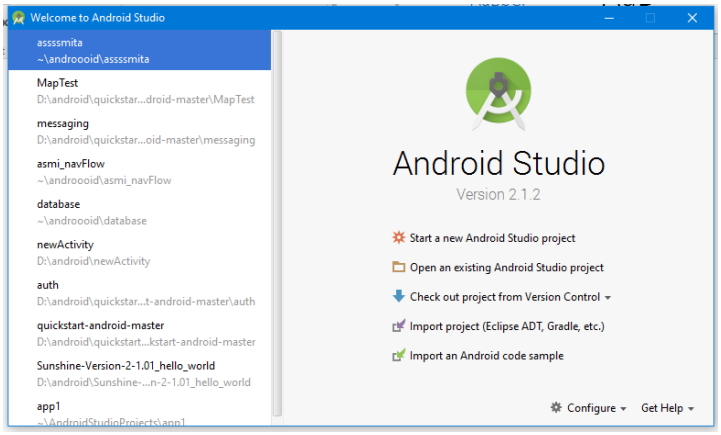
11	1	1 (1.0)	java.lang.IllegalStateException AppCompatViewInflater.java - Line 325 Fatal	android.support.v7.app.AppCompatViewInflater$DeclaredOnClickListen. android.support.v7.app.AppCompatViewInflater$DeclaredOnClickListen.
7	1	1 (1.0)	java.lang.RuntimeException ActivityThread.java - Line 2326 Fatal	android.app.ActivityThread.performLaunchActivity (ActivityThread.j. android.app.ActivityThread.handleLaunchActivity (ActivityThread.ja.
6	1	1 (1.0)	java.lang.RuntimeException ActivityThread.java - Line 2227 Fatal	android.app.ActivityThread.performLaunchActivity (ActivityThread.j. android.app.ActivityThread.handleLaunchActivity (ActivityThread.ja.
3	1	1 (1.0)	java.lang.RuntimeException ActivityThread.java - Line 3566 Fatal	android.app.ActivityThread.deliverResults (ActivityThread.java:356. android.app.ActivityThread.handleSendResult (ActivityThread.java:3.
2	1	1 (1.0)	java.lang.RuntimeException ActivityThread.java - Line 2339 Fatal	android.app.ActivityThread.performLaunchActivity (ActivityThread.j. android.app.ActivityThread.handleLaunchActivity (ActivityThread.ja.
2	1	1 (1.0)	java.lang.IllegalStateException View.java - Line 3994 Fatal	android.view.View$1.onClick (View.java:3994) com.facebook.FacebookButtonBase.callExternalOnClickListener (Faceb.
1	1	1 (1.0)	java.lang.NullPointerException MainActivity.java - Line 176 Fatal	sara.bookcart.MainActivity$3.onComplete (MainActivity.java:176) com.google.android.gms.tasks.zzc$1.run ()
1	1	1 (1.0)	java.lang.NullPointerException DetailActivity.java - Line 40	sara.bookcart.DetailActivity$1.onAuthStateChanged (DetailActivity..

Our work with firebase is done for now. Let's head over to android studio to start building tha app.

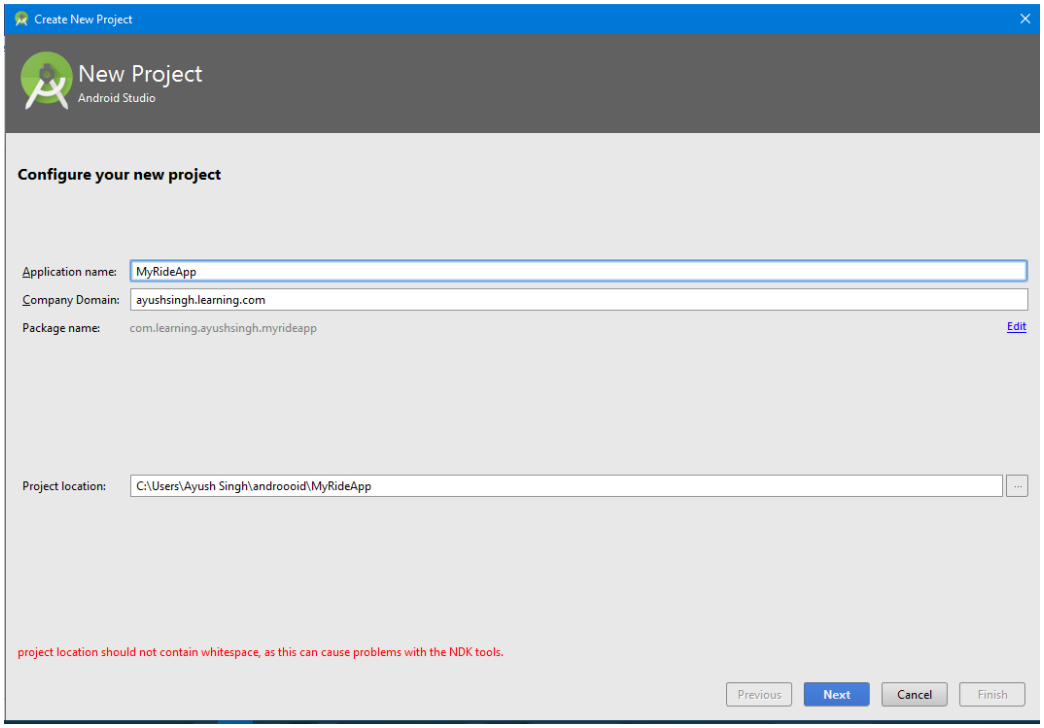
6.2 Android Studio

Now, moving on to the second part of building the application, i.e. working on android studio.

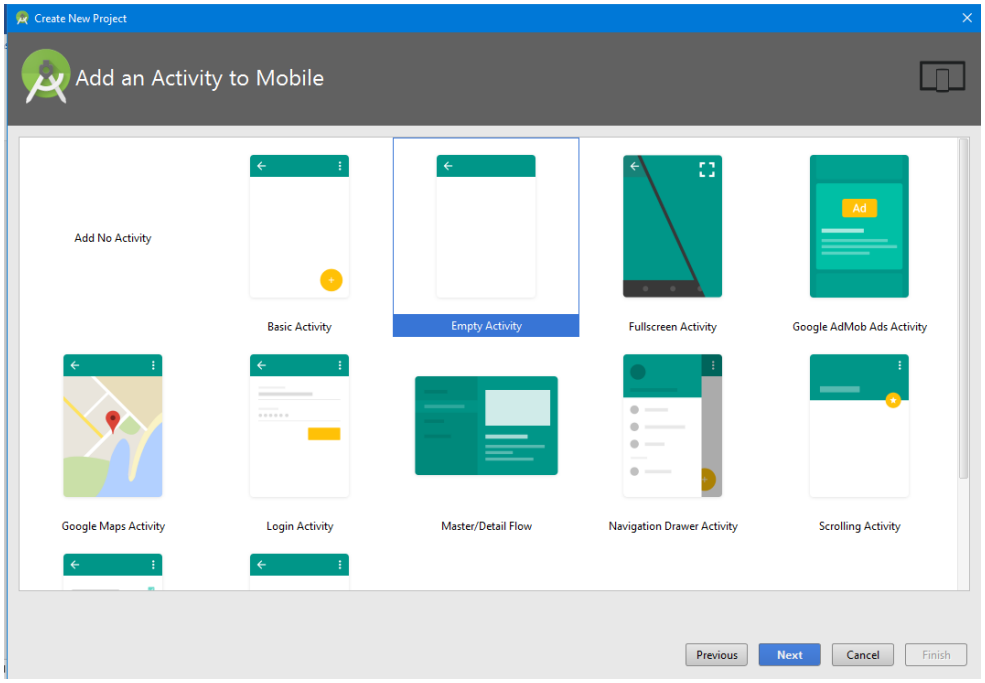
1. Open Android Studio and select **Start a new Android Studio project.**



2. On the next page, choose the name of the application and the name of your company.

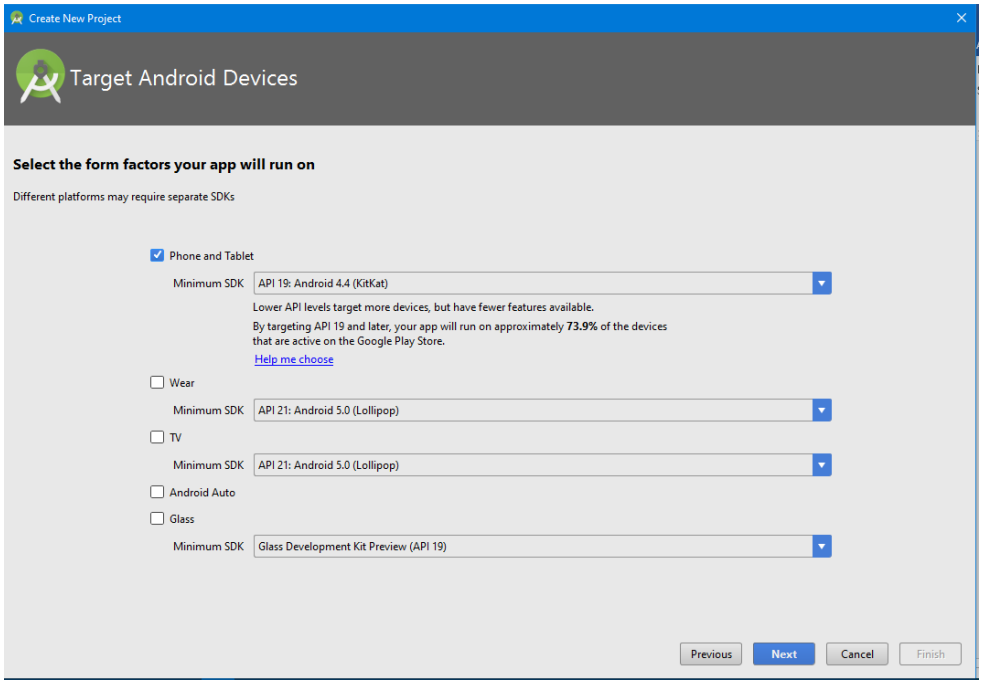


3. On the target devices page, choose phone and tablet only as we will not be programming the application for other devices now.

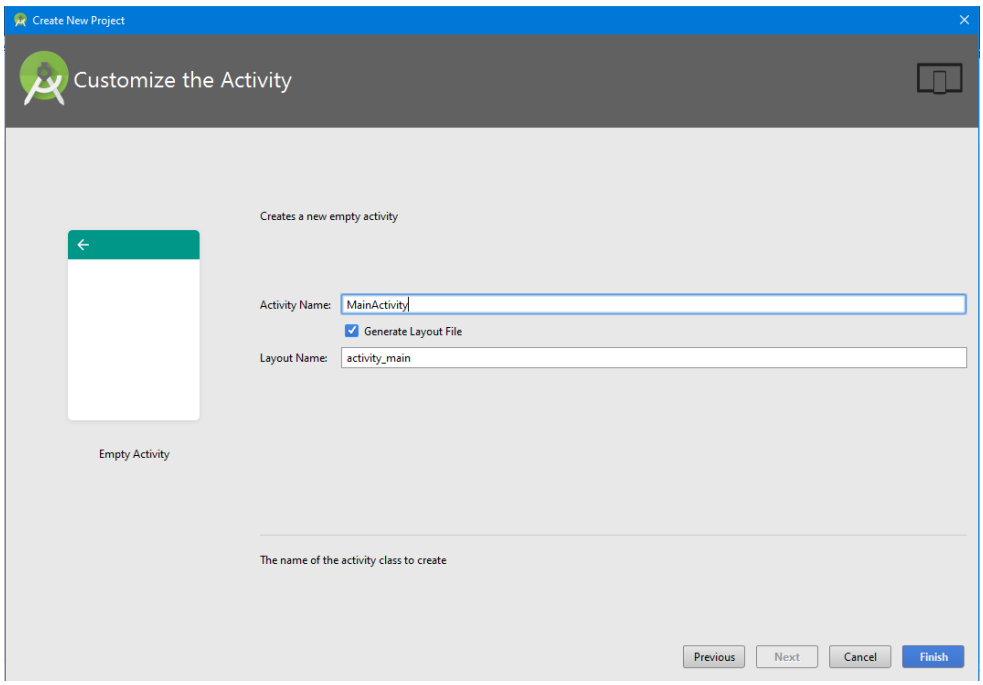


4.

On the next page, select a blank activity to start with.



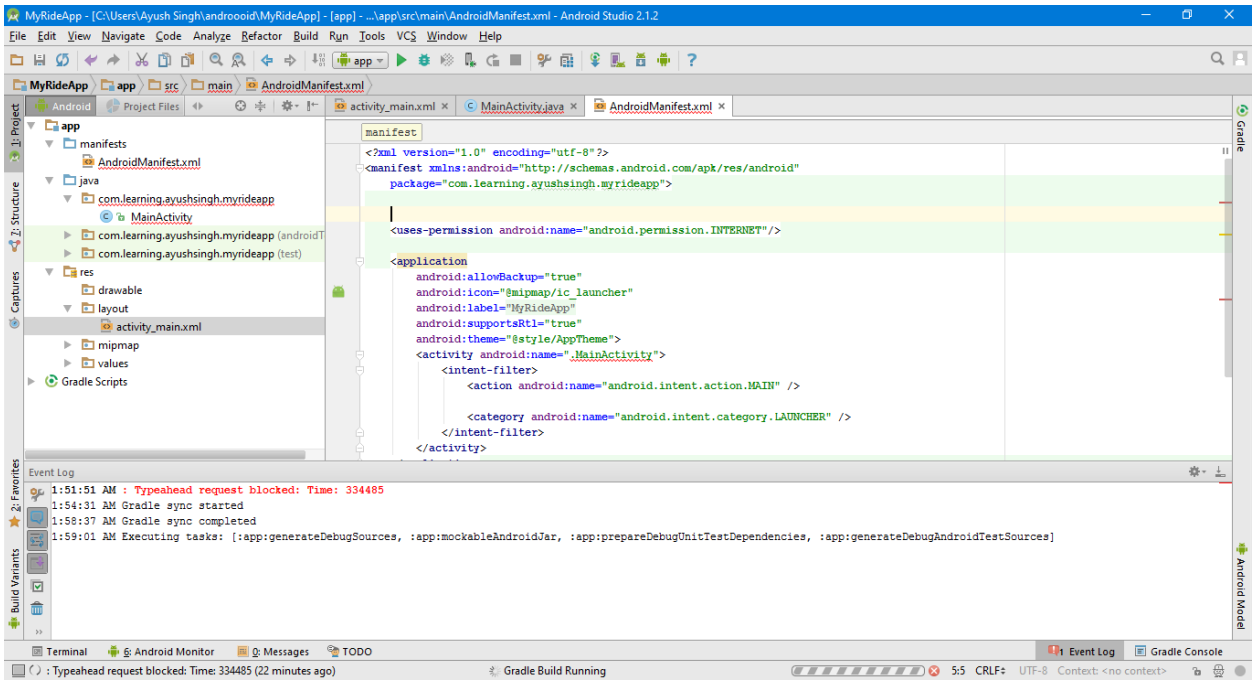
Choose any name for the blank activity and then click finish. This will now leave you with the universal Hello World! Application for android which will do nothing but display hello world on the screen.



5. Now, since we will be using internet in our application, we need to declare that in the manifest file of the application.

The manifest file keeps record of all the permissions and activities in the application. Any activity that is not mentioned in the manifest will not be comiled in the final building of the application, and will be removed if someone performs a clean build of the application.

We are all set up and reay to code now.



- Design a base acivity class and put any global data that you want to use in the rest of the application here. Other classes can inherit this class, and override its methods. This BaseActivity class should extend AppCompactActivity as we neeed all the classes to extend AppCompactActivity.

```
import ...
/**
 * taken from firebase db
 */

public class BaseActivity extends AppCompatActivity {

    private ProgressDialog mProgressDialog;

    public void showProgressDialog() {
        if (mProgressDialog == null) {
            mProgressDialog = new ProgressDialog(this);
            mProgressDialog.setCancelable(false);
            mProgressDialog.setMessage("Loading...");
        }

        mProgressDialog.show();
    }

    public void hideProgressDialog() {
        if (mProgressDialog != null && mProgressDialog.isShowing()) {
            mProgressDialog.dismiss();
        }
    }

    public String getUid() { return FirebaseAuth.getInstance().getCurrentUser().getUid(); }
}
```

- The above image shows the code contained in the BaseActivity class for our application.
- The next step is to design a login screen. The XML file or the layout file of the login screen contains the following code.

```
<?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:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/bg2"
    android:paddingBottom="16dp"
    android:paddingLeft="16dp"
    android:paddingRight="16dp"
    android:paddingTop="16dp"
    tools:context="sara.bookcart.OtherLogin">

    <ImageView
        android:id="@+id/icon"
        android:layout_width="170dp"
        android:layout_height="160dp"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="40dp"
        android:layout_marginBottom="16dp"
        android:src="@drawable/cablogo" />

    <LinearLayout
        android:id="@+id/layout_email_password"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="10dp"
        android:layout_below="@+id/icon"
        android:layout_centerHorizontal="true"
        android:orientation="horizontal">
```

```
    <EditText
        android:id="@+id/field_email"
        android:layout_width="150dp"
        android:layout_height="wrap_content"
        android:ellipsize="end"
        android:gravity="center_horizontal"
        android:hint="email"
        android:inputType="textEmailAddress"
        android:maxLines="1" />

    <EditText
        android:id="@+id/field_password"
        android:layout_width="150dp"
        android:layout_height="wrap_content"
        android:ellipsize="end"
        android:gravity="center_horizontal"
        android:hint="password"
        android:inputType="textPassword"
        android:maxLines="1" />

</LinearLayout>

<LinearLayout
    android:id="@+id/layout_buttons"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@+id/layout_email_password"
    android:layout_centerHorizontal="true"
    android:orientation="horizontal">

    <Button
        android:id="@+id/button_sign_in"
        android:layout_width="150dp"
        android:layout_height="wrap_content"
        android:text="Sign In" />


    <Button
        android:id="@+id/button_sign_up"
        android:layout_width="150dp"
        android:layout_height="wrap_content"
        android:text="Sign Up" />

</LinearLayout>
<com.facebook.login.widget.LoginButton
    android:id="@+id/login_button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerVertical="true"
    android:layout_centerHorizontal="true"
    android:layout_below="@id/layout_buttons"
    android:layout_marginTop="60dp"
    />


</RelativeLayout>
```

The following image shows the various stages in the life cycle of an application. These methods can be overridden fpr a better user experience and more control on how the app handles these events.

Similrly, we code the rest of the application step by step. All the code cant be shared here due to the size of the files.

7. Result and Discussion

Who doesn’t want to save both time and money when it comes to everyday things. Instead of needing to look for peopple manually, it would be mush better if there was someone or something that could do it for us.

In times of the odd even rule, finding someone we can commute to work with would be the best thing that could happen as we wouldn’t have to get into the crowded bus, and avoiding all that hassle and hardwork just to get to work would be cleared or removed.

We created an android application would help us find fellow travellers so that we can split the cab fare or share our personal vehicle.

This application tackles the everyday problem of a lot of people by finding other people with same destination instantly. Such users can then talk to each other and decide the terms and conditions themselves.

There is no middleman or middle service involved here that they will have to pay the taxes or any kind of money to, so the money matters are totally in the hands of the people travelling together.

Who knows, you just might make some new friends.

8. Conclusion(s) and Recommendations

The Basic idea when building the app was to help others in their daily commute. The app fulfils this basic idea by letting all other nearby users find each other by creating or looking for entries from each other.

This is a money saving app, and the small amount saved daily per ride becomes big over time. This can be invested or used elsewhere, After all, who doesn’t want to save money?

The app works wonderfully, and the material UI implementedin the app gives it a good look and feel. However, it is recommended that people do not go wth complete strangers that they do not trust sinceanything can happen through the journey.

9. Future Research and Work

This has a lot of scope for future research and work as this app helps us in everyday situations. Right now, this app supports only Amty University as starting point, but this can be later changed. Also, several other options such as changing passwords, forgotten passwords links, changing usernames or setting up custom usernames, setting user profile images etc can be added in the later stages of development.

The present version of the app runs on the free service of firebase, and can have only upto 50 simultaneous connctions at a time. In future developments, a full paid version of firebase can be usedto allow more user to this app, since the more the users, the easier it is to find other **Riders.**

10. References

1. Andrei Frumusanu (July 1, 2014). "*A Closer Look at Android RunTime (ART) in Android L"*. AnandTech. Retrieved July 5, 2016
2. Boyer, Rick & Mew, Kyle, *Android Application Development Cookbook,* Second Edition.
3. Google Inc, Website *https://developers.android.com,*
4. Portales, Raul, *Mastering Android Application Development.*
5. Wikipedia, *www.wikipedia.com, Android versions history,* Retrieved July 5, 2016.
6. Android Development Follow-up instructions on getting started with Android Development.
7. https://developer.android.com/develop/
8. Stack Overflow Q&A website aiming for programmers and developers. *http://stackoverflow.com/*
9. Google Codelab, Website https://codelabs.developers.google.com