STOC-UP

FINAL PROJECT DOCUMENTATION

DIVYA SAI SEKHAR MULLAPUDI - 755658447

DUSHYANT SHEORAN - 728452028

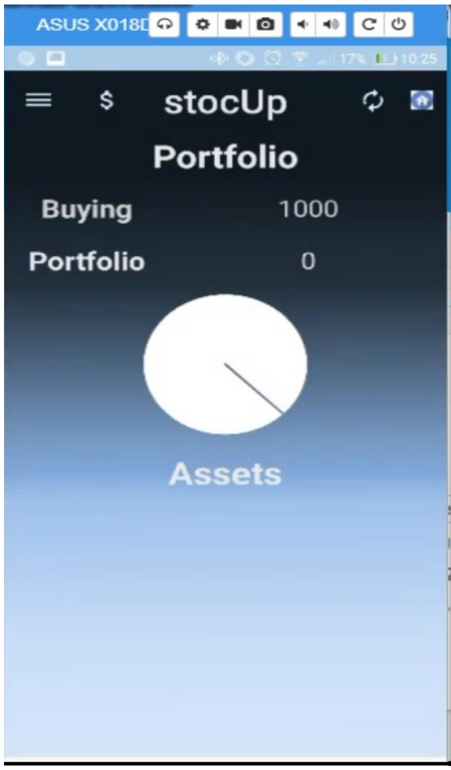
Contents

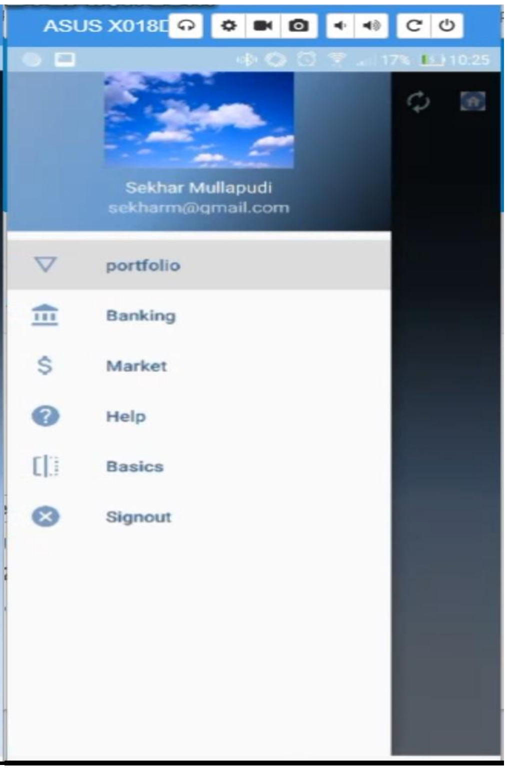
1. INTRODUCTION

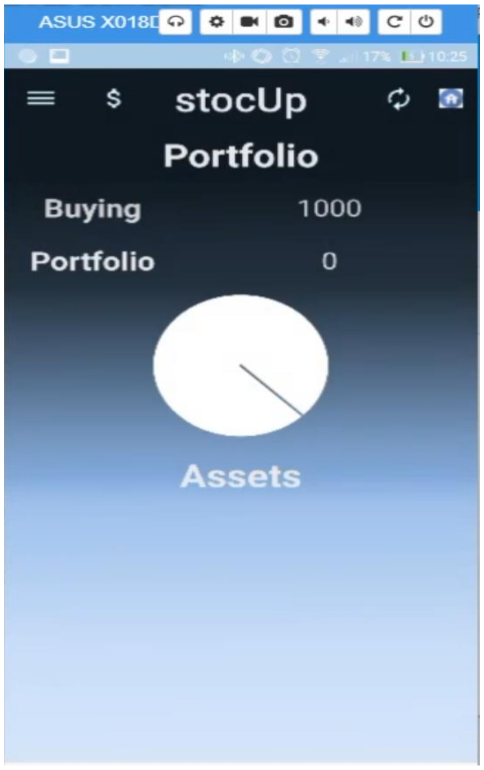
- An application that helps users to get a first-hand experience of trading stocks and crypto-currencies. Users can buy and sell stocks.
- Users can check their portfolio and track activity of stocks and crypto-currencies in the virtual data which tries to reciprocate the real world data.
- Users can maintain their Wallets easily. The app is based on a virtual currency system.
- Users can learn about stock markets and Crypto-Currency Investment using the Lecture notes provided in the app.

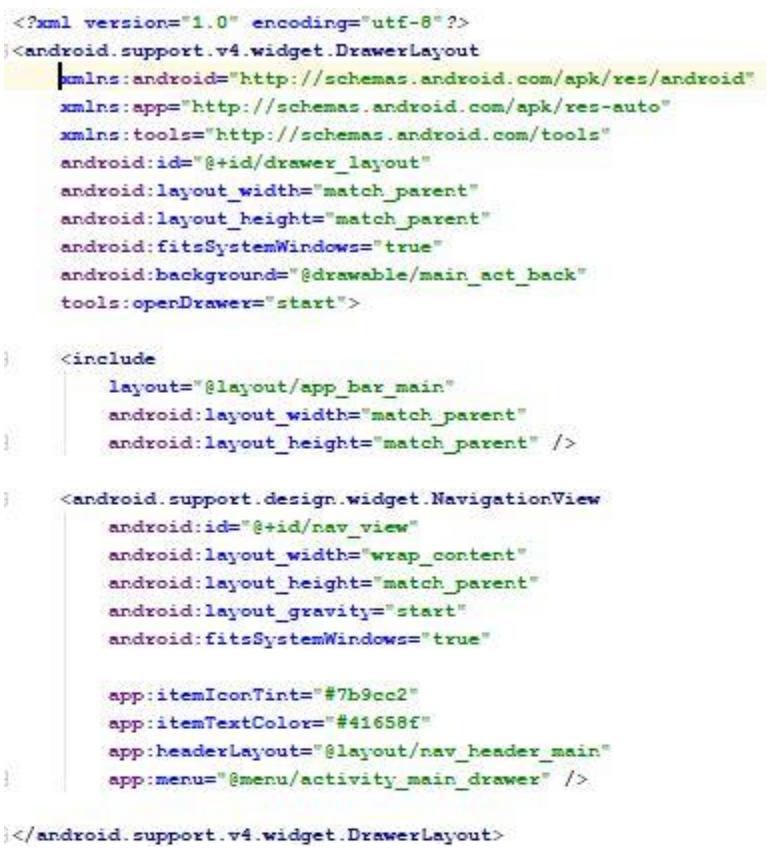
2. NAVIGATION DRAWER

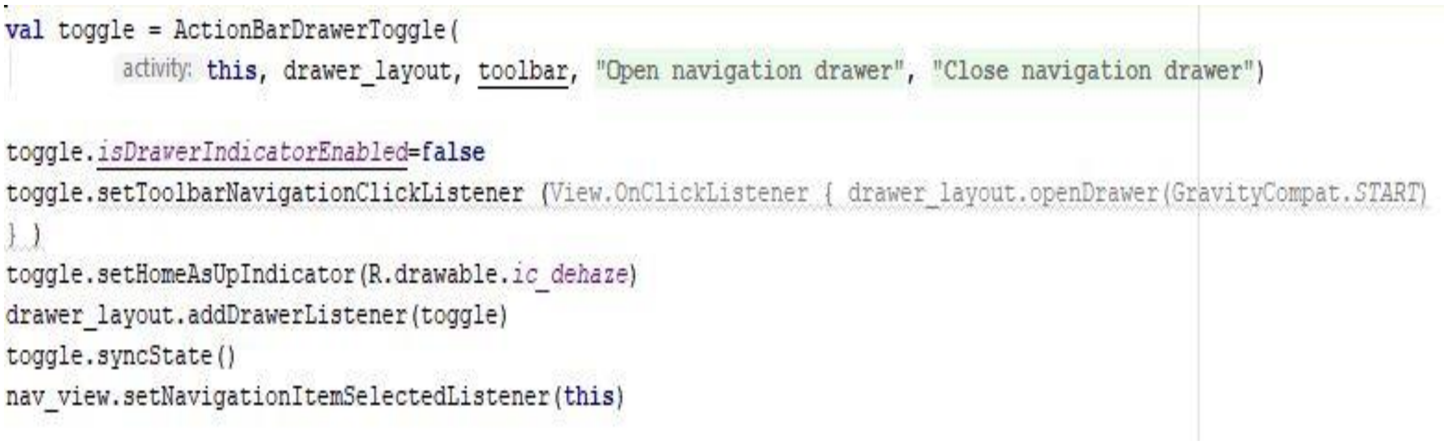
- The navigation drawer is a panel that displays the app's main navigation options on the left edge of the screen.
- It is hidden most of the time and is revealed when the user swipes a finger from the left edge of the screen or, while at the top level of the app, the user touches the app icon in the action bar.
- User interface which is the layout file is "Drawer Layout" as the root view of your layout. Drawer layout contains two child's: fragment and a Frame Layout.
- Main content populated at runtime using the fragment and a navigation view for navigation drawer.
- Initialize the drawer list and have to handle navigation clicks by redirecting to particular activity using intents and to fragments using transaction manager calling the new instance method of fragment
- Listen for close and open events for navigation drawer.
- Uses navigation drawer adapter to populate the navigation drawer list.



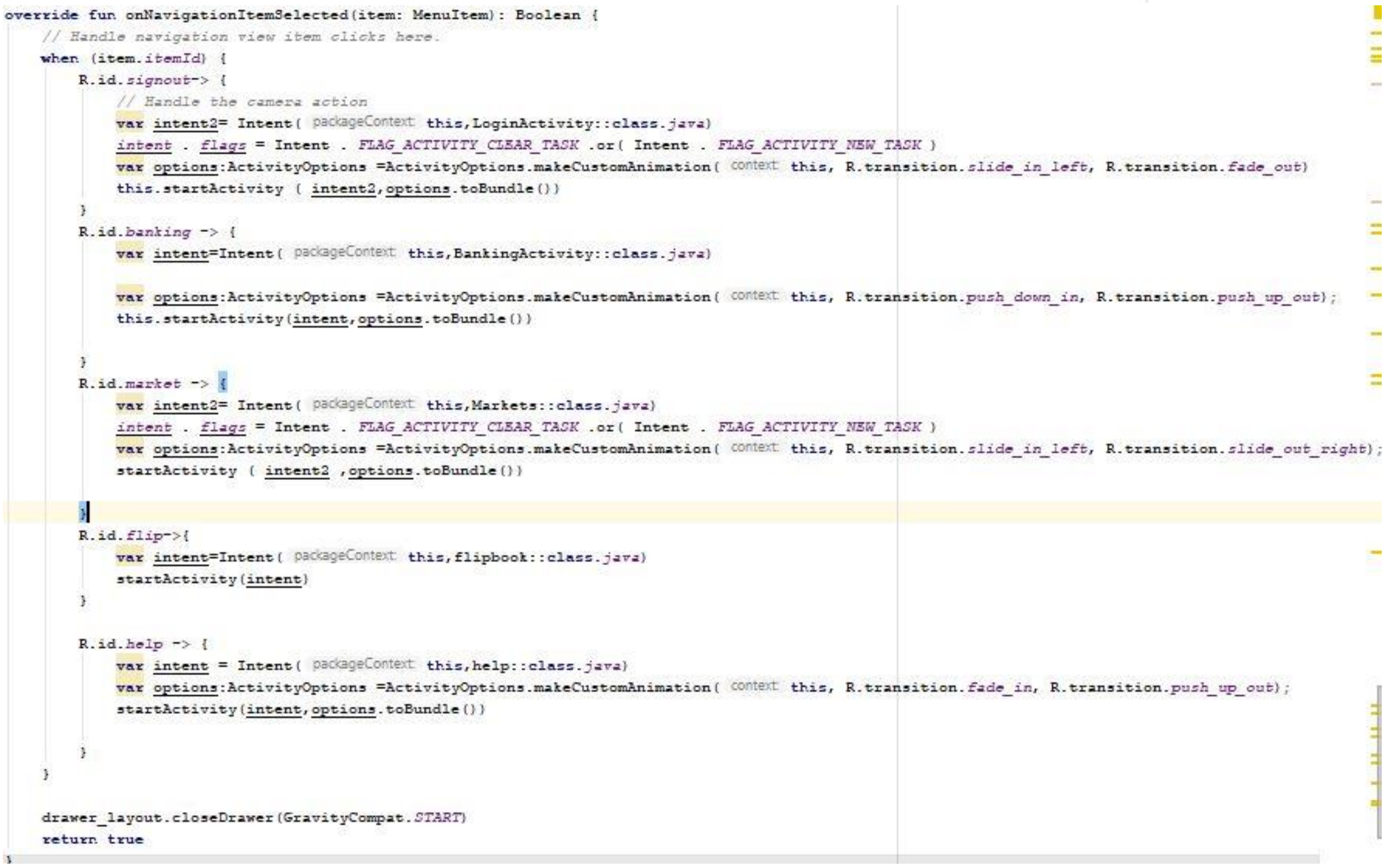





```
<?xml version="1.0" encoding="utf-8"?>
<android.support.v4.widget.DrawerLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/drawer_layout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true"
    android:background="@drawable/main_act_back"
    tools:openDrawer="start">

    <include
        layout="@layout/app_bar_main"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />

    <android.support.design.widget.NavigationView
        android:id="@+id/nav_view"
        android:layout_width="wrap_content"
        android:layout_height="match_parent"
        android:layout_gravity="start"
        android:fitsSystemWindows="true"

        app:itemIconTint="#7b9cc2"
        app:itemTextColor="#41658f"
        app:headerLayout="@layout/nav_header_main"
        app:menu="@menu/activity_main_drawer" />

</android.support.v4.widget.DrawerLayout>
```



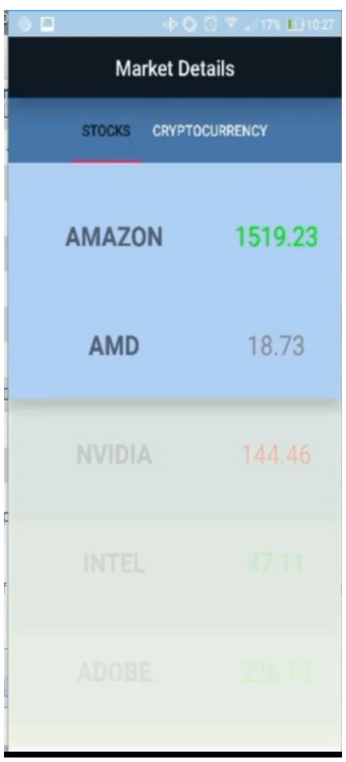
```
val toggle = ActionBarDrawerToggle(
        activity: this, drawer_layout, toolbar, "Open navigation drawer", "Close navigation drawer")

toggle.isDrawerIndicatorEnabled=false
toggle.setToolbarNavigationClickListener (View.OnClickListener { drawer_layout.openDrawer(GravityCompat.START)
} )
toggle.setHomeAsUpIndicator(R.drawable.ic_dehaze)
drawer_layout.addDrawerListener(toggle)
toggle.syncState()
nav_view.setNavigationItemSelectedListener(this)
```

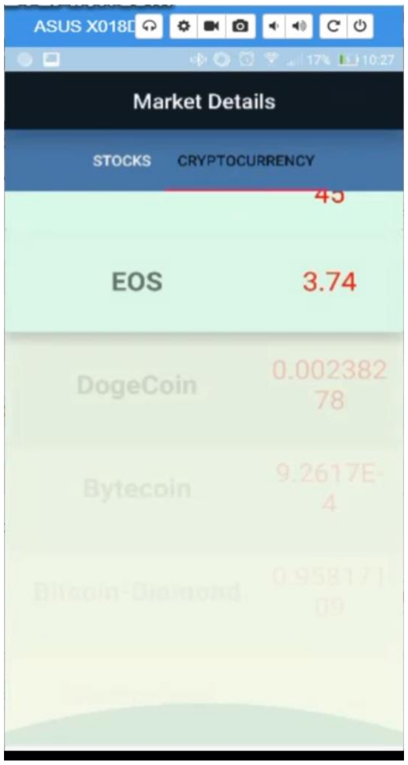


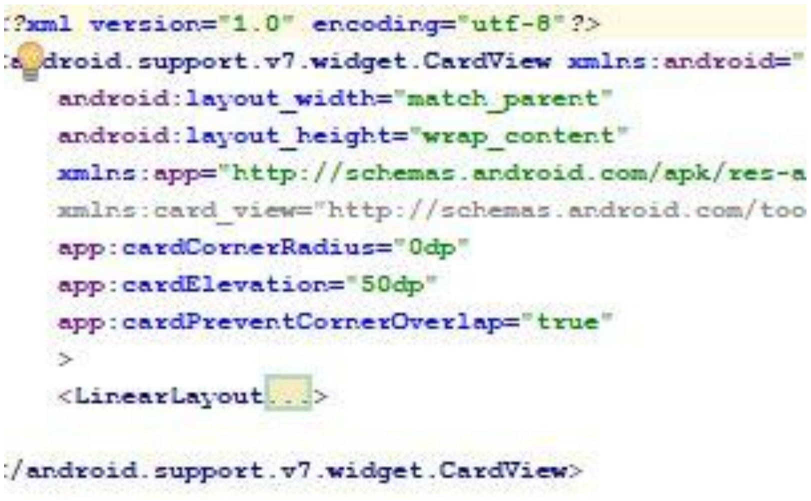
```
override fun onNavigationItemSelected(item: MenuItem): Boolean {
    // Handle navigation view item clicks here.
    when (item.itemId) {
        R.id.signout-> {
            // Handle the camera action
            var intent2= Intent( packageContext: this,LoginActivity::class.java)
            intent . flags = Intent . FLAG_ACTIVITY_CLEAR_TASK .or( Intent . FLAG_ACTIVITY_NEW_TASK )
            var options:ActivityOptions =ActivityOptions.makeCustomAnimation( context: this, R.transition.slide_in_left, R.transition.fade_out)
            this.startActivity ( intent2,options.toBundle())
        }
        R.id.banking -> {
            var intent=Intent( packageContext: this,BankingActivity::class.java)

            var options:ActivityOptions =ActivityOptions.makeCustomAnimation( context: this, R.transition.push_down_in, R.transition.push_up_out);
            this.startActivity(intent,options.toBundle())

        }
        R.id.market -> {
            var intent2= Intent( packageContext: this,Markets::class.java)
            intent . flags = Intent . FLAG_ACTIVITY_CLEAR_TASK .or( Intent . FLAG_ACTIVITY_NEW_TASK )
            var options:ActivityOptions =ActivityOptions.makeCustomAnimation( context: this, R.transition.slide_in_left, R.transition.slide_out_right);
            startActivity ( intent2 ,options.toBundle())

        }
        R.id.flip->{
            var intent=Intent( packageContext: this,flipbook::class.java)
            startActivity(intent)
        }

        R.id.help -> {
            var intent = Intent( packageContext: this,help::class.java)
            var options:ActivityOptions =ActivityOptions.makeCustomAnimation( context: this, R.transition.fade_in, R.transition.push_up_out);
            startActivity(intent,options.toBundle())

        }
    }

    drawer_layout.closeDrawer(GravityCompat.START)
    return true
}
```

3. Recycler View

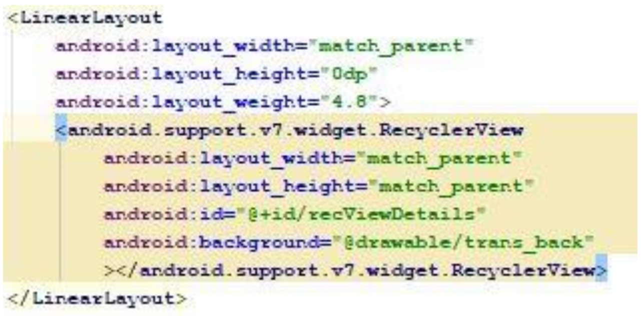
- Recycler View overcomes the drawbacks of listViews. Recycler view has been used at multiple places in our app including
- Will use recycler view with recycle adapter provided by the firebase to populate data to recycle view-holder.
- We can do animations in recycler-view while populating the data.
- Recycler view animations have been implemented when user selects a particular stock from the home page. We can see a great animation effect when the stock window loads.



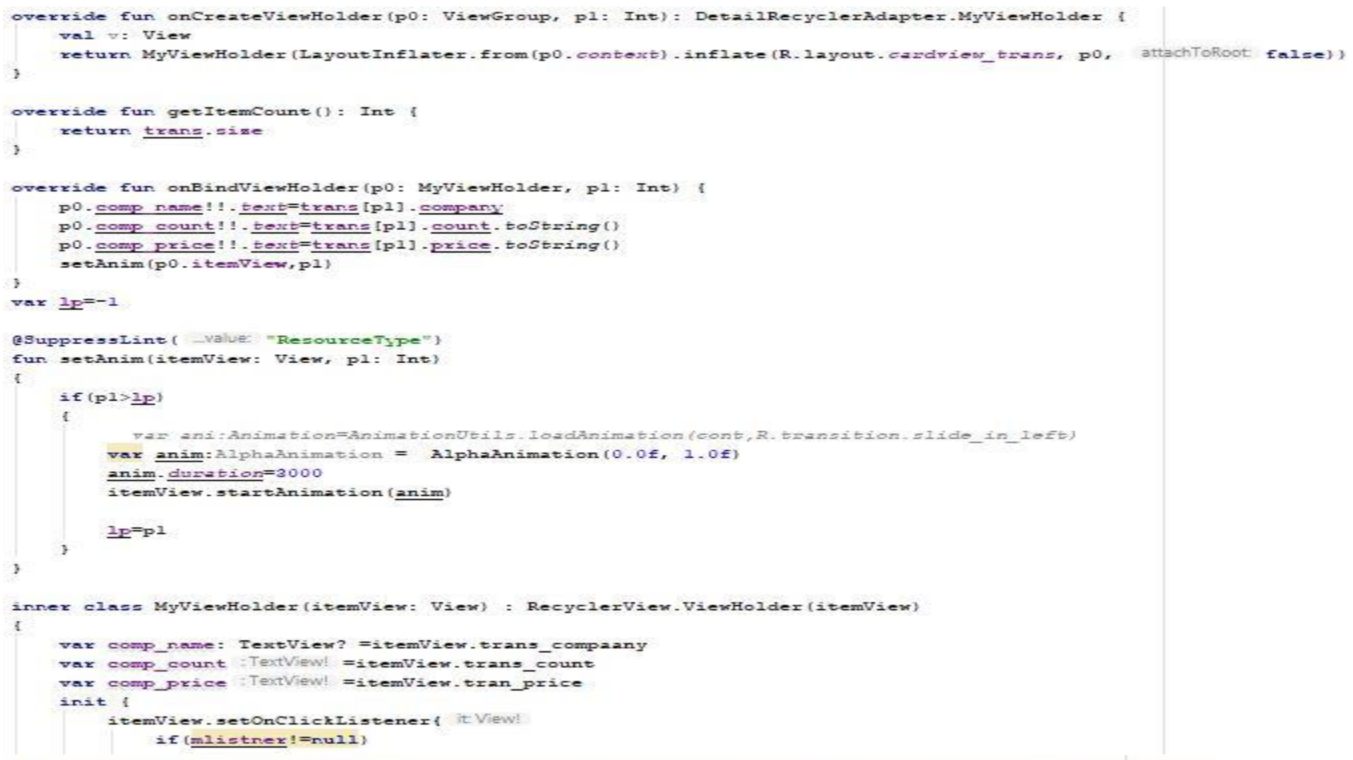




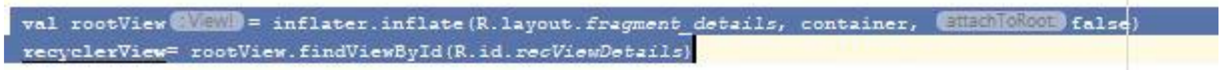
```
<?xml version="1.0" encoding="utf-8"?>
<android.support.v7.widget.CardView xmlns:android="
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    xmlns:app="http://schemas.android.com/apk/res-a
    xmlns:card_view="http://schemas.android.com/too
    app:cardCornerRadius="0dp"
    app:cardElevation="50dp"
    app:cardPreventCornerOverlap="true"
    >
    <LinearLayout...>

</android.support.v7.widget.CardView>
```



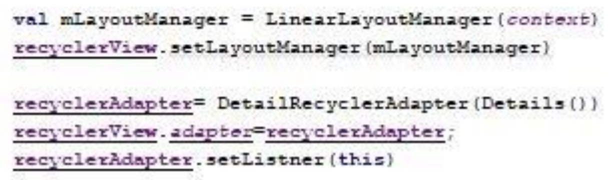
```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="4.8">
    <android.support.v7.widget.RecyclerView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/recViewDetails"
        android:background="@drawable/trans_back"
        ></android.support.v7.widget.RecyclerView>
</LinearLayout>
```



```
override fun onCreateViewHolder(p0: ViewGroup, p1: Int): DetailRecyclerAdapter.MyViewHolder {
    val v: View
    return MyViewHolder(LayoutInflater.from(p0.context).inflate(R.layout.cardview_trans, p0,  attachToRoot: false))
}

override fun getItemCount(): Int {
    return trans.size
}

override fun onBindViewHolder(p0: MyViewHolder, p1: Int) {
    p0.comp_name!!.text=trans[p1].company
    p0.comp_count!!.text=trans[p1].count.toString()
    p0.comp_price!!.text=trans[p1].price.toString()
    setAnim(p0.itemView,p1)
}
var lp=-1

@SuppressLint( ...value: "ResourceType")
fun setAnim(itemView: View, p1: Int)
{
    if(p1>lp)
    {
          var ani:Animation=AnimationUtils.loadAnimation(cont,R.transition.slide_in_left)
        var anim:AlphaAnimation =  AlphaAnimation(0.0f, 1.0f)
        anim.duration=3000
        itemView.startAnimation(anim)

        lp=p1
    }
}

inner class MyViewHolder(itemView: View) : RecyclerView.ViewHolder(itemView)
{
    var comp_name: TextView? =itemView.trans_compaany
    var comp_count :TextView! =itemView.trans_count
    var comp_price :TextView! =itemView.tran_price
    init {
        itemView.setOnClickListener{ it: View!
            if(mlistner!=null)
```



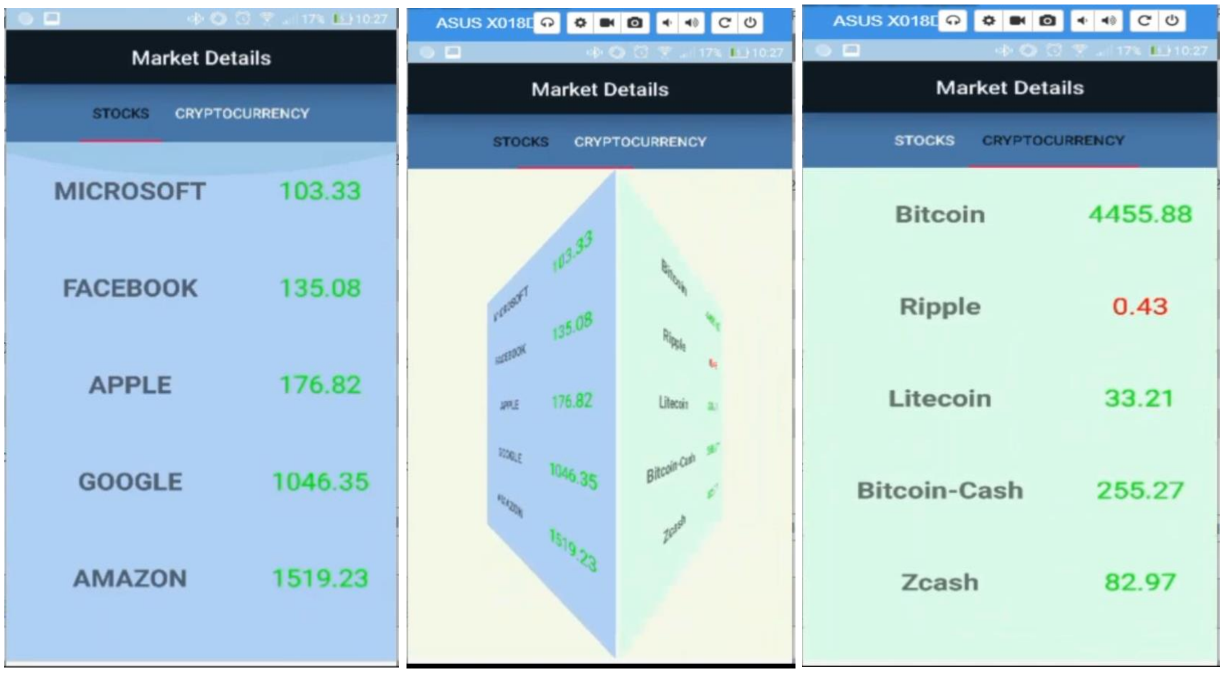
```
val rootView :View! = inflater.inflate(R.layout.fragment_details, container,  attachToRoot: false)
recyclerView= rootView.findViewById(R.id.recViewDetails)
```



```
val mLayoutManager = LinearLayoutManager(context)
recyclerView.setLayoutManager(mLayoutManager)

recyclerAdapter= DetailRecyclerAdapter(Details())
recyclerView.adapter=recyclerAdapter;
recyclerAdapter.setListner(this)
```

4. View Pager

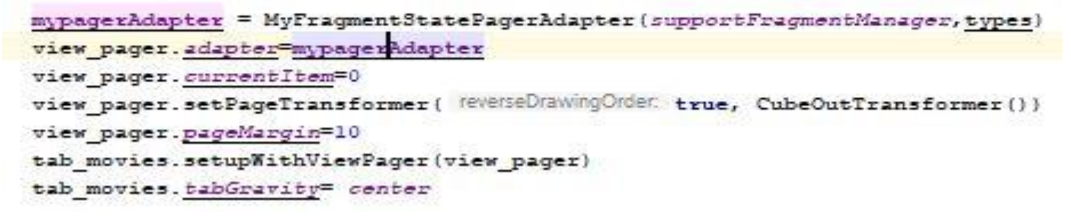
- Flip the pages by recycling the holder of the page layout.
- Page adapter is used to fill in the data for the holder provided by the View Pager.
- Two adapters are provided for the view Pager. They are FragmentPagerAdapter and FragmentStatePagerAdapter.
- Views which are annotated with the View Pager. Décor View annotations are treated as part of the view pagers 'decor'. Each decor view's position can be controlled via its android:layout_gravity attribute
- Depending on build SDK we can set the decor values.
- View Pager has been used to switch between Stocks and Crypto-currencies in the market tab.



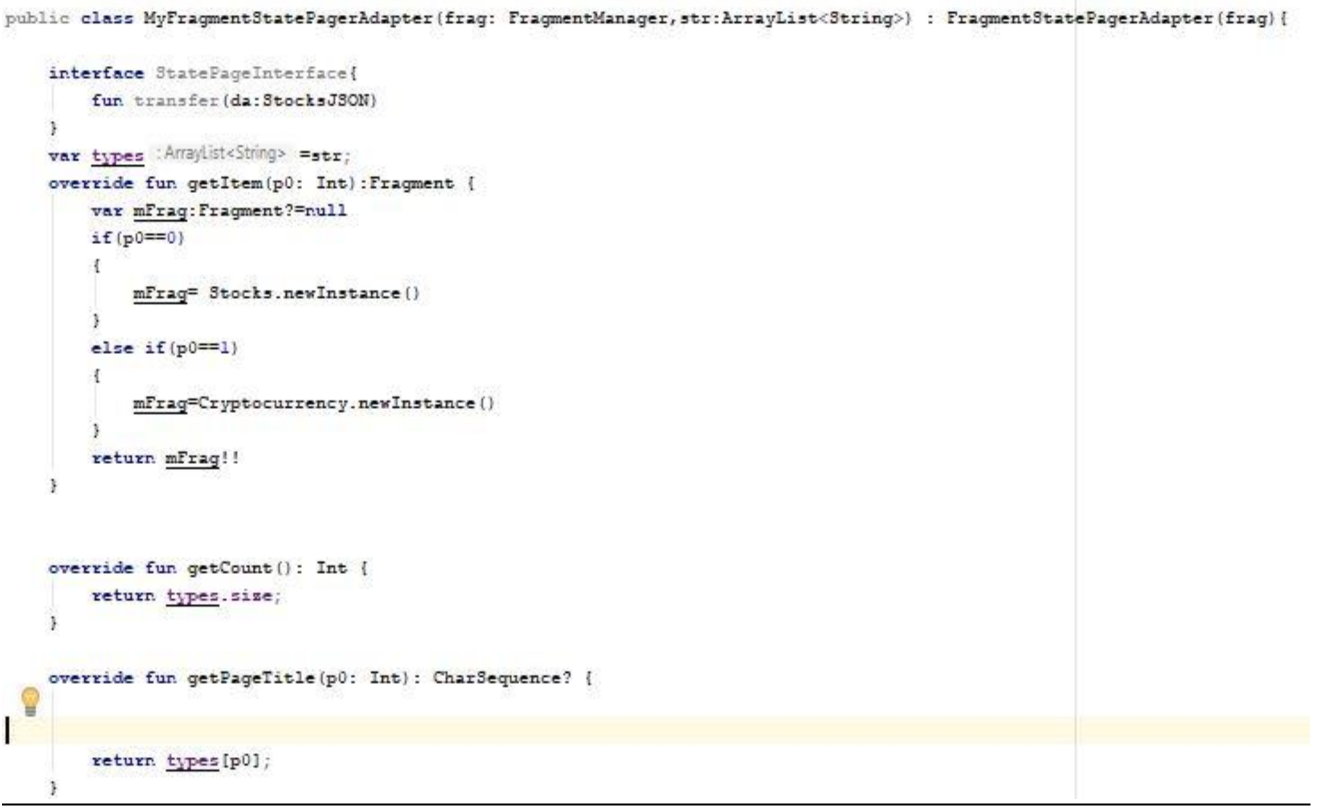
MICROSOFT	103.33
FACEBOOK	135.08
APPLE	176.82
GOOGLE	1046.35
AMAZON	1519.23

Bitcoin	4455.88
Ripple	0.43
Litecoin	33.21
Bitcoin-Cash	255.27
Zcash	82.97

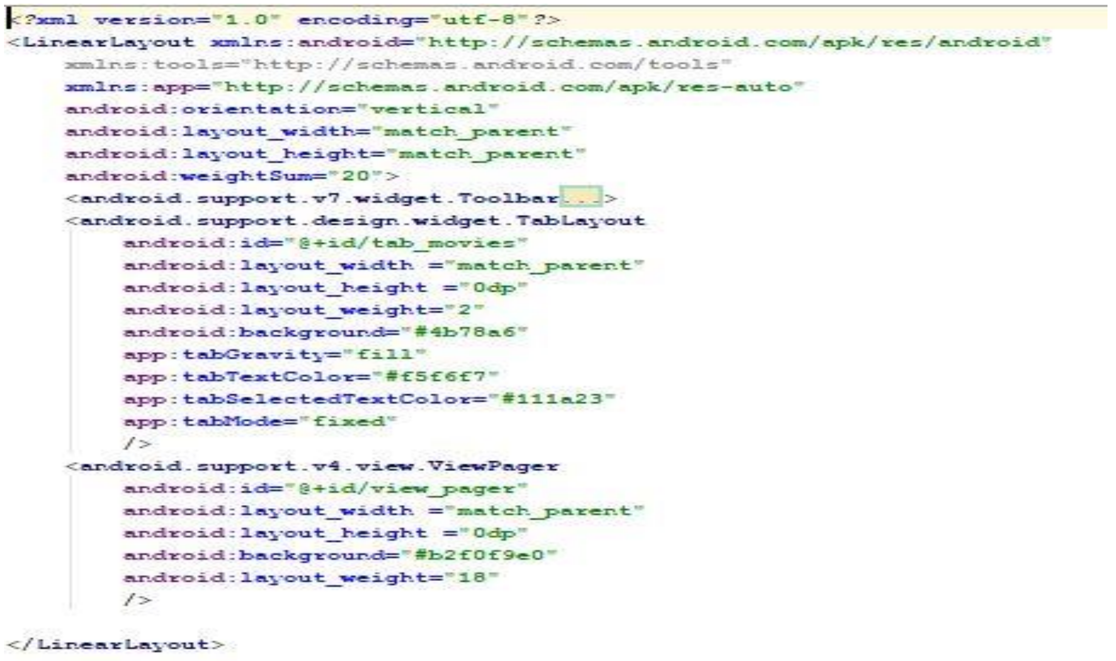
View Pager with Transition Effects



```
mypagerAdapter = MyFragmentStatePagerAdapter(supportFragmentManager,types)
view_pager.adapter=mypagerAdapter
view_pager.currentItem=0
view_pager.setPageTransformer( reverseDrawingOrder: true, CubeOutTransformer())
view_pager.pageMargin=10
tab_movies.setupWithViewPager(view_pager)
tab_movies.tabGravity= center
```



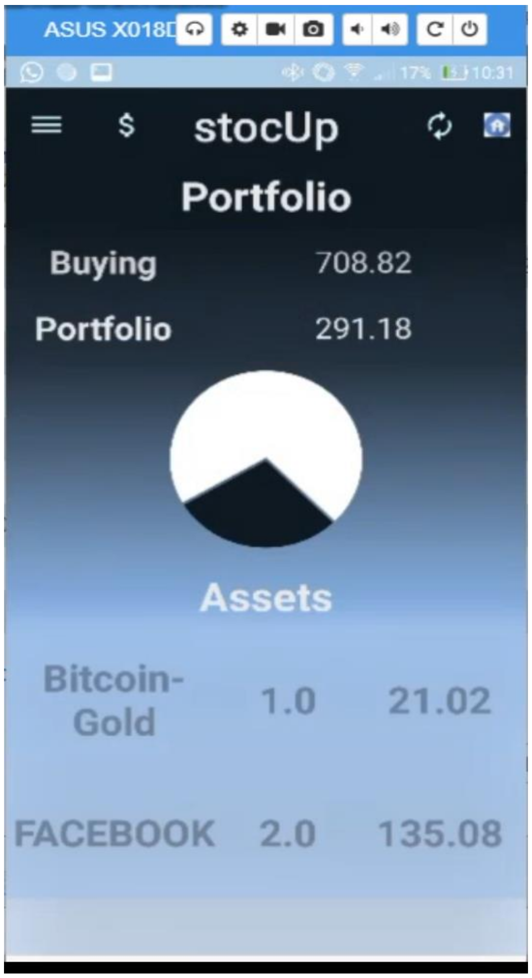
```
public class MyFragmentStatePagerAdapter(frag: FragmentManager,str:ArrayList<String>) : FragmentStatePagerAdapter(frag){

    interface StatePageInterface{
        fun transfer(da:StocksJSON)
    }
    var types : ArrayList<String> =str;
    override fun getItem(p0: Int):Fragment {
        var mFrag:Fragment?=null
        if(p0==0)
        {
            mFrag= Stocks.newInstance()
        }
        else if(p0==1)
        {
            mFrag=Cryptocurrency.newInstance()
        }
        return mFrag!!
    }


    override fun getCount(): Int {
        return types.size;
    }

    override fun getPageTitle(p0: Int): CharSequence? {


        return types[p0];
    }
```

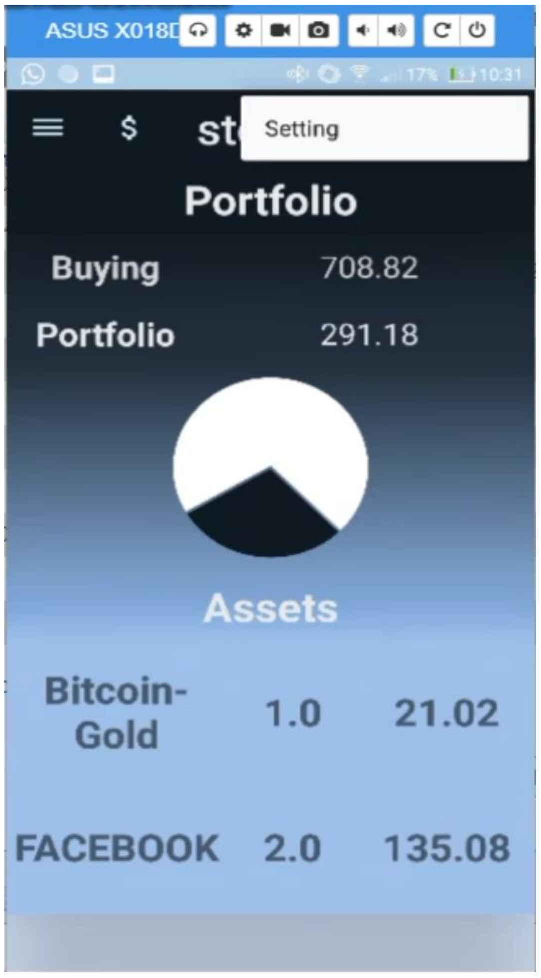


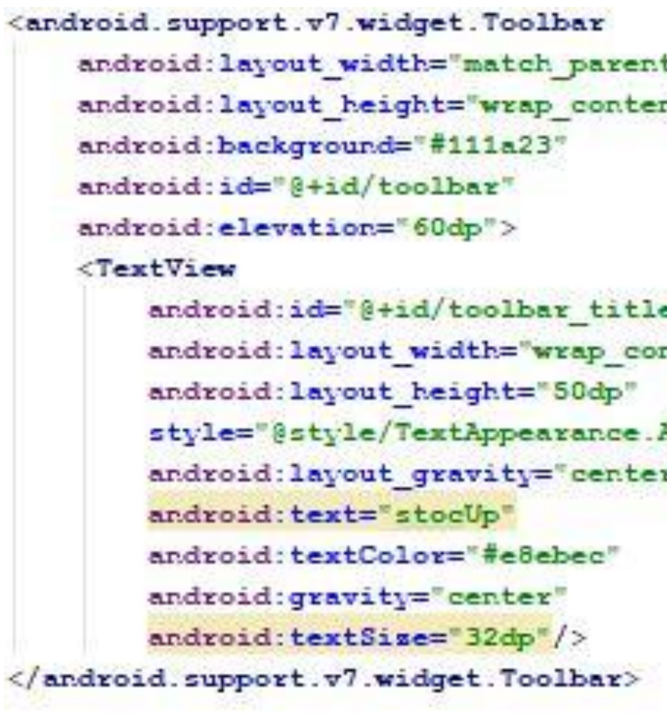
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:weightSum="20">
    <android.support.v7.widget.Toolbar...>
    <android.support.design.widget.TabLayout
        android:id="@+id/tab_movies"
        android:layout_width ="match_parent"
        android:layout_height ="0dp"
        android:layout_weight="2"
        android:background="#4b78a6"
        app:tabGravity="fill"
        app:tabTextColor="#f5f6f7"
        app:tabSelectedTextColor="#111a23"
        app:tabMode="fixed"
        />
    <android.support.v4.view.ViewPager
        android:id="@+id/view_pager"
        android:layout_width ="match_parent"
        android:layout_height ="0dp"
        android:background="#b2f0f9e0"
        android:layout_weight="18"
        />

</LinearLayout>
```

5. Customized Action Bar

- Inserted multiple things on action bar to have an easy of navigation for the user.
- The action bar on the main window is added with multiple buttons to provide features like Refresh the state, shortcut to go to settings directly and a button to open the navigation drawer.



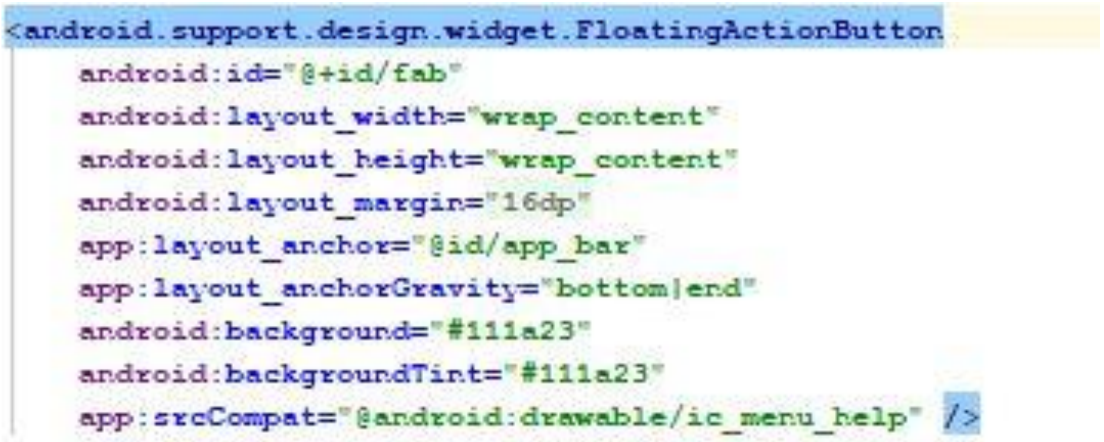




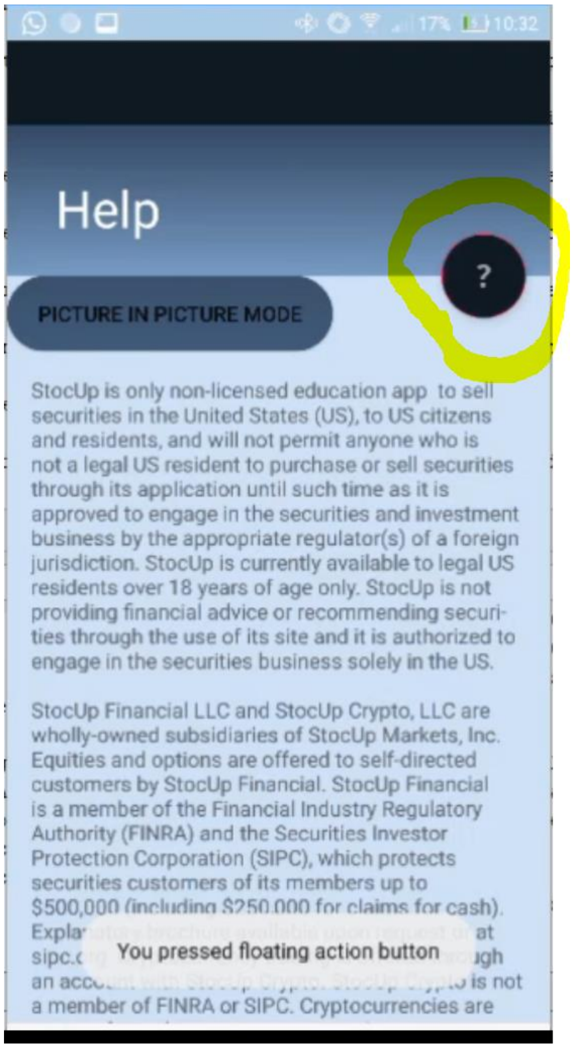
```
<android.support.v7.widget.Toolbar
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:background="#111a23"
    android:id="@+id/toolbar"
    android:elevation="60dp">
    <TextView
        android:id="@+id/toolbar_title"
        android:layout_width="wrap_content"
        android:layout_height="50dp"
        style="@style/TextAppearance.AppCompat.Widget.ActionBar.Title"
        android:layout_gravity="center"
        android:text="stocUp"
        android:textColor="#e8ebec"
        android:gravity="center"
        android:textSize="32dp"/>
</android.support.v7.widget.Toolbar>
```

6. Floating Action Button

- Floating action buttons are button that can provide handy features to maneuver through the application.
- We see such use of a button in the messaging app to write a new message.
- We have implemented a floating action button on our help tab. The floating action button is used to display a toast message describing the current role of the button.



```
<android.support.design.widget.FloatingActionButton
    android:id="@+id/fab"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_margin="16dp"
    app:layout_anchor="@id/app_bar"
    app:layout_anchorGravity="bottom|end"
    android:background="#111a23"
    android:backgroundTint="#111a23"
    app:srcCompat="@android:drawable/ic_menu_help" />
```



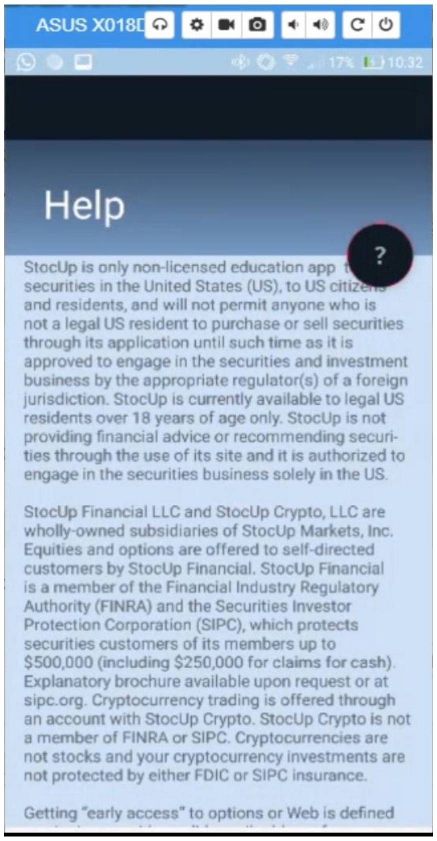
StocUp is only non-licensed education app to sell securities in the United States (US), to US citizens and residents, and will not permit anyone who is not a legal US resident to purchase or sell securities through its application until such time as it is approved to engage in the securities and investment business by the appropriate regulator(s) of a foreign jurisdiction. StocUp is currently available to legal US residents over 18 years of age only. StocUp is not providing financial advice or recommending securi-ties through the use of its site and it is authorized to engage in the securities business solely in the US.

StocUp Financial LLC and StocUp Crypto, LLC are wholly-owned subsidiaries of StocUp Markets, Inc. Equities and options are offered to self-directed customers by StocUp Financial. StocUp Financial is a member of the Financial Industry Regulatory Authority (FINRA) and the Securities Investor Protection Corporation (SIPC), which protects securities customers of its members up to $500,000 (including $250,000 for claims for cash). Expla at sipc.c ugh an acc is not a member of FINRA or SIPC. Cryptocurrencies are

You pressed floating action button

7. Collapsing Toolbar with Coordinator Layout

- We have used coordinator layout in the User help page.
- We have used collapsing toolbar layout inside this coordinator layout . The help toolbar occupies more space in the page initially but will contract or expand Because we used collapsing toolbar layout.
- The other frame of the coordinator layout is filled with a nested scroll view fragment that contains the help information data for the user.

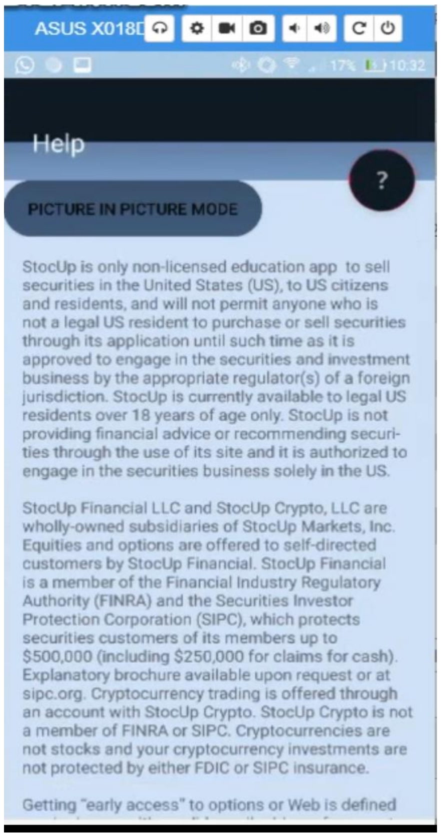


Help

StocUp is only non-licensed education app t securities in the United States (US), to US citizens and residents, and will not permit anyone who is not a legal US resident to purchase or sell securities through its application until such time as it is approved to engage in the securities and investment business by the appropriate regulator(s) of a foreign jurisdiction. StocUp is currently available to legal US residents over 18 years of age only. StocUp is not providing financial advice or recommending securities through the use of its site and it is authorized to engage in the securities business solely in the US.

StocUp Financial LLC and StocUp Crypto, LLC are wholly-owned subsidiaries of StocUp Markets, Inc. Equities and options are offered to self-directed customers by StocUp Financial. StocUp Financial is a member of the Financial Industry Regulatory Authority (FINRA) and the Securities Investor Protection Corporation (SIPC), which protects securities customers of its members up to $500,000 (including $250,000 for claims for cash). Explanatory brochure available upon request or at sipc.org. Cryptocurrency trading is offered through an account with StocUp Crypto. StocUp Crypto is not a member of FINRA or SIPC. Cryptocurrencies are not stocks and your cryptocurrency investments are not protected by either FDIC or SIPC insurance.

Getting "early access" to options or Web is defined

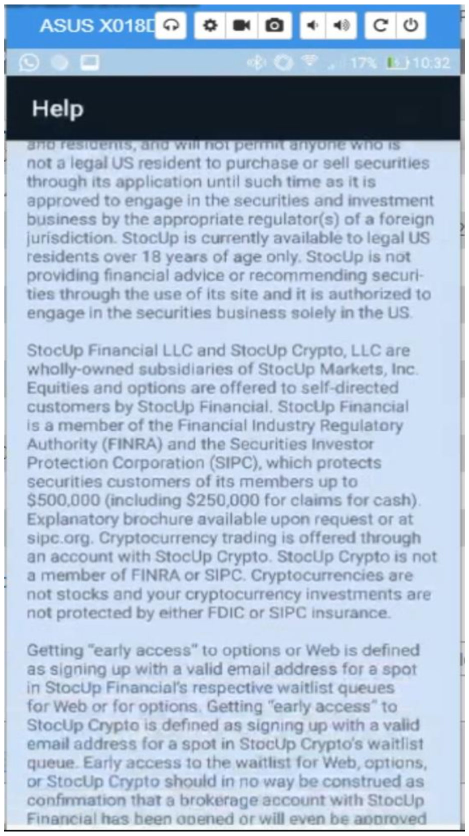


Help

StocUp is only non-licensed education app to sell securities in the United States (US), to US citizens and residents, and will not permit anyone who is not a legal US resident to purchase or sell securities through its application until such time as it is approved to engage in the securities and investment business by the appropriate regulator(s) of a foreign jurisdiction. StocUp is currently available to legal US residents over 18 years of age only. StocUp is not providing financial advice or recommending securities through the use of its site and it is authorized to engage in the securities business solely in the US.

StocUp Financial LLC and StocUp Crypto, LLC are wholly-owned subsidiaries of StocUp Markets, Inc. Equities and options are offered to self-directed customers by StocUp Financial. StocUp Financial is a member of the Financial Industry Regulatory Authority (FINRA) and the Securities Investor Protection Corporation (SIPC), which protects securities customers of its members up to $500,000 (including $250,000 for claims for cash). Explanatory brochure available upon request or at sipc.org. Cryptocurrency trading is offered through an account with StocUp Crypto. StocUp Crypto is not a member of FINRA or SIPC. Cryptocurrencies are not stocks and your cryptocurrency investments are not protected by either FDIC or SIPC insurance.

Getting "early access" to options or Web is defined

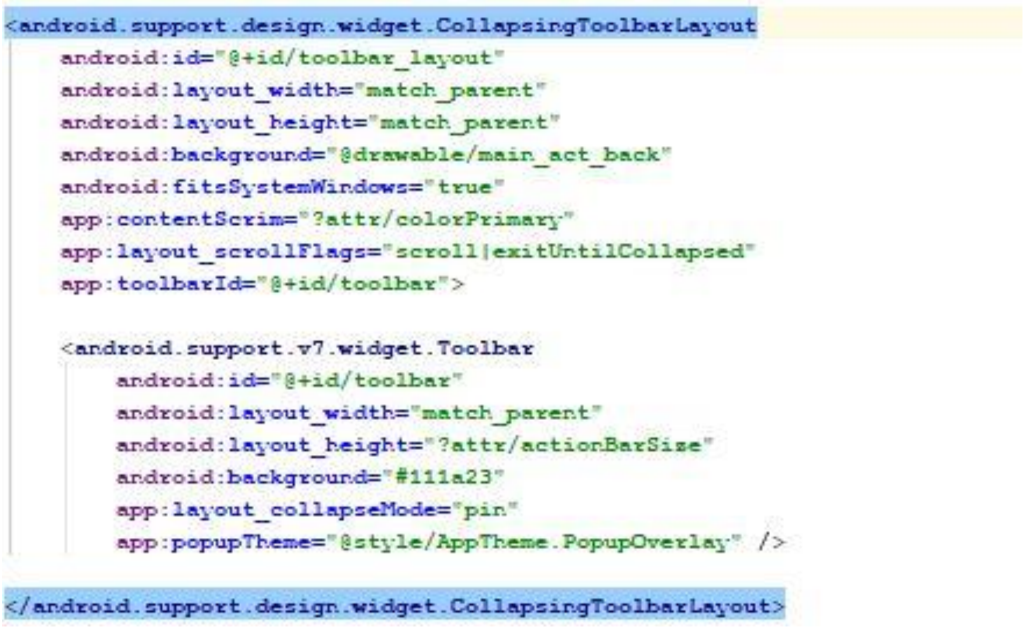


Help

and residents, and will not permit anyone who is not a legal US resident to purchase or sell securities through its application until such time as it is approved to engage in the securities and investment business by the appropriate regulator(s) of a foreign jurisdiction. StocUp is currently available to legal US residents over 18 years of age only. StocUp is not providing financial advice or recommending securities through the use of its site and it is authorized to engage in the securities business solely in the US.

StocUp Financial LLC and StocUp Crypto, LLC are wholly-owned subsidiaries of StocUp Markets, Inc. Equities and options are offered to self-directed customers by StocUp Financial. StocUp Financial is a member of the Financial Industry Regulatory Authority (FINRA) and the Securities Investor Protection Corporation (SIPC), which protects securities customers of its members up to $500,000 (including $250,000 for claims for cash). Explanatory brochure available upon request or at sipc.org. Cryptocurrency trading is offered through an account with StocUp Crypto. StocUp Crypto is not a member of FINRA or SIPC. Cryptocurrencies are not stocks and your cryptocurrency investments are not protected by either FDIC or SIPC insurance.

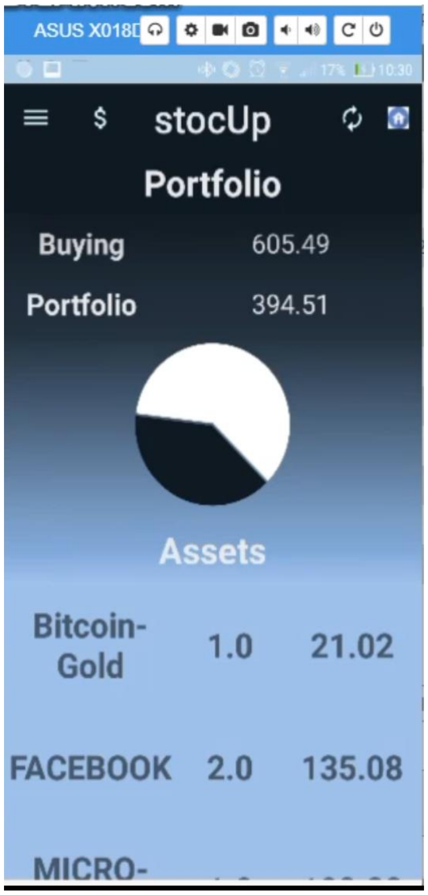
Getting "early access" to options or Web is defined as signing up with a valid email address for a spot in StocUp Financial's respective waitlist queues for Web or for options. Getting "early access" to StocUp Crypto is defined as signing up with a valid email address for a spot in StocUp Crypto's waitlist queue. Early access to the waitlist for Web, options, or StocUp Crypto should in no way be construed as confirmation that a brokerage account with StocUp Financial has been opened or will even be approved

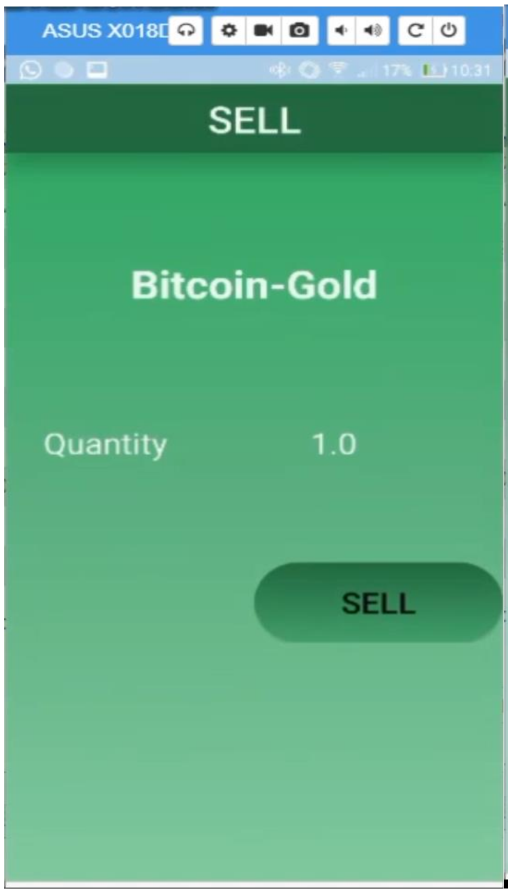


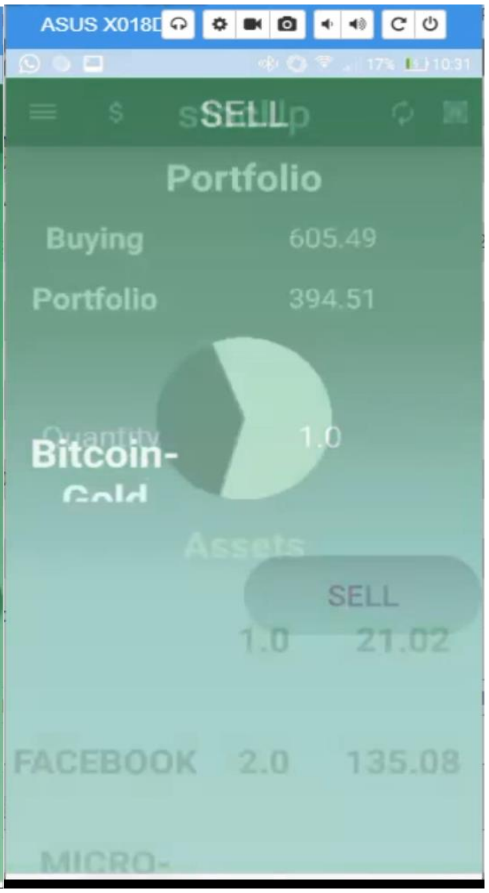
```
<android.support.design.widget.CollapsingToolbarLayout
    android:id="@+id/toolbar_layout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/main_act_back"
    android:fitsSystemWindows="true"
    app:contentScrim="?attr/colorPrimary"
    app:layout_scrollFlags="scroll|exitUntilCollapsed"
    app:toolbarId="@+id/toolbar">

    <android.support.v7.widget.Toolbar
        android:id="@+id/toolbar"
        android:layout_width="match_parent"
        android:layout_height="?attr/actionBarSize"
        android:background="#111a23"
        app:layout_collapseMode="pin"
        app:popupTheme="@style/AppTheme.PopupOverlay" />

</android.support.design.widget.CollapsingToolbarLayout>
```

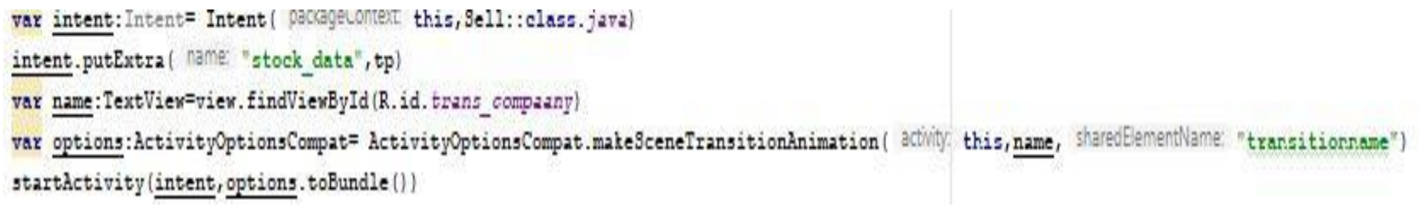
8. Multiple Fragments and Activities(using shared element)

- We have used fragments to show the content at many instances.
- One of the instance that uses shared element is when we display the home page and the user selects any of the assets purchased.
- The details are already in the page. We use the same element to populate the data for the next fragment which is specific to the current asset.

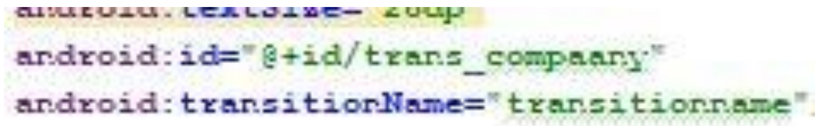




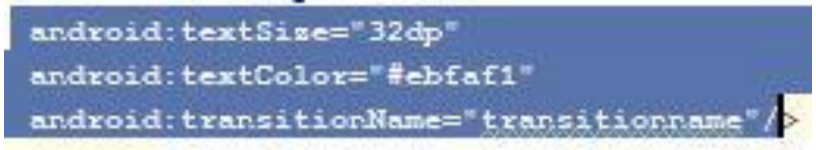




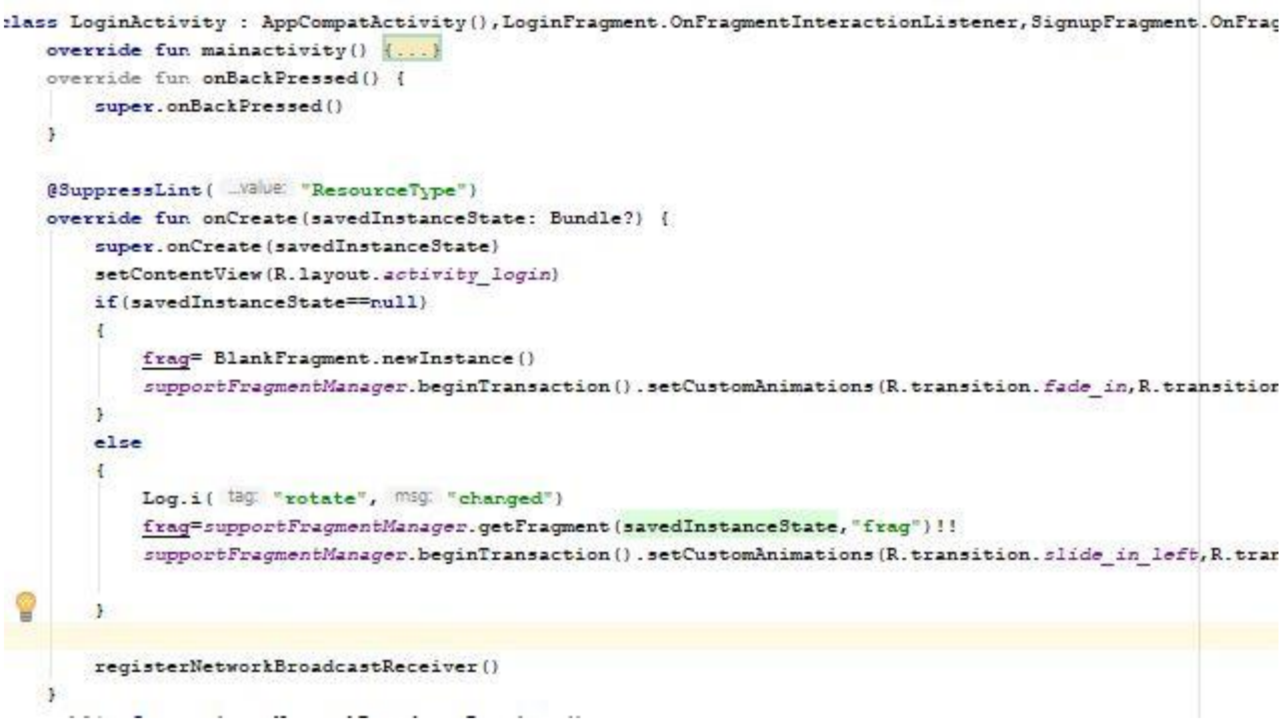
```
var intent:Intent= Intent( packageContext: this,Sell::class.java)
intent.putExtra( name: "stock_data",tp)
var name:TextView=view.findViewById(R.id.trans_compaany)
var options:ActivityOptionsCompat= ActivityOptionsCompat.makeSceneTransitionAnimation( activity: this,name, sharedElementName: "transitionname")
startActivity(intent,options.toBundle())
```



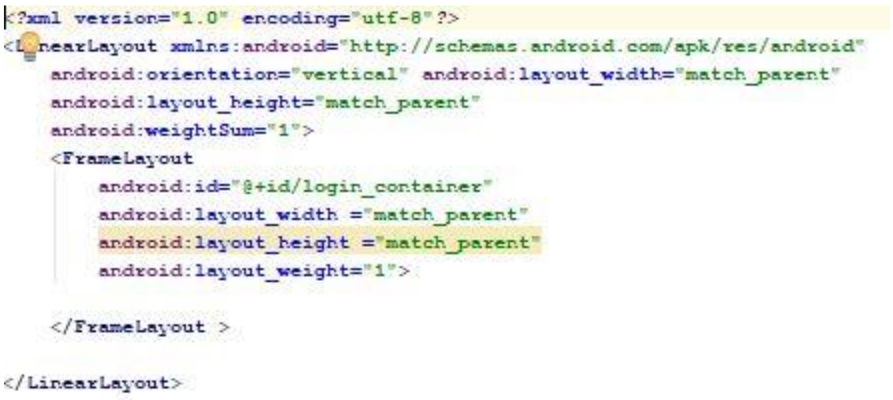
```
android:id="@+id/trans_compaany"
android:transitionName="transitionname".
```



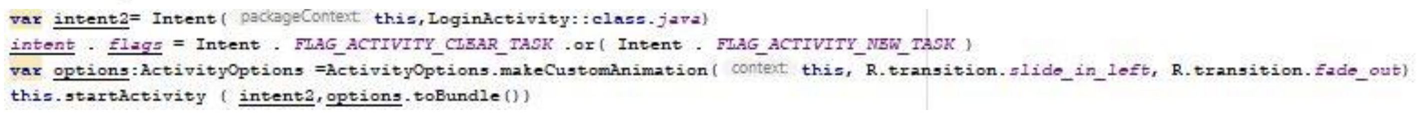
```
android:textSize="32dp"
android:textColor="#ebfaf1"
android:transitionName="transitionname"/>
```



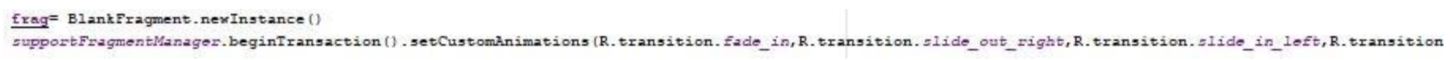
```
class LoginActivity : AppCompatActivity(),LoginFragment.OnFragmentInteractionListener,SignupFragment.OnFrag
    override fun mainactivity() {...}
    override fun onBackPressed() {
        super.onBackPressed()
    }

    @SuppressLint( value: "ResourceType")
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_login)
        if(savedInstanceState==null)
        {
            frag= BlankFragment.newInstance()
            supportFragmentManager.beginTransaction().setCustomAnimations(R.transition.fade_in,R.transitior
        }
        else
        {
            Log.i( tag: "rotate", msg: "changed")
            frag=supportFragmentManager.getFragment(savedInstanceState,"frag")!!
            supportFragmentManager.beginTransaction().setCustomAnimations(R.transition.slide_in_left,R.trar

        }

        registerNetworkBroadcastReceiver()
    }
```



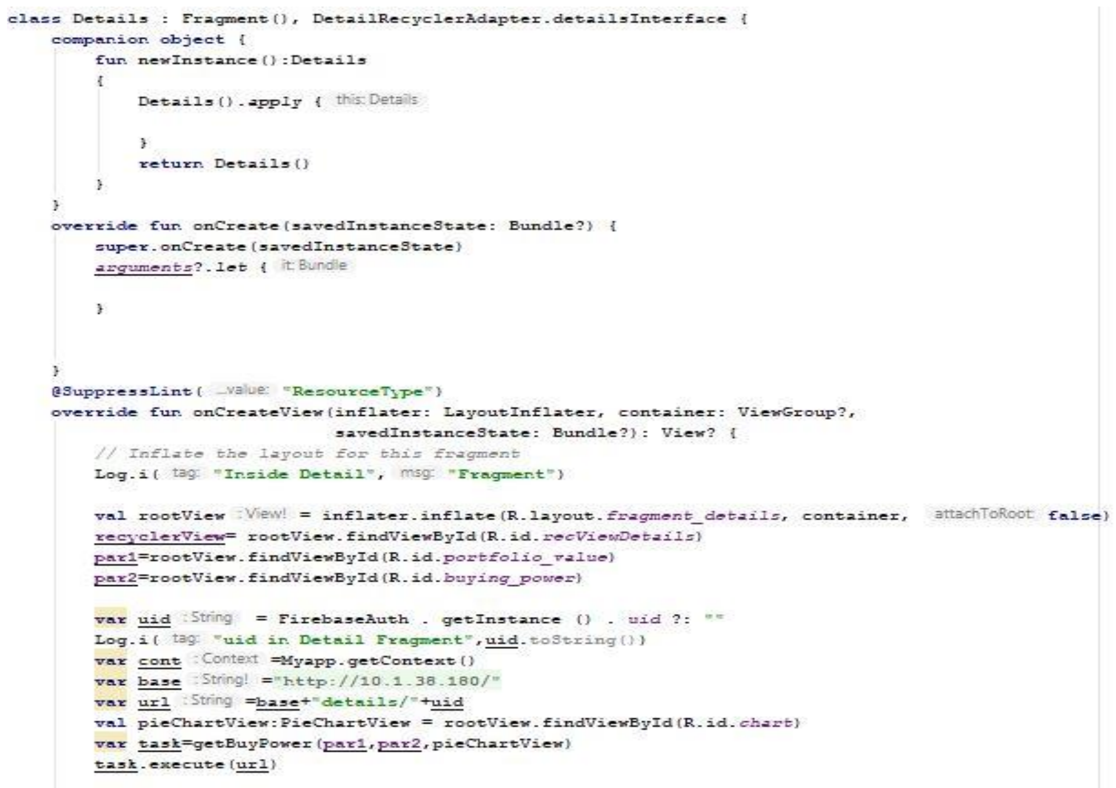
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:weightSum="1">
    <FrameLayout
        android:id="@+id/login_container"
        android:layout_width ="match_parent"
        android:layout_height ="match_parent"
        android:layout_weight="1">

    </FrameLayout >

</LinearLayout>
```



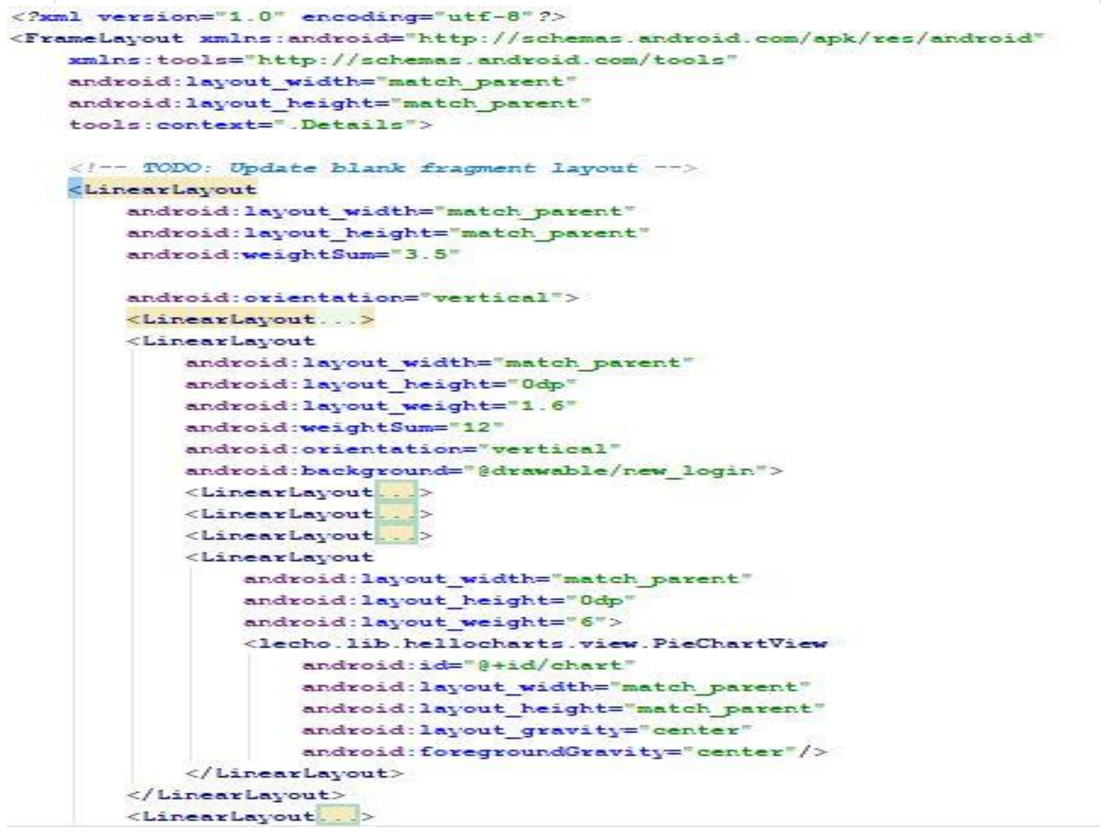
```
var intent2= Intent( packageContext: this,LoginActivity::class.java)
intent . flags = Intent . FLAG_ACTIVITY_CLEAR_TASK .or( Intent . FLAG_ACTIVITY_NEW_TASK )
var options:ActivityOptions =ActivityOptions.makeCustomAnimation( context: this, R.transition.slide_in_left, R.transition.fade_out)
this.startActivity ( intent2,options.toBundle())
```



```
frag= BlankFragment.newInstance()
supportFragmentManager.beginTransaction().setCustomAnimations(R.transition.fade_in,R.transition.slide_out_right,R.transition.slide_in_left,R.transition
```



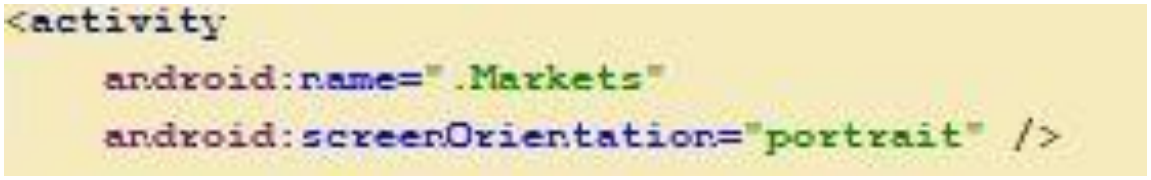
```
class Details : Fragment(), DetailRecyclerAdapter.detailsInterface {
    companion object {
        fun newInstance():Details
        {
            Details().apply {  this: Details

            }
            return Details()
        }
    }
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        arguments?.let {  it: Bundle

        }


    }
    @SuppressLint( ...value: "ResourceType")
    override fun onCreateView(inflater: LayoutInflater, container: ViewGroup?,
                              savedInstanceState: Bundle?): View? {
        // Inflate the layout for this fragment
        Log.i( tag: "Inside Detail", msg: "Fragment")

        val rootView :View! = inflater.inflate(R.layout.fragment_details, container,  attachToRoot: false)
        recyclerView= rootView.findViewById(R.id.recViewDetails)
        par1=rootView.findViewById(R.id.portfolio_value)
        par2=rootView.findViewById(R.id.buying_power)

        var uid :String  = FirebaseAuth . getInstance () . uid ?: ""
        Log.i( tag: "uid in Detail Fragment",uid.toString())
        var cont :Context =Myapp.getContext()
        var base :String! ="http://10.1.38.180/"
        var url :String =base+"details/"+uid
        val pieChartView:PieChartView = rootView.findViewById(R.id.chart)
        var task=getBuyPower(par1,par2,pieChartView)
        task.execute(url)
```



```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout 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"
    tools:context=".Details">

    <!-- TODO: Update blank fragment layout -->
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:weightSum="3.5"

        android:orientation="vertical">
        <LinearLayout...>
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="0dp"
            android:layout_weight="1.6"
            android:weightSum="12"
            android:orientation="vertical"
            android:background="@drawable/new_login">
            <LinearLayout...>
            <LinearLayout...>
            <LinearLayout...>
            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="0dp"
                android:layout_weight="6">
                <lecho.lib.hellocharts.view.PieChartView
                    android:id="@+id/chart"
                    android:layout_width="match_parent"
                    android:layout_height="match_parent"
                    android:layout_gravity="center"
                    android:foregroundGravity="center"/>
            </LinearLayout>
        </LinearLayout>
        <LinearLayout...>
```

9. Multiple Devices/Screens & Orientation Changes

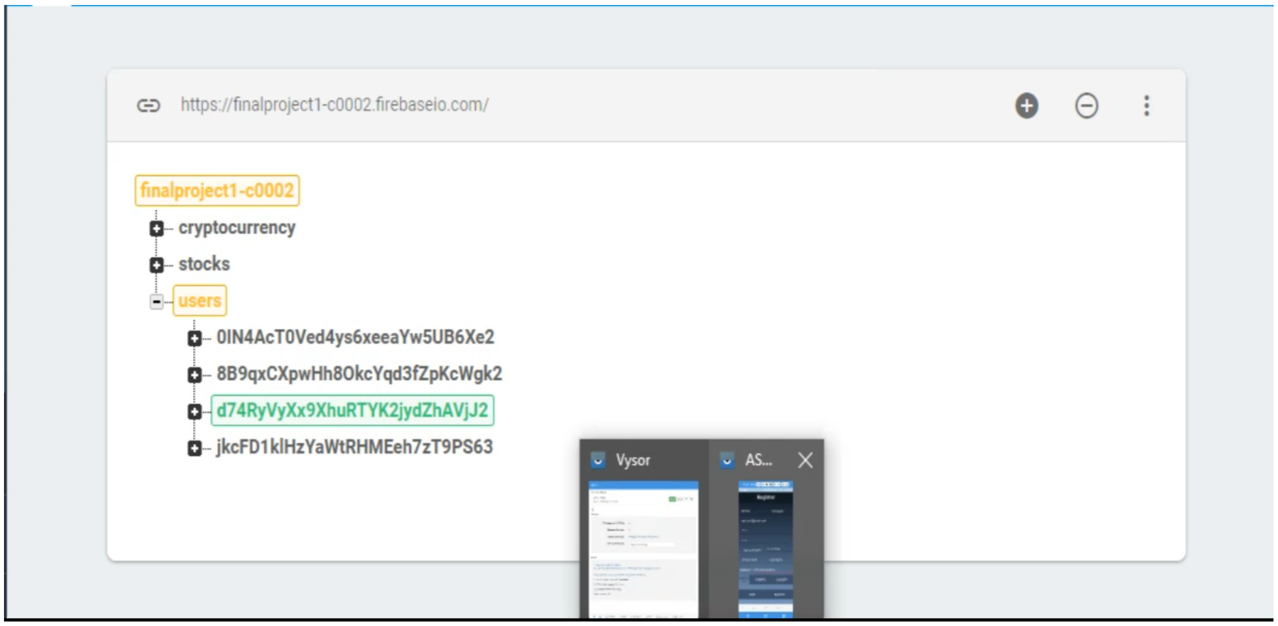
- Will provide only portrait mode to the application to reduce the configuration issues that keep changing from SDK version to SDK version and some issues will be created during runtime.
- To make app work efficiently will lock the app's configuration change to potrait.
- There are two ways to do it :
- setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_POTRAIT);
- android:screenOrientation="portrait"



```
<activity
    android:name=".Markets"
    android:screenOrientation="portrait" />
```

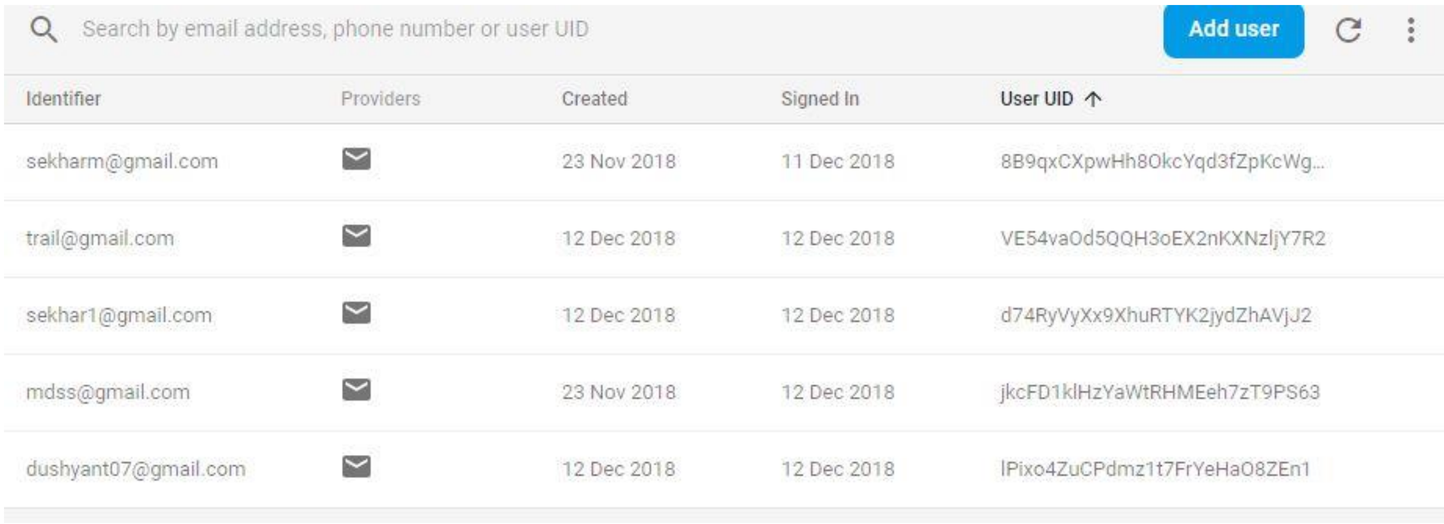
10. Host your data in the Firebase

- Google provides powerful tool for cloud storage, nosql "key-value" based database and user authentication with security rules that helps developer to chase the functionalities of application quickly.
- In this app we used cloud storage to store the profile picture of the user's.
- Used cloud database to store the user details.
- The Firebase Realtime Database is a cloud-hosted database. Data is stored as JSON and synchronized in realtime to every connected client.
- Authentication feature provided by the Firebase with email-password
- Also data for stocks and crypto-currencies has been stored on the cloud.

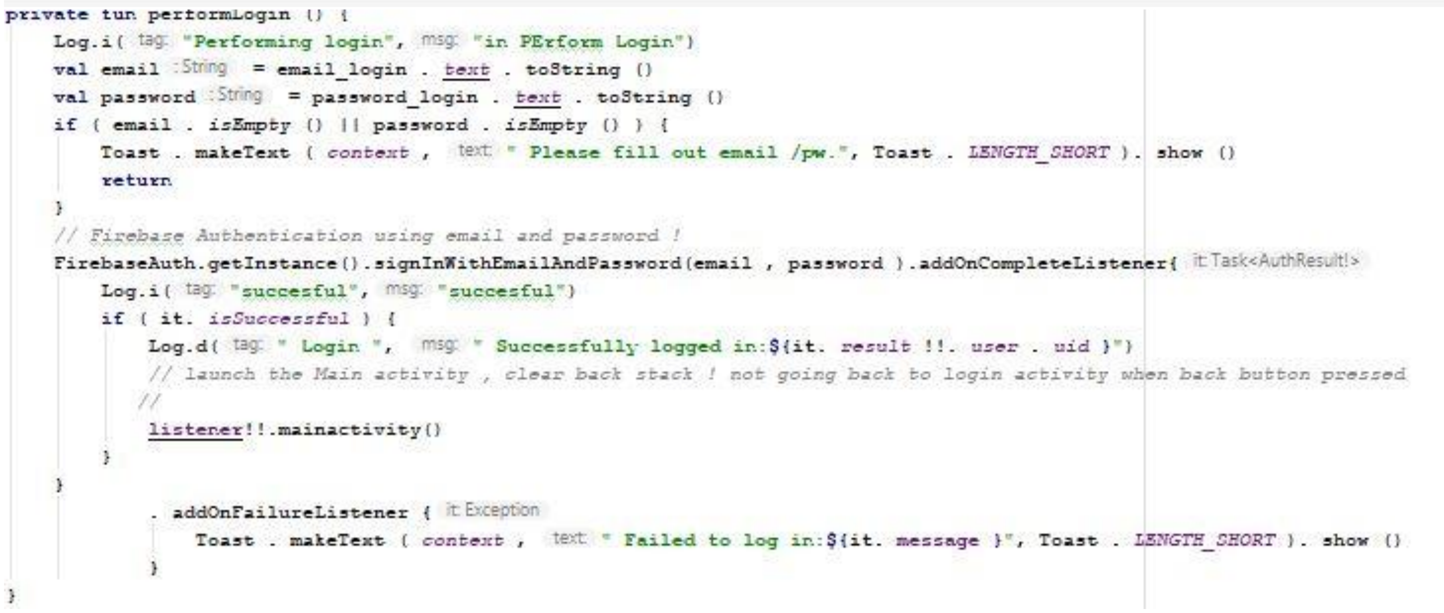


```
https://finalproject1-c0002.firebaseio.com/

finalproject1-c0002
 + cryptocurrency
 + stocks
 - users
    + 0IN4AcT0Ved4ys6xeeaYw5UB6Xe2
    + 8B9qxCXpwHh8OkcYqd3fZpKcWgk2
    + d74RyVyXx9XhuRTYK2jydZhAVjJ2
    + jkcFD1klHzYaWtRHMEeh7zT9PS63
```

New u

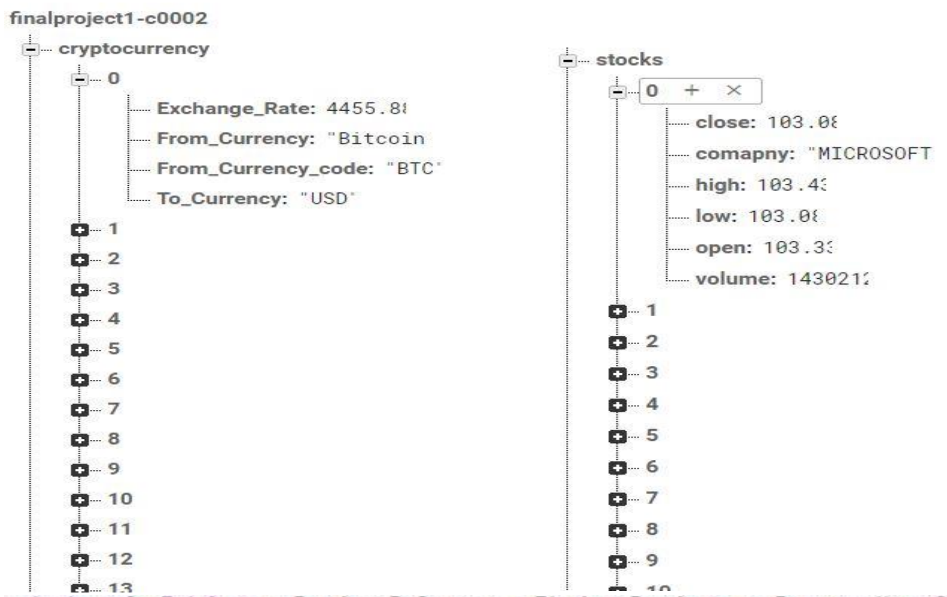


Identifier	Providers	Created	Signed In	User UID ↑
sekharm@gmail.com	✉	23 Nov 2018	11 Dec 2018	8B9qxCXpwHh8OkcYqd3fZpKcWg...
trail@gmail.com	✉	12 Dec 2018	12 Dec 2018	VE54vaOd5QQH3oEX2nKXNzljY7R2
sekhar1@gmail.com	✉	12 Dec 2018	12 Dec 2018	d74RyVyXx9XhuRTYK2jydZhAVjJ2
mdss@gmail.com	✉	23 Nov 2018	12 Dec 2018	jkcFD1klHzYaWtRHMEeh7zT9PS63
dushyant07@gmail.com	✉	12 Dec 2018	12 Dec 2018	lPixo4ZuCPdmz1t7FrYeHaO8ZEn1

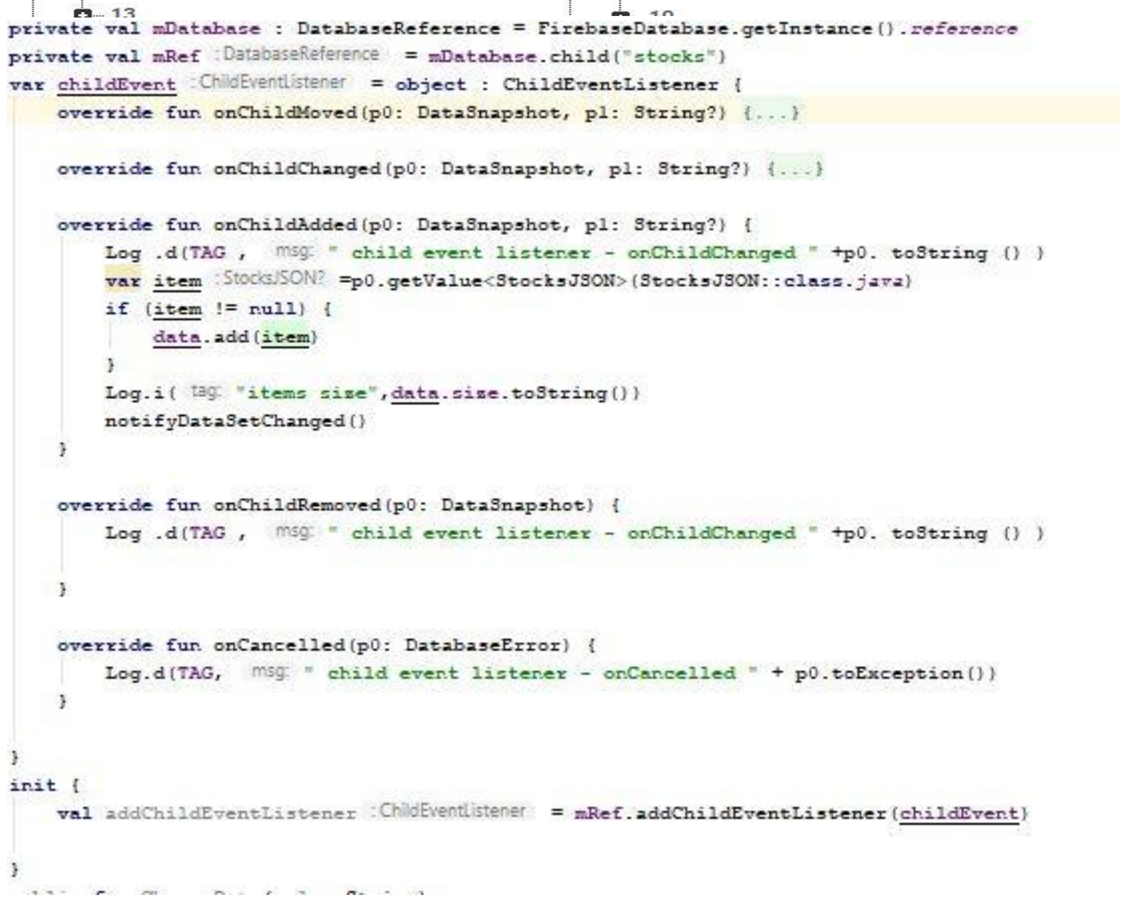


```
private fun performLogin () {
    Log.i( tag: "Performing login", msg: "in PErform Login")
    val email :String  = email_login . text . toString ()
    val password :String  = password_login . text . toString ()
    if ( email . isEmpty () || password . isEmpty () ) {
        Toast . makeText ( context ,  text: " Please fill out email /pw.", Toast . LENGTH_SHORT ). show ()
        return
    }
    // Firebase Authentication using email and password !
    FirebaseAuth.getInstance().signInWithEmailAndPassword(email , password ).addOnCompleteListener{ it: Task<AuthResult!>
        Log.i( tag: "succesful", msg: "succesful")
        if ( it. isSuccessful ) {
            Log.d( tag: " Login ",  msg: " Successfully logged in:${it. result !!. user . uid }")
            // launch the Main activity , clear back stack ! not going back to login activity when back button pressed
            //
            listener!!.mainactivity()
        }
    }
            . addOnFailureListener { it: Exception
                Toast . makeText ( context ,  text: " Failed to log in:${it. message }", Toast . LENGTH_SHORT ). show ()
            }
}
```

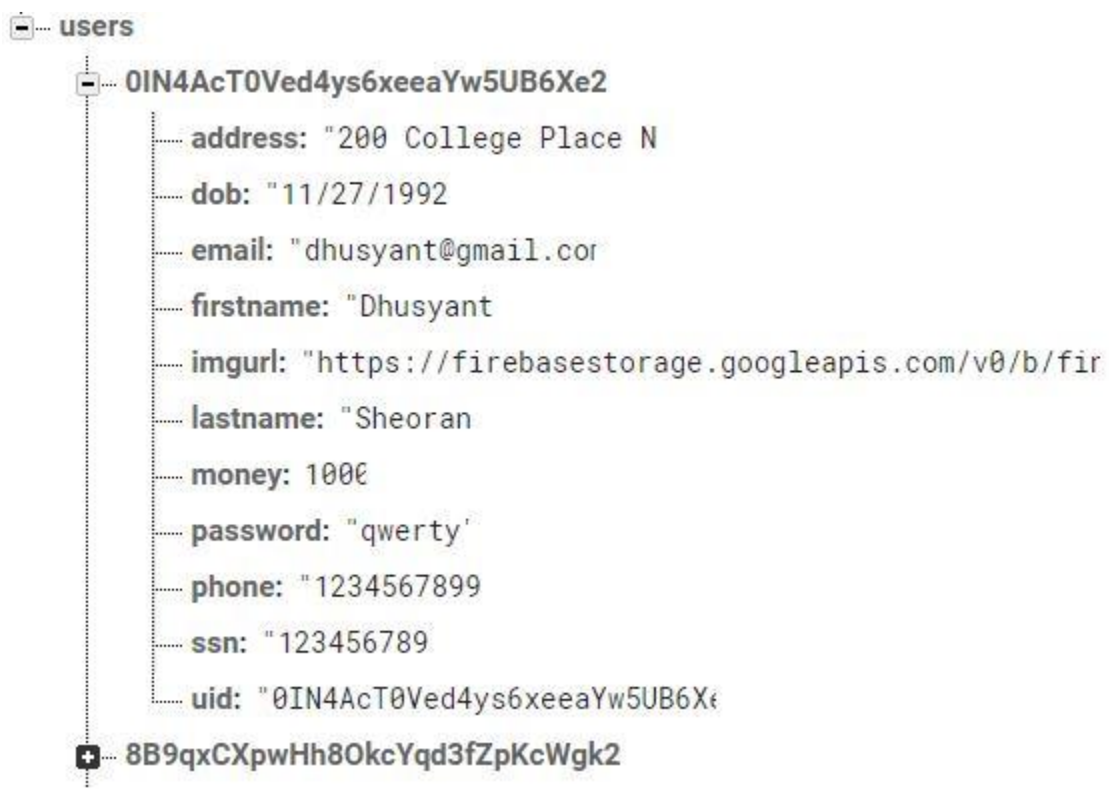
ser Creation in Firebase.



```
finalproject1-c0002
  cryptocurrency
    0
      Exchange_Rate: 4455.8
      From_Currency: "Bitcoin
      From_Currency_code: "BTC"
      To_Currency: "USD"
    1
    2
    3
    4
    5
    6
    7
    8
    9
    10
    11
    12
    13

  stocks
    0  +  ×
      close: 103.0
      comapny: "MICROSOFT
      high: 103.4
      low: 103.0
      open: 103.3
      volume: 143021
    1
    2
    3
    4
    5
    6
    7
    8
    9
```



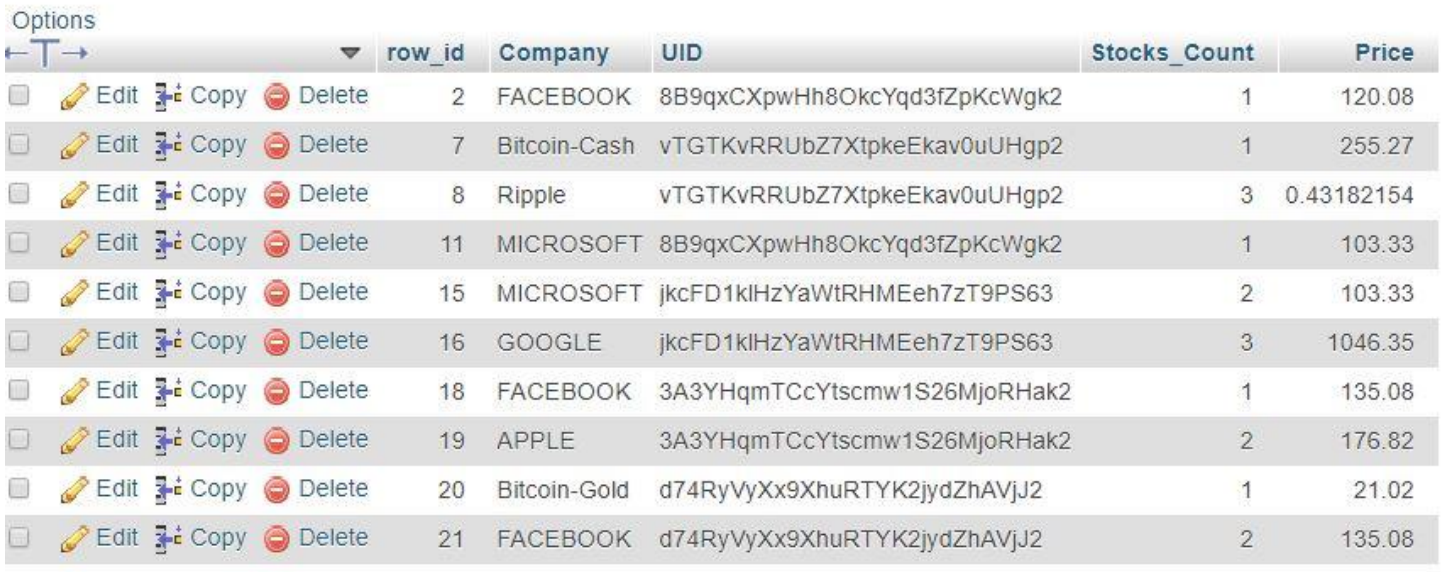
```
private val mDatabase : DatabaseReference = FirebaseDatabase.getInstance().reference
private val mRef :DatabaseReference  = mDatabase.child("stocks")
var childEvent :ChildEventListener  = object : ChildEventListener {
    override fun onChildMoved(p0: DataSnapshot, p1: String?) {...}

    override fun onChildChanged(p0: DataSnapshot, p1: String?) {...}

    override fun onChildAdded(p0: DataSnapshot, p1: String?) {
        Log .d(TAG ,  msg: " child event listener - onChildChanged " +p0. toString () )
        var item :StocksJSON? =p0.getValue<StocksJSON>(StocksJSON::class.java)
        if (item != null) {
            data.add(item)
        }
        Log.i( tag: "items size",data.size.toString())
        notifyDataSetChanged()
    }

    override fun onChildRemoved(p0: DataSnapshot) {
        Log .d(TAG ,  msg: " child event listener - onChildChanged " +p0. toString () )

    }

    override fun onCancelled(p0: DatabaseError) {
        Log.d(TAG,  msg: " child event listener - onCancelled " + p0.toException())
    }

}
init {
    val addChildEventListener :ChildEventListener  = mRef.addChildEventListener(childEvent)

}
```



```
users
  0IN4AcT0Ved4ys6xeeaYw5UB6Xe2
    address: "200 College Place N
    dob: "11/27/1992
    email: "dhusyant@gmail.cor
    firstname: "Dhusyant
    imgurl: "https://firebasestorage.googleapis.com/v0/b/fir
    lastname: "Sheoran
    money: 100€
    password: "qwerty'
    phone: "1234567899
    ssn: "123456789
    uid: "0IN4AcT0Ved4ys6xeeaYw5UB6X€
  8B9qxCXpwHh8OkcYqd3fZpKcWgk2
```

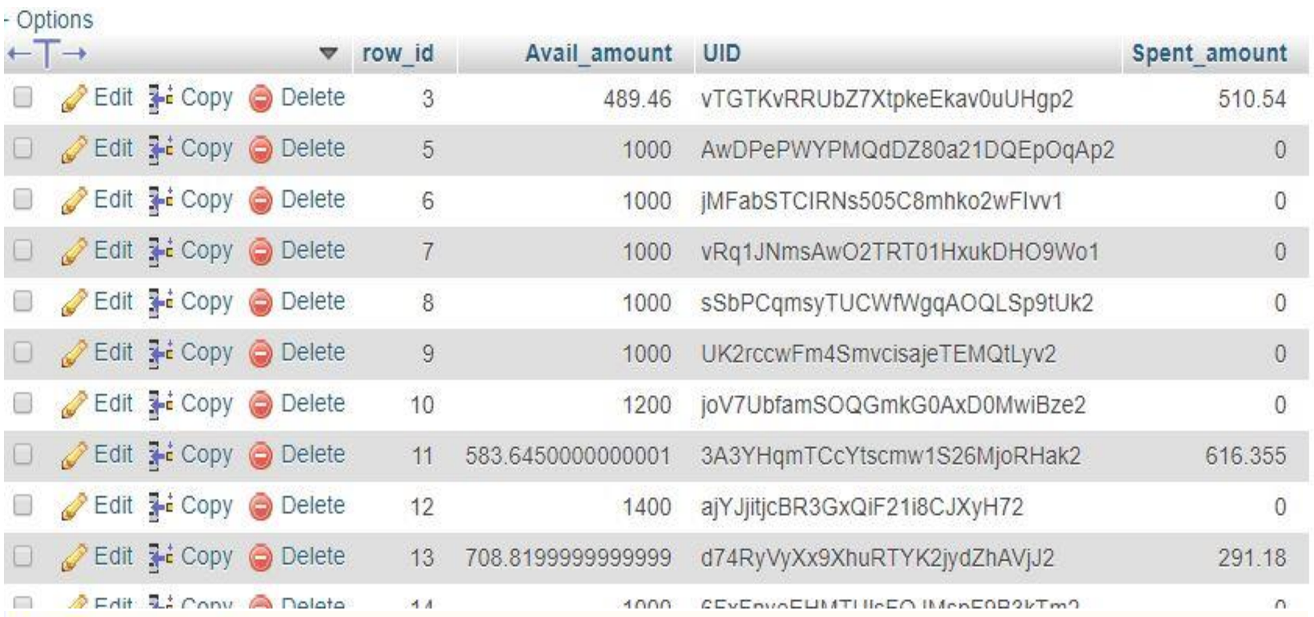
11. Created own host using XAMPP. Used PHP.

- We have implemented our own database server which holds the user specific transaction details.
- Implement using MYSQL and PHP and used XAMPP to create our server.
- Same User-ID as the User-ID randomly created during sign-up process is used to distinguish placed orders.

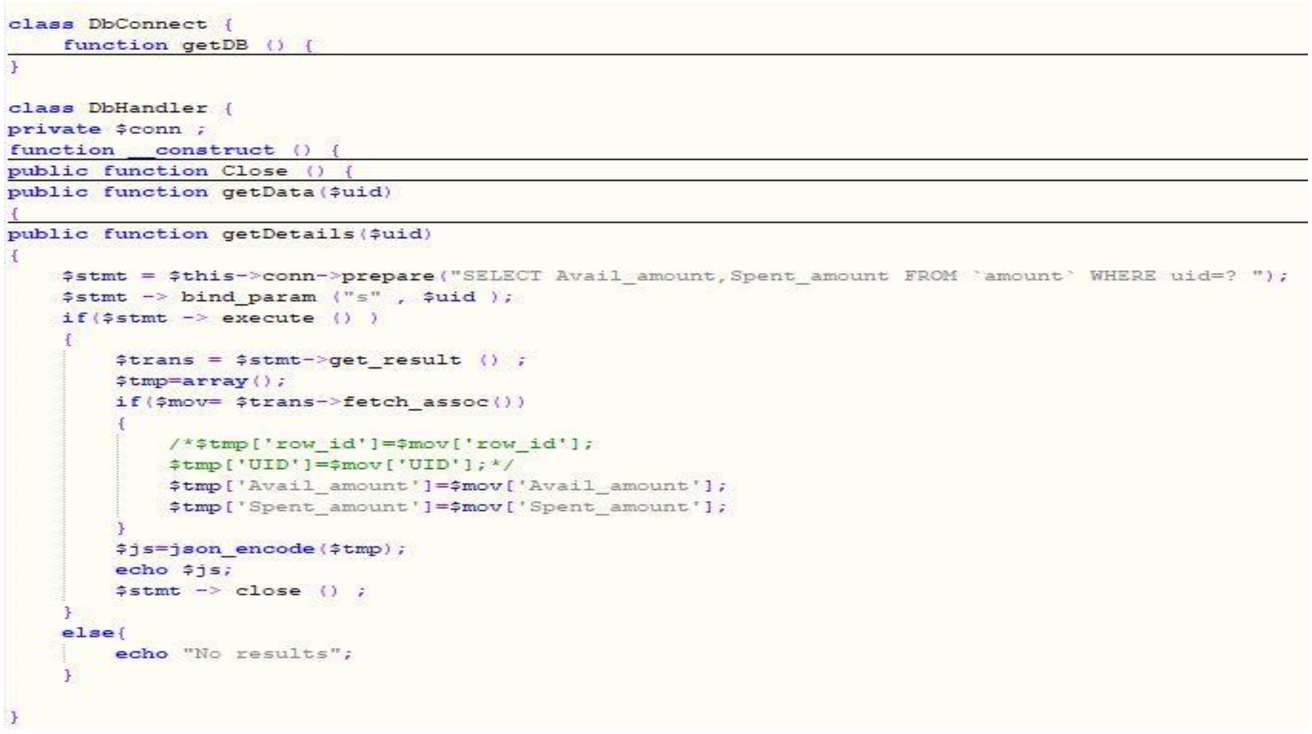


Options

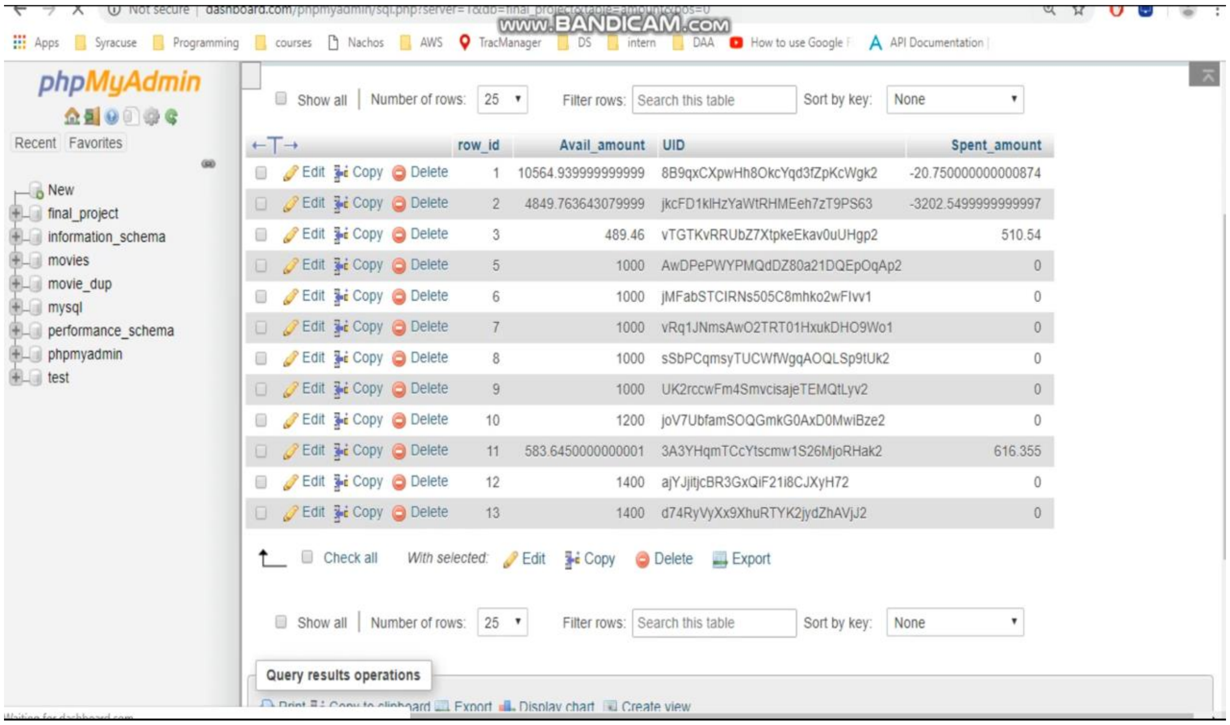
	row_id	Company	UID	Stocks_Count	Price
Edit Copy Delete	2	FACEBOOK	8B9qxCXpwHh8OkcYqd3fZpKcWgk2	1	120.08
Edit Copy Delete	7	Bitcoin-Cash	vTGTKvRRUbZ7XtpkeEkav0uUHgp2	1	255.27
Edit Copy Delete	8	Ripple	vTGTKvRRUbZ7XtpkeEkav0uUHgp2	3	0.43182154
Edit Copy Delete	11	MICROSOFT	8B9qxCXpwHh8OkcYqd3fZpKcWgk2	1	103.33
Edit Copy Delete	15	MICROSOFT	jkcFD1klHzYaWtRHMEeh7zT9PS63	2	103.33
Edit Copy Delete	16	GOOGLE	jkcFD1klHzYaWtRHMEeh7zT9PS63	3	1046.35
Edit Copy Delete	18	FACEBOOK	3A3YHqmTCcYtscmw1S26MjoRHak2	1	135.08
Edit Copy Delete	19	APPLE	3A3YHqmTCcYtscmw1S26MjoRHak2	2	176.82
Edit Copy Delete	20	Bitcoin-Gold	d74RyVyXx9XhuRTYK2jydZhAVjJ2	1	21.02
Edit Copy Delete	21	FACEBOOK	d74RyVyXx9XhuRTYK2jydZhAVjJ2	2	135.08



row_id	Avail_amount	UID	Spent_amount
3	489.46	vTGTKvRRUbZ7XtpkeEkav0uUHgp2	510.54
5	1000	AwDPePWYPMQdDZ80a21DQEpOqAp2	0
6	1000	jMFabSTCIRNs505C8mhko2wFIvv1	0
7	1000	vRq1JNmsAwO2TRT01HxukDHO9Wo1	0
8	1000	sSbPCqmsyTUCWfWgqAOQLSp9tUk2	0
9	1000	UK2rccwFm4SmvcisajeTEMQtLyv2	0
10	1200	joV7UbfamSOQGmkG0AxD0MwiBze2	0
11	583.6450000000001	3A3YHqmTCcYtscmw1S26MjoRHak2	616.355
12	1400	ajYJjitjcBR3GxQiF21i8CJXyH72	0
13	708.8199999999999	d74RyVyXx9XhuRTYK2jydZhAVjJ2	291.18
14	1000	[illegible]	0



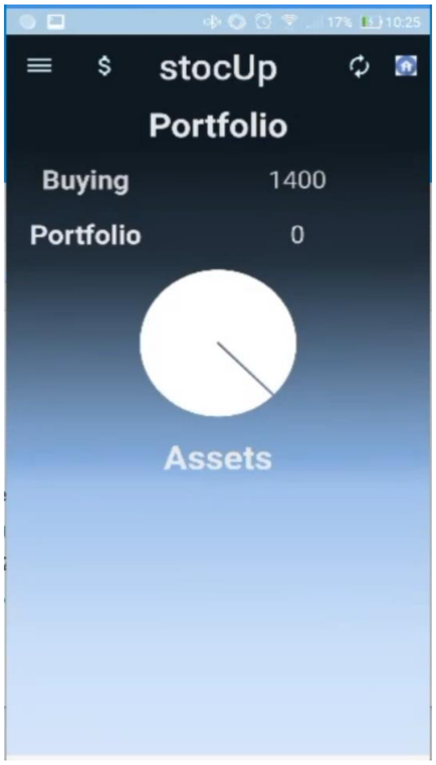
```
class DbConnect {
    function getDB () {
}

class DbHandler {
private $conn ;
function __construct () {
public function Close () {
public function getData($uid)
{
public function getDetails($uid)
{
    $stmt = $this->conn->prepare("SELECT Avail_amount,Spent_amount FROM `amount` WHERE uid=? ");
    $stmt -> bind_param ("s" , $uid );
    if($stmt -> execute () )
    {
        $trans = $stmt->get_result () ;
        $tmp=array();
        if($mov= $trans->fetch_assoc())
        {
            /*$tmp['row_id']=$mov['row_id'];
            $tmp['UID']=$mov['UID'];*/
            $tmp['Avail_amount']=$mov['Avail_amount'];
            $tmp['Spent_amount']=$mov['Spent_amount'];
        }
        $js=json_encode($tmp);
        echo $js;
        $stmt -> close () ;
    }
    else{
        echo "No results";
    }

}
```

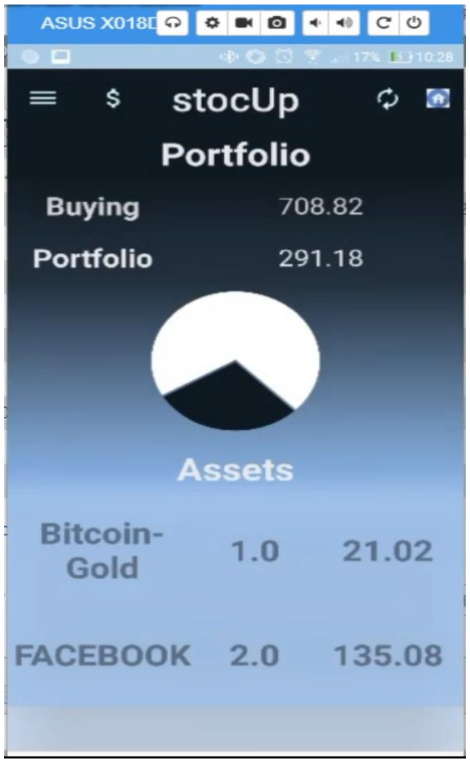


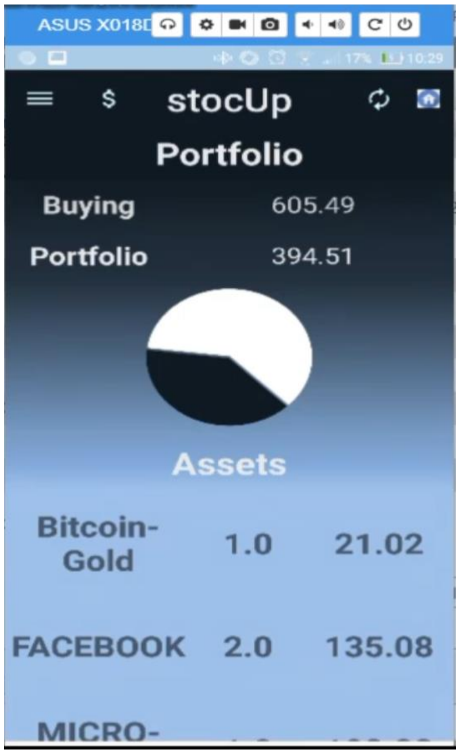
row_id	Avail_amount	UID	Spent_amount
1	10564.939999999999	8B9qxCXpwHh8OkcYqd3fZpKcWgk2	-20.750000000000874
2	4849.763643079999	jkcFD1klHzYaWtRHMEeh7zT9PS63	-3202.5499999999997
3	489.46	vTGTKvRRUbZ7XtpkeEkav0uUHgp2	510.54
5	1000	AwDPePWYPMQdDZ80a21DQEpOqAp2	0
6	1000	jMFabSTCIRNs505C8mhko2wFIvv1	0
7	1000	vRq1JNmsAwO2TRT01HxukDHO9Wo1	0
8	1000	sSbPCqmsyTUCWfWgqAOQLSp9tUk2	0
9	1000	UK2rccwFm4SmvcisajeTEMQtLyv2	0
10	1200	joV7UbfamSOQGmkG0AxD0MwiBze2	0
11	583.6450000000001	3A3YHqmTCcYtscmw1S26MjoRHak2	616.355
12	1400	ajYJjitjcBR3GxQiF21i8CJXyH72	0
13	1400	d74RyVyXx9XhuRTYK2jydZhAVjJ2	0

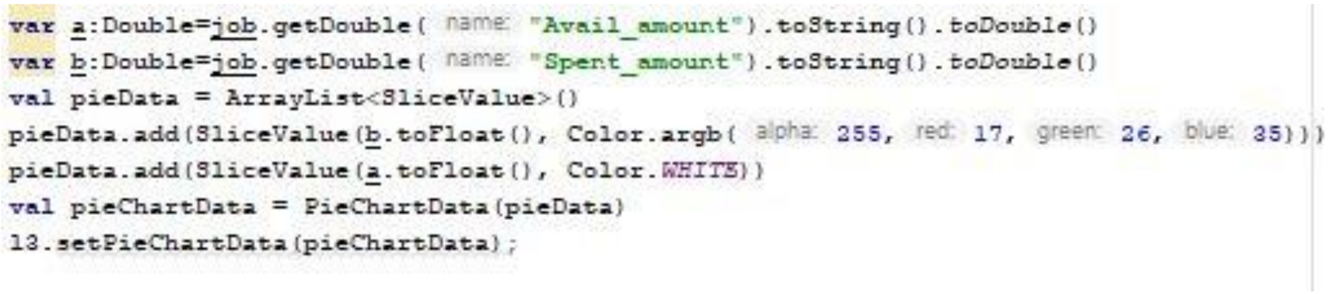
12. Used Custom Views to Display Data Interactively

- It is important for an application in todays world to be enriching and interactive to look at.
- Custom Views are user created view element that give a smart look to plain data.
- We have implemented Custom view on our home screen to display the current balance between the money currently invested by client and the money available.
- The Custom View makes data look refreshing and at the same time easier to understand.



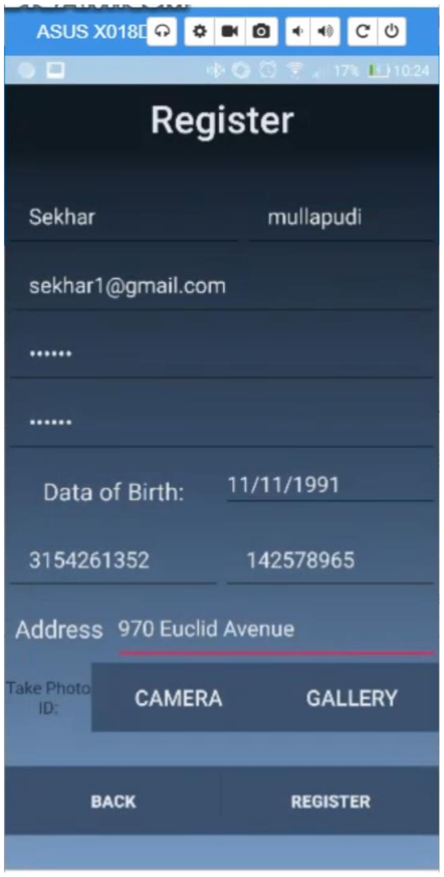





```
var a:Double=job.getDouble( name: "Avail_amount").toString().toDouble()
var b:Double=job.getDouble( name: "Spent_amount").toString().toDouble()
val pieData = ArrayList<SliceValue>()
pieData.add(SliceValue(b.toFloat(), Color.argb( alpha: 255, red: 17, green: 26, blue: 35)))
pieData.add(SliceValue(a.toFloat(), Color.WHITE))
val pieChartData = PieChartData(pieData)
l3.setPieChartData(pieChartData);
```

13. Camera and Gallery

- We have used camera and Gallery to upload profile picture of the user.
- The user on clicking on the floating button in the user profile page, he will get an option to select either "take a photo" or "choose from gallery".
- In "Take a photo" option , if the user is using the camera for first time, the application will ask for permissions to use the camera. Once he takes the picture, the picture will be stored in firebase database.
- In "Choose from library" option, the user can choose the image from gallery and can upload it. The image will store in firebase at the time of account creation itself.
- The tasks are accomplished in the following way:



Register

Sekhar mullapudi

sekhar1@gmail.com

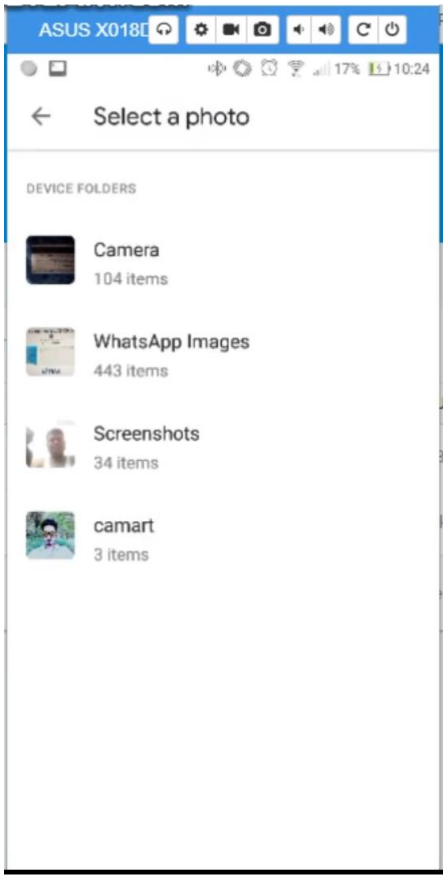
••••••

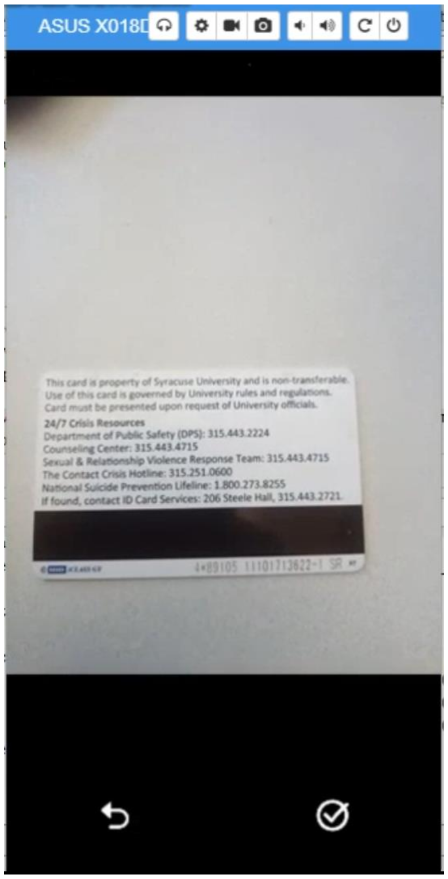
••••••

Data of Birth: 11/11/1991

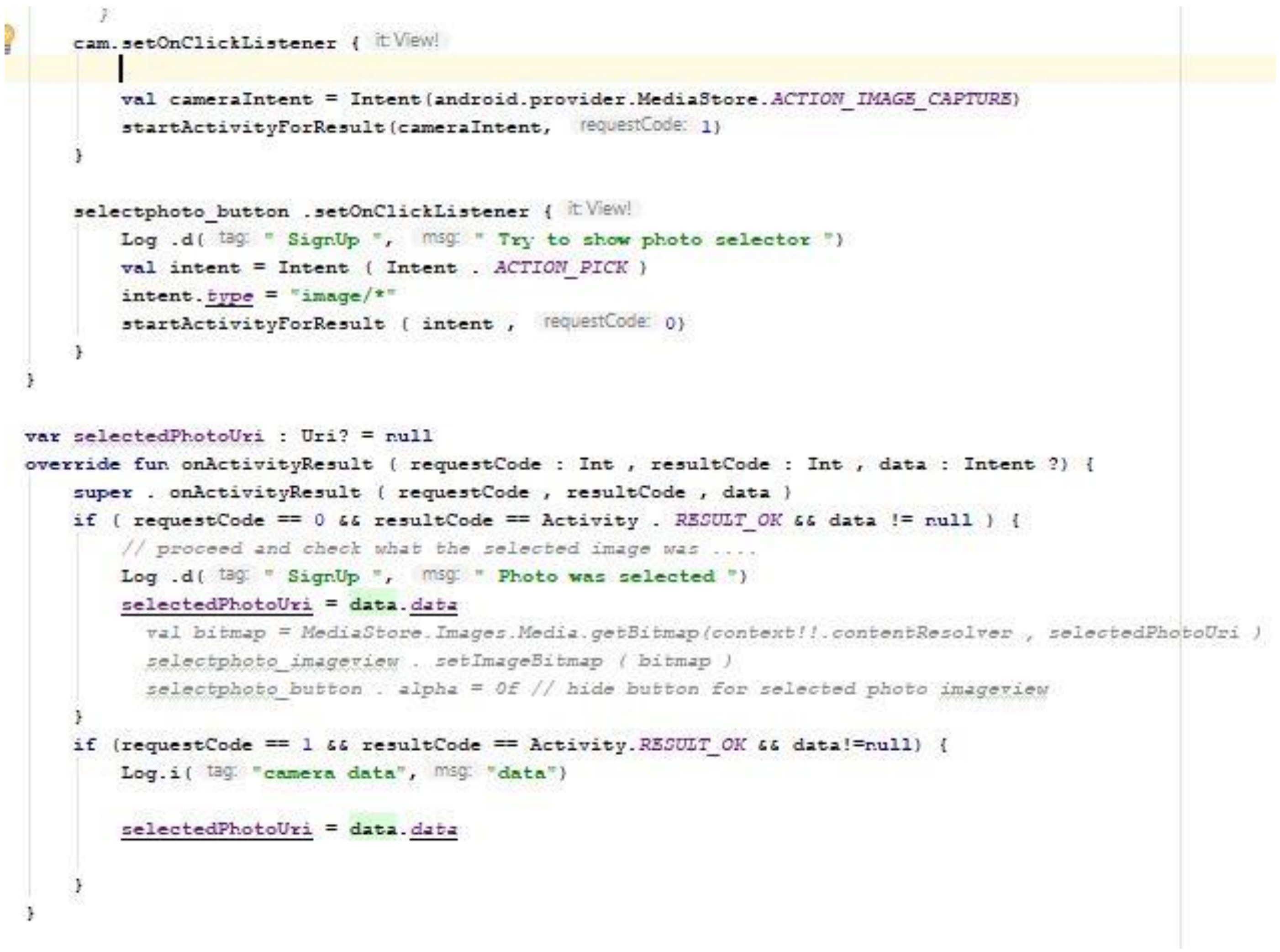
3154261352 142578965

Address 970 Euclid Avenue





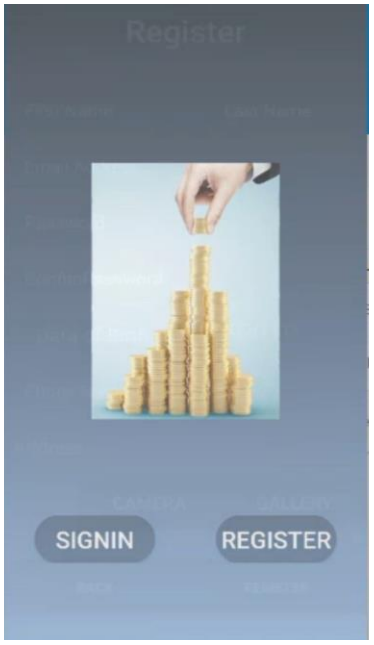
This card is property of Syracuse University and is non-transferable.
Use of this card is governed by University rules and regulations.
Card must be presented upon request of University officials.
24/7 Crisis Resources
Department of Public Safety (DPS): 315.443.2224
Counseling Center: 315.443.4715
Sexual & Relationship Violence Response Team: 315.443.4715
The Contact Crisis Hotline: 315.251.0600
National Suicide Prevention Lifeline: 1.800.273.8255
If found, contact ID Card Services: 206 Steele Hall, 315.443.2721

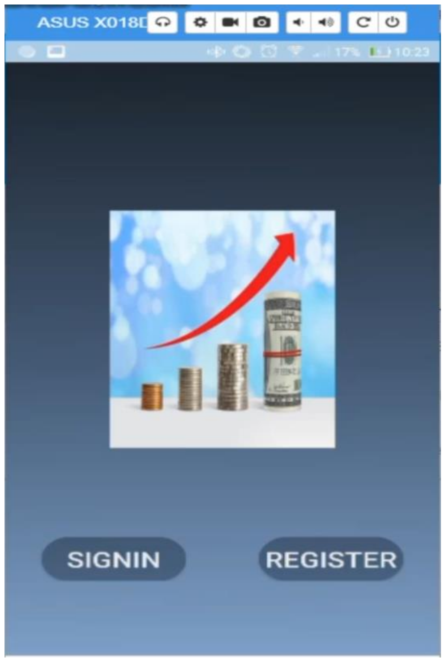


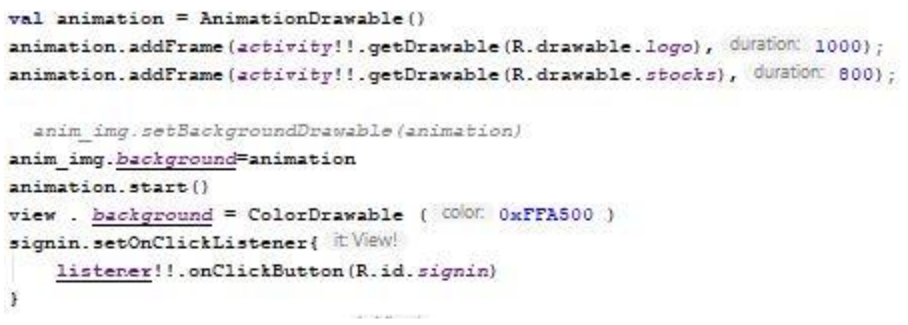
```
        }
    cam.setOnClickListener { it: View!

        val cameraIntent = Intent(android.provider.MediaStore.ACTION_IMAGE_CAPTURE)
        startActivityForResult(cameraIntent, requestCode: 1)
    }

    selectphoto_button .setOnClickListener { it: View!
        Log .d( tag: " SignUp ", msg: " Try to show photo selector ")
        val intent = Intent ( Intent . ACTION_PICK )
        intent.type = "image/*"
        startActivityForResult ( intent , requestCode: 0)
    }
}

var selectedPhotoUri : Uri? = null
override fun onActivityResult ( requestCode : Int , resultCode : Int , data : Intent ?) {
    super . onActivityResult ( requestCode , resultCode , data )
    if ( requestCode == 0 && resultCode == Activity . RESULT_OK && data != null ) {
        // proceed and check what the selected image was ....
        Log .d( tag: " SignUp ", msg: " Photo was selected ")
        selectedPhotoUri = data.data
          val bitmap = MediaStore.Images.Media.getBitmap(context!!.contentResolver , selectedPhotoUri )
          selectphoto_imageview . setImageBitmap ( bitmap )
          selectphoto_button . alpha = 0f // hide button for selected photo imageview
    }
    if (requestCode == 1 && resultCode == Activity.RESULT_OK && data!=null) {
        Log.i( tag: "camera data", msg: "data")

        selectedPhotoUri = data.data

    }
}
```

14. Advanced Animation

- Advanced animation have been used to display smooth transition between fragments and activities.
- One of the best example is the transition between the fragments of the view pager in the market tab.
- Also when user switches from one activity to another, the transition is slow and takes effect using beautiful animations.
- They were done using:





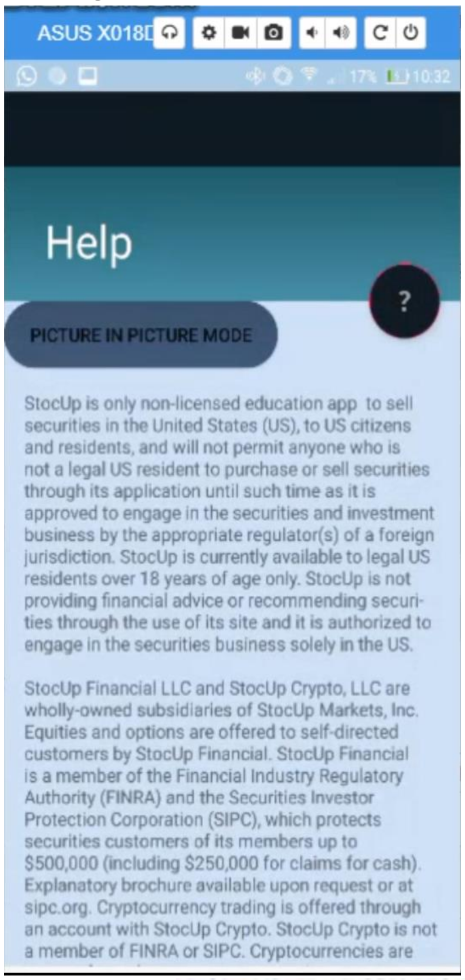


```
val animation = AnimationDrawable()
animation.addFrame(activity!!.getDrawable(R.drawable.logo),  duration: 1000);
animation.addFrame(activity!!.getDrawable(R.drawable.stocks),  duration: 800);

  anim_img.setBackgroundDrawable(animation)
anim_img.background=animation
animation.start()
view . background = ColorDrawable (  color: 0xFFA500 )
signin.setOnClickListener{  it: View!
    listener!!.onClickButton(R.id.signin)
}
```

15. Internal Storage

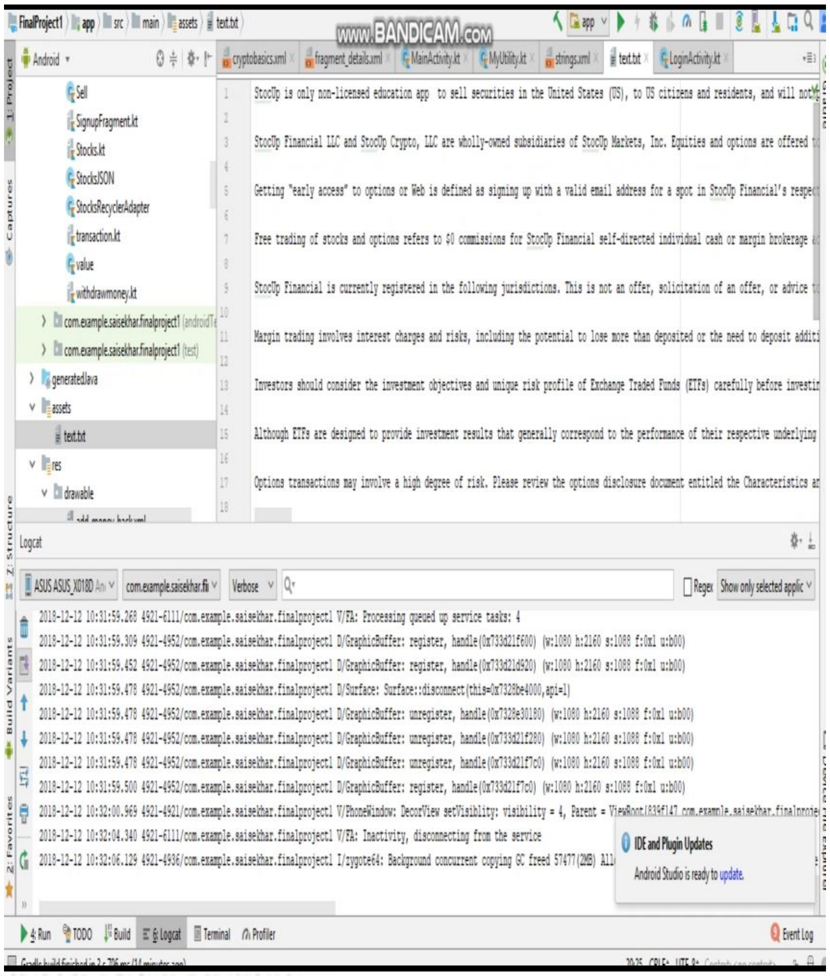
- We have stored data that is displayed in the help tab inside the application system.
- The is pre-populated with static data and the same data is used to dislay in the tab.
- This makes fetching the data that is less likely to change and be the same for all users faster.

- Also if we store the data on cloud, there is un-necessary calls to retrieve the data which internal storage helps in avoiding.



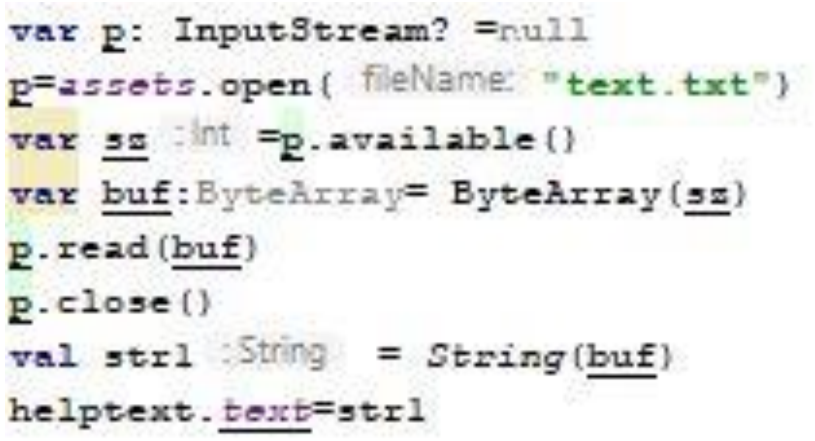
StocUp is only non-licensed education app to sell securities in the United States (US), to US citizens and residents, and will not permit anyone who is not a legal US resident to purchase or sell securities through its application until such time as it is approved to engage in the securities and investment business by the appropriate regulator(s) of a foreign jurisdiction. StocUp is currently available to legal US residents over 18 years of age only. StocUp is not providing financial advice or recommending securities through the use of its site and it is authorized to engage in the securities business solely in the US.

StocUp Financial LLC and StocUp Crypto, LLC are wholly-owned subsidiaries of StocUp Markets, Inc. Equities and options are offered to self-directed customers by StocUp Financial. StocUp Financial is a member of the Financial Industry Regulatory Authority (FINRA) and the Securities Investor Protection Corporation (SIPC), which protects securities customers of its members up to $500,000 (including $250,000 for claims for cash). Explanatory brochure available upon request or at sipc.org. Cryptocurrency trading is offered through an account with StocUp Crypto. StocUp Crypto is not a member of FINRA or SIPC. Cryptocurrencies are


FinalProject1 app src main assets text.txt
WWW.BANDICAM.COM
Android
cryptobasics.xml fragment_details.xml MainActivity.kt MyUtility.kt strings.xml text.txt LoginActivity.kt
Sell
SignupFragment.kt
Stocks.kt
StocksJSON
StocksRecyclerAdapter
transaction.kt
value
withdrawmoney.kt
com.example.saisekhar.finalproject1 (androidTe
com.example.saisekhar.finalproject1 (test)
generatedJava
assets
text.txt
res
drawable
Logcat
ASUS ASUS_X018D
com.example.saisekhar.fi
Verbose
Regex
Show only selected applic
IDE and Plugin Updates
Android Studio is ready to update.
4: Run
TODO
Build
6: Logcat
Terminal
Profiler
Event Log

```
StocUp is only non-licensed education app  to sell securities in the United States (US), to US citizens and residents, and will not

StocUp Financial LLC and StocUp Crypto, LLC are wholly-owned subsidiaries of StocUp Markets, Inc. Equities and options are offered t

Getting "early access" to options or Web is defined as signing up with a valid email address for a spot in StocUp Financial's respec

Free trading of stocks and options refers to $0 commissions for StocUp Financial self-directed individual cash or margin brokerage a

StocUp Financial is currently registered in the following jurisdictions. This is not an offer, solicitation of an offer, or advice t

Margin trading involves interest charges and risks, including the potential to lose more than deposited or the need to deposit additi

Investors should consider the investment objectives and unique risk profile of Exchange Traded Funds (ETFs) carefully before investin

Although ETFs are designed to provide investment results that generally correspond to the performance of their respective underlying

Options transactions may involve a high degree of risk. Please review the options disclosure document entitled the Characteristics ar
```

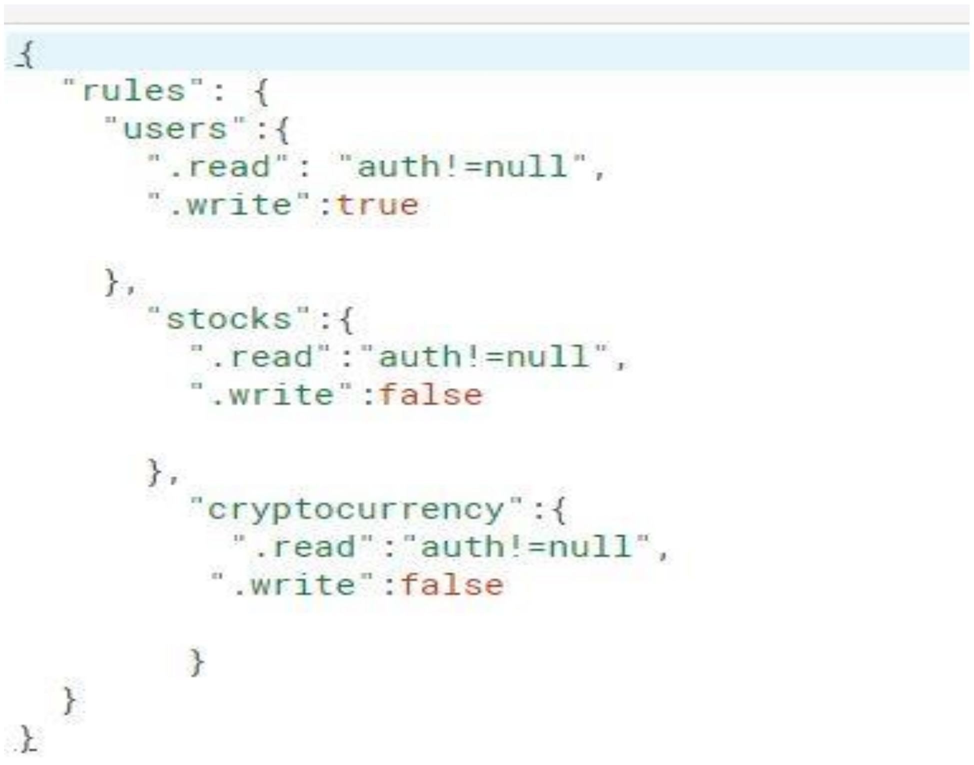
```
2018-12-12 10:31:59.268 4921-6111/com.example.saisekhar.finalproject1 V/FA: Processing queued up service tasks: 4
2018-12-12 10:31:59.309 4921-4952/com.example.saisekhar.finalproject1 D/GraphicBuffer: register, handle(0x733d21f600) (w:1080 h:2160 s:1088 f:0x1 u:b00)
2018-12-12 10:31:59.452 4921-4952/com.example.saisekhar.finalproject1 D/GraphicBuffer: register, handle(0x733d21d920) (w:1080 h:2160 s:1088 f:0x1 u:b00)
2018-12-12 10:31:59.478 4921-4952/com.example.saisekhar.finalproject1 D/Surface: Surface::disconnect(this=0x7328be4000,api=1)
2018-12-12 10:31:59.478 4921-4952/com.example.saisekhar.finalproject1 D/GraphicBuffer: unregister, handle(0x7328e30180) (w:1080 h:2160 s:1088 f:0x1 u:b00)
2018-12-12 10:31:59.478 4921-4952/com.example.saisekhar.finalproject1 D/GraphicBuffer: unregister, handle(0x733d21f280) (w:1080 h:2160 s:1088 f:0x1 u:b00)
2018-12-12 10:31:59.478 4921-4952/com.example.saisekhar.finalproject1 D/GraphicBuffer: unregister, handle(0x733d21f7c0) (w:1080 h:2160 s:1088 f:0x1 u:b00)
2018-12-12 10:31:59.500 4921-4952/com.example.saisekhar.finalproject1 D/GraphicBuffer: register, handle(0x733d21f7c0) (w:1080 h:2160 s:1088 f:0x1 u:b00)
2018-12-12 10:32:00.969 4921-4921/com.example.saisekhar.finalproject1 V/PhoneWindow: DecorView setVisiblity: visibility = 4, Parent = ViewRoot{839f147 com.example.saisekhar.finalproje
2018-12-12 10:32:04.340 4921-6111/com.example.saisekhar.finalproject1 V/FA: Inactivity, disconnecting from the service
2018-12-12 10:32:06.129 4921-4936/com.example.saisekhar.finalproject1 I/zygote64: Background concurrent copying GC freed 57477(2MB) All
```



```
var p: InputStream? =null
p=assets.open( fileName: "text.txt")
var ss :Int =p.available()
var buf:ByteArray= ByteArray(ss)
p.read(buf)
p.close()
val str1 :String = String(buf)
helptext.text=str1
```

16. Protecting data in Cloud

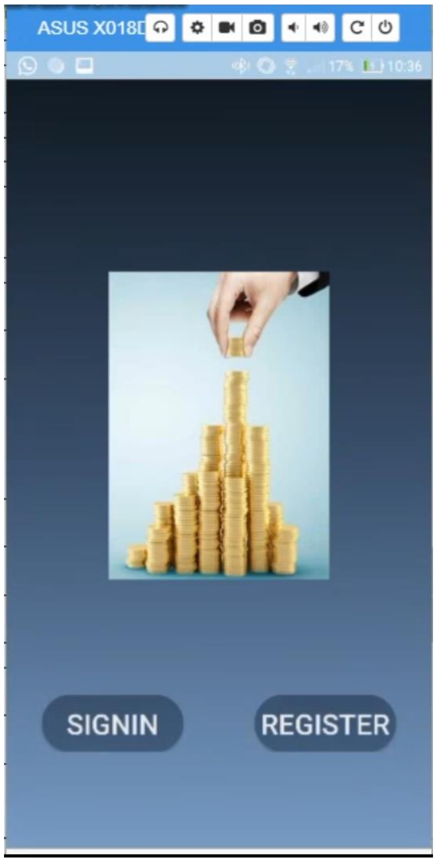
- It is of utmost importance for a financial application that no user can update the values of stocks.
- We are saving all stocks and crypto-currency prices on the cloud, there appropriate rules of access have been deployed so that users cannot change these two data fields.
- Also since users can update their own accounts, the user can update its own profile entry in the cloud.

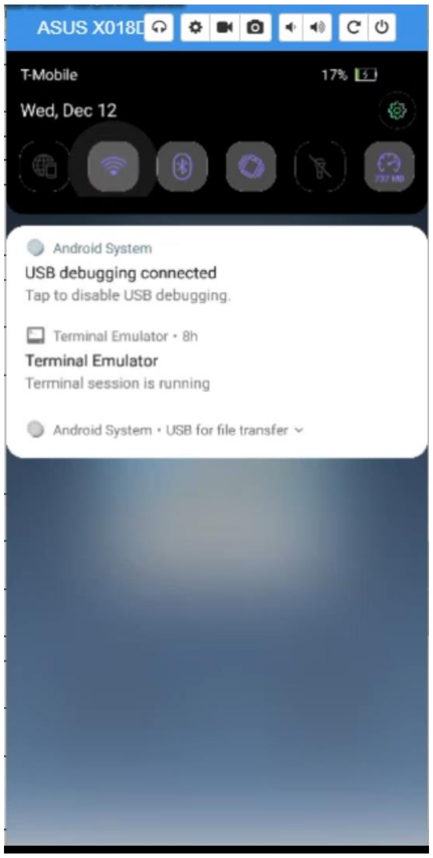


```
{
  "rules": {
    "users":{
      ".read": "auth!=null",
      ".write":true

    },
      "stocks":{
        ".read":"auth!=null",
        ".write":false

      },
        "cryptocurrency":{
          ".read":"auth!=null",
         ".write":false

        }
  }
}
```

17. Broadcast Receiver

- Broadcast messages are released by the android device when certain actions like internet disconnects.
- If there are proper adapters waiting to listen to these broadcast messages they can do actions.
- Because it is necessary to have internet connectivity to user StocUp we have implemented a broadcast receiver which tells the users that network has changed.

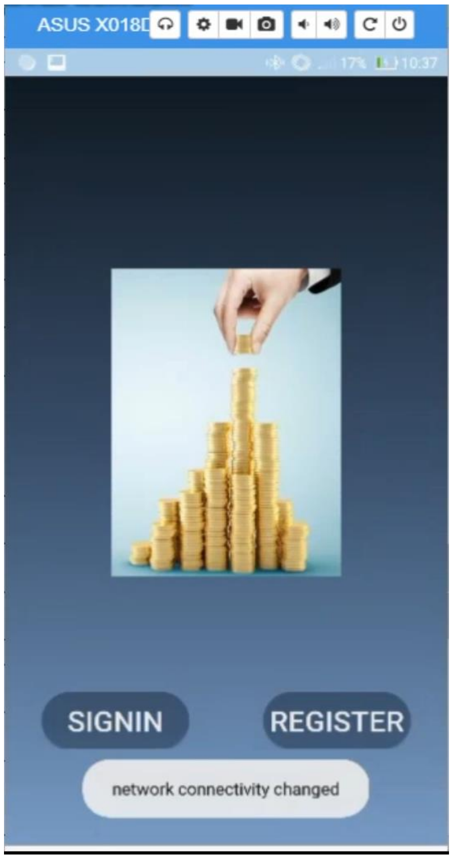




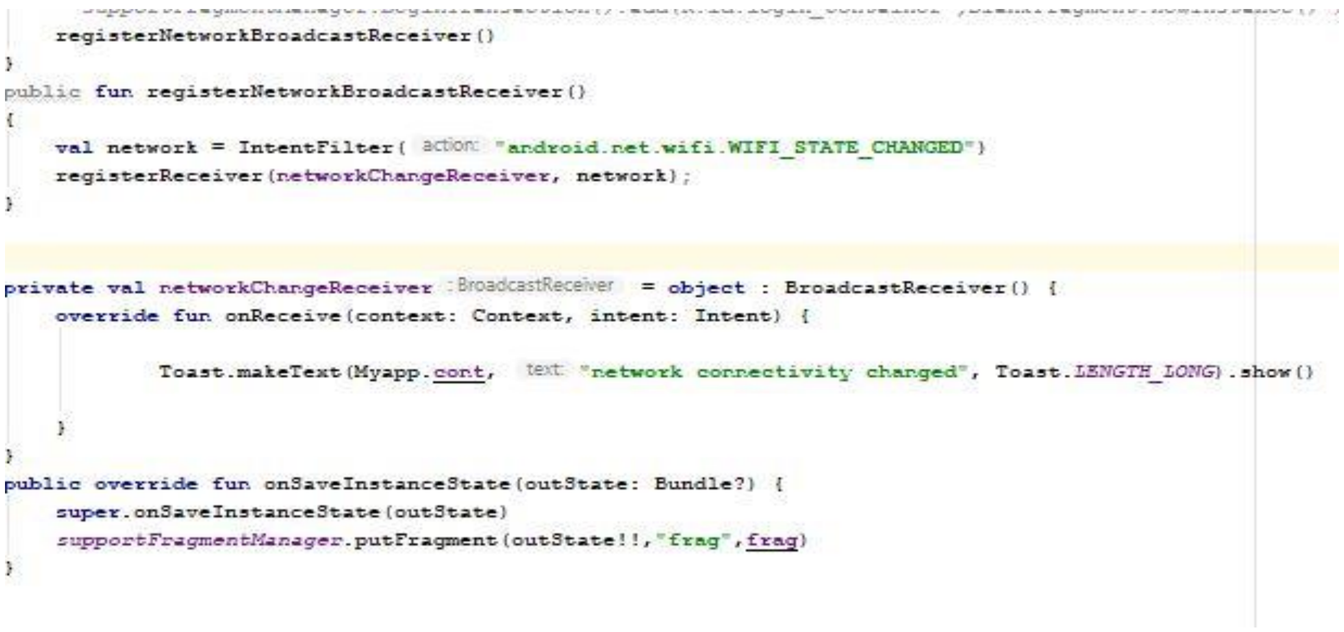
Android System
USB debugging connected
Tap to disable USB debugging.

Terminal Emulator • 8h
Terminal Emulator
Terminal session is running

Android System • USB for file transfer



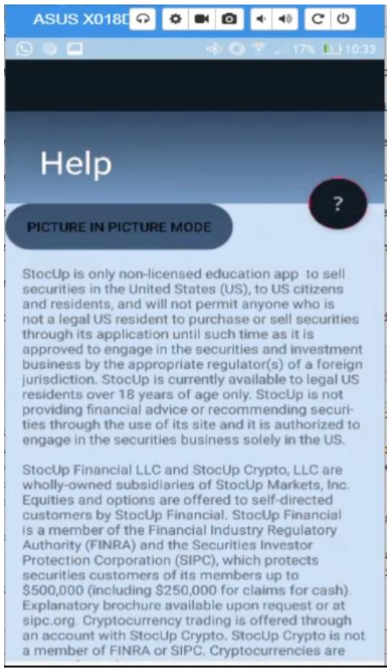
network connectivity changed



```
    registerNetworkBroadcastReceiver()
}
public fun registerNetworkBroadcastReceiver()
{
    val network = IntentFilter( action: "android.net.wifi.WIFI_STATE_CHANGED")
    registerReceiver(networkChangeReceiver, network);
}

private val networkChangeReceiver :BroadcastReceiver = object : BroadcastReceiver() {
    override fun onReceive(context: Context, intent: Intent) {

            Toast.makeText(Myapp.cont,  text: "network connectivity changed", Toast.LENGTH_LONG).show()

    }
}
public override fun onSaveInstanceState(outState: Bundle?) {
    super.onSaveInstanceState(outState)
    supportFragmentManager.putFragment(outState!!,"frag",frag)
}
```

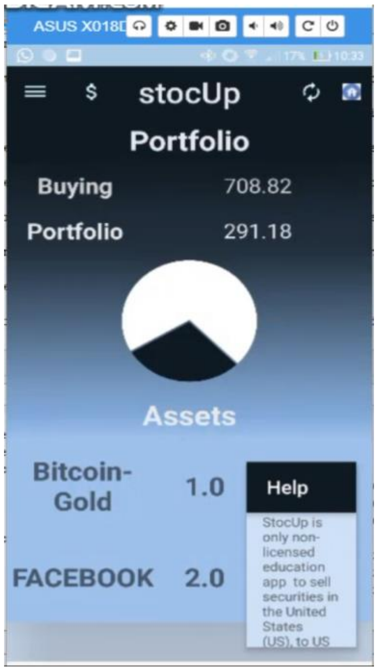
18. Picture In Picture

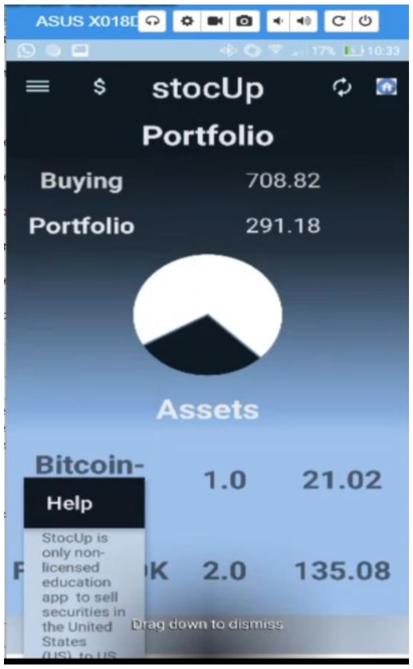
- Picture in Picture is a way to change display more than one window at the same time.
- We have implanted a Picture in Picture mode in the help window which allows users to use this feature.
- As we can see, when the user clicks this button the picture in picture mode is displayed. You can click the picture, move it around and when you click it again, the help window opens again.

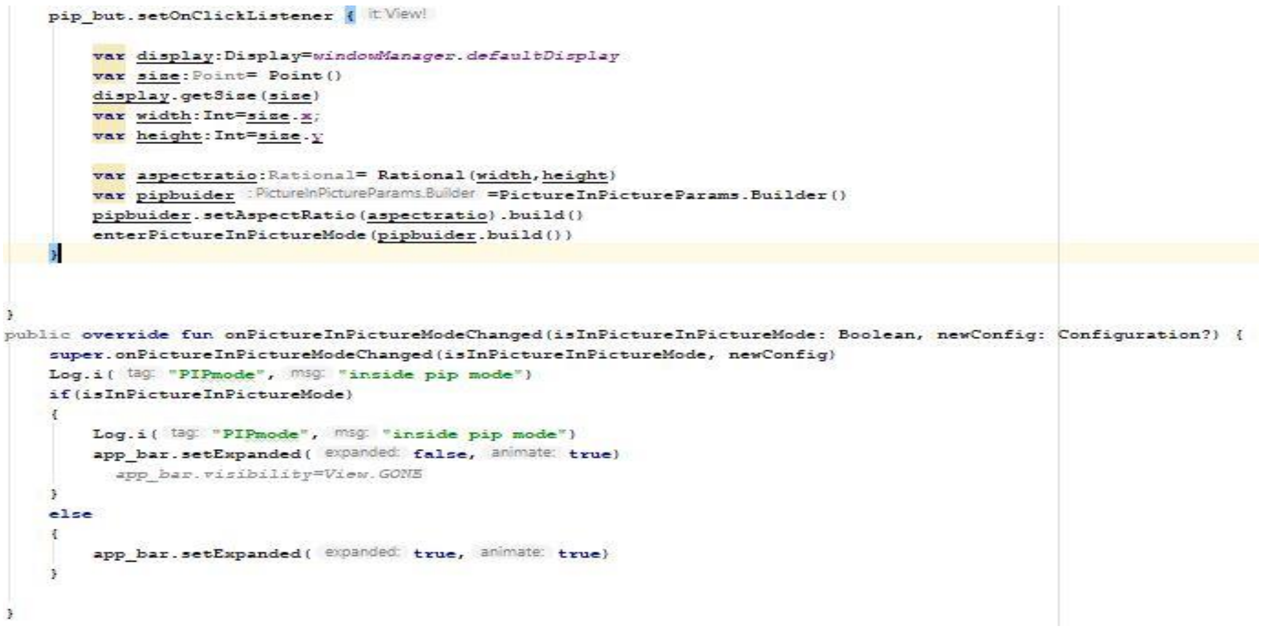


StocUp is only non-licensed education app to sell securities in the United States (US), to US citizens and residents, and will not permit anyone who is not a legal US resident to purchase or sell securities through its application until such time as it is approved to engage in the securities and investment business by the appropriate regulator(s) of a foreign jurisdiction. StocUp is currently available to legal US residents over 18 years of age only. StocUp is not providing financial advice or recommending securities through the use of its site and it is authorized to engage in the securities business solely in the US.

StocUp Financial LLC and StocUp Crypto, LLC are wholly-owned subsidiaries of StocUp Markets, Inc. Equities and options are offered to self-directed customers by StocUp Financial. StocUp Financial is a member of the Financial Industry Regulatory Authority (FINRA) and the Securities Investor Protection Corporation (SIPC), which protects securities customers of its members up to $500,000 (including $250,000 for claims for cash). Explanatory brochure available upon request or at sipc.org. Cryptocurrency trading is offered through an account with StocUp Crypto. StocUp Crypto is not a member of FINRA or SIPC. Cryptocurrencies are


ASUS X018D
stocUp
Portfolio
Buying 708.82
Portfolio 291.18
Assets
Bitcoin-Gold 1.0
FACEBOOK 2.0
Help
StocUp is only non-licensed education app to sell securities in the United States (US), to US

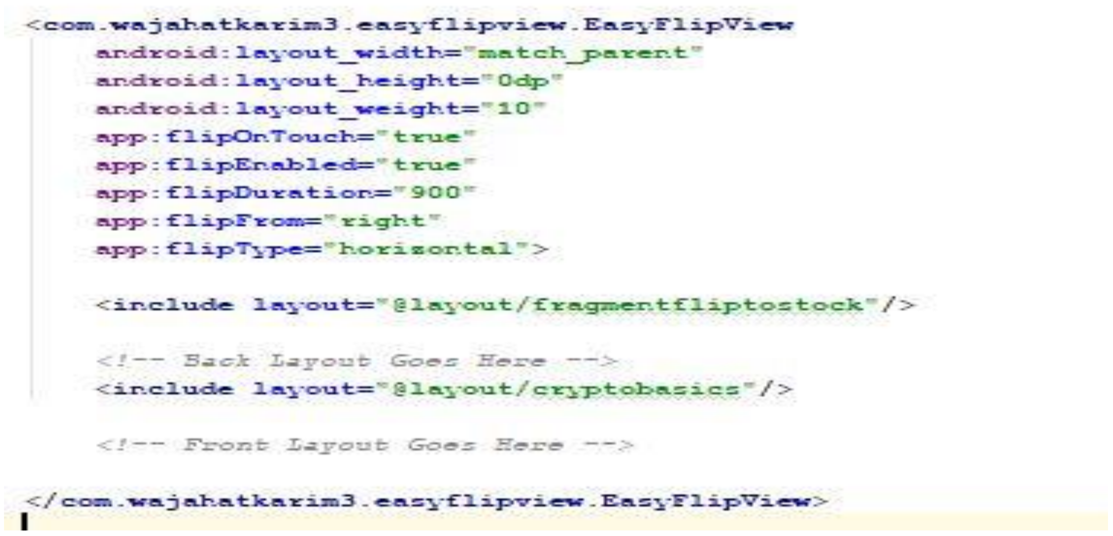




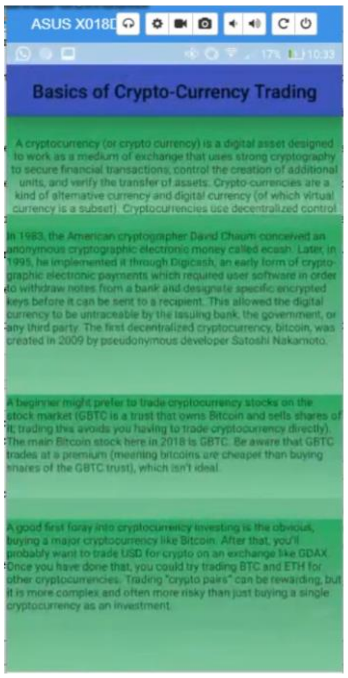
```
    pip_but.setOnClickListener { it: View!

        var display:Display=windowManager.defaultDisplay
        var size:Point= Point()
        display.getSize(size)
        var width:Int=size.x;
        var height:Int=size.y

        var aspectratio:Rational= Rational(width,height)
        var pipbuider :PictureInPictureParams.Builder =PictureInPictureParams.Builder()
        pipbuider.setAspectRatio(aspectratio).build()
        enterPictureInPictureMode(pipbuider.build())
    }


}
public override fun onPictureInPictureModeChanged(isInPictureInPictureMode: Boolean, newConfig: Configuration?) {
    super.onPictureInPictureModeChanged(isInPictureInPictureMode, newConfig)
    Log.i( tag: "PIPmode", msg: "inside pip mode")
    if(isInPictureInPictureMode)
    {
        Log.i( tag: "PIPmode", msg: "inside pip mode")
        app_bar.setExpanded( expanded: false, animate: true)
         app_bar.visibility=View.GONE
    }
    else
    {
        app_bar.setExpanded( expanded: true, animate: true)
    }

}
```

19. Flipbook Style

- Flipbook style is an impressive way to show user details or content in a way that makes it easy for them to read.
- The flipbook makes it look similar to the way a book works.
- We have implemented flipbook in our tutorial tab. This tab contains basic information for how stocks and crypto-currencies work.
- As we can see the flipbook makes the transition between different views look great.



```
<com.wajahatkarim3.easyflipview.EasyFlipView
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="10"
    app:flipOnTouch="true"
    app:flipEnabled="true"
    app:flipDuration="900"
    app:flipFrom="right"
    app:flipType="horizontal">

    <include layout="@layout/fragmentfliptostock"/>

    <!-- Back Layout Goes Here -->
    <include layout="@layout/cryptobasics"/>

    <!-- Front Layout Goes Here -->

</com.wajahatkarim3.easyflipview.EasyFlipView>
```



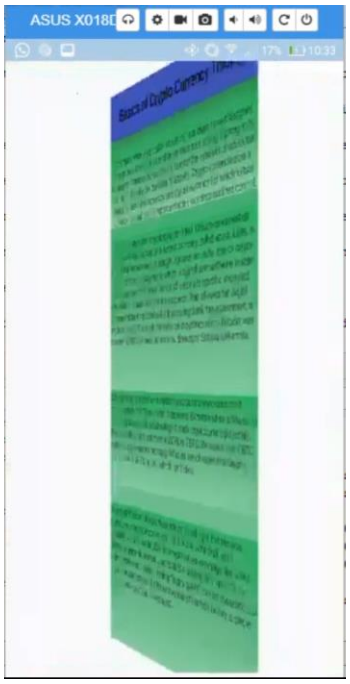
Basics of Crypto-Currency Trading

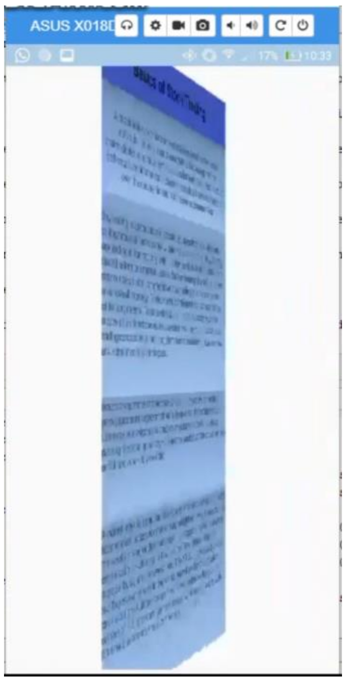
A cryptocurrency (or crypto currency) is a digital asset designed to work as a medium of exchange that uses strong cryptography to secure financial transactions, control the creation of additional units, and verify the transfer of assets. Crypto-currencies are a kind of alternative currency and digital currency (of which virtual currency is a subset). Cryptocurrencies use decentralized control

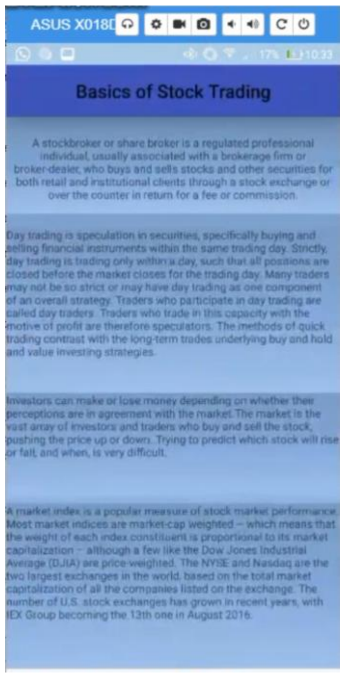
In 1983, the American cryptographer David Chaum conceived an anonymous cryptographic electronic money called ecash. Later, in 1995, he implemented it through Digicash, an early form of cryptographic electronic payments which required user software in order to withdraw notes from a bank and designate specific encrypted keys before it can be sent to a recipient. This allowed the digital currency to be untraceable by the issuing bank, the government, or any third party. The first decentralized cryptocurrency, bitcoin, was created in 2009 by pseudonymous developer Satoshi Nakamoto.

A beginner might prefer to trade cryptocurrency stocks on the stock market (GBTC is a trust that owns Bitcoin and sells shares of it; trading this avoids you having to trade cryptocurrency directly). The main Bitcoin stock here in 2018 is GBTC. Be aware that GBTC trades at a premium (meaning bitcoins are cheaper than buying shares of the GBTC trust), which isn't ideal.

A good first foray into cryptocurrency investing is the obvious, buying a major cryptocurrency like Bitcoin. After that, you'll probably want to trade USD for crypto on an exchange like GDAX. Once you have done that, you could try trading BTC and ETH for other cryptocurrencies. Trading "crypto pairs" can be rewarding, but it is more complex and often more risky than just buying a single cryptocurrency as an investment.







Basics of Stock Trading

A stockbroker or share broker is a regulated professional individual, usually associated with a brokerage firm or broker-dealer, who buys and sells stocks and other securities for both retail and institutional clients through a stock exchange or over the counter in return for a fee or commission.

Day trading is speculation in securities, specifically buying and selling financial instruments within the same trading day. Strictly, day trading is trading only within a day, such that all positions are closed before the market closes for the trading day. Many traders may not be so strict or may have day trading as one component of an overall strategy. Traders who participate in day trading are called day traders. Traders who trade in this capacity with the motive of profit are therefore speculators. The methods of quick trading contrast with the long-term trades underlying buy and hold and value investing strategies.

Investors can make or lose money depending on whether their perceptions are in agreement with the market. The market is the vast array of investors and traders who buy and sell the stock, pushing the price up or down. Trying to predict which stock will rise or fall, and when, is very difficult.

A market index is a popular measure of stock market performance. Most market indices are market-cap weighted – which means that the weight of each index constituent is proportional to its market capitalization – although a few like the Dow Jones Industrial Average (DJIA) are price-weighted. The NYSE and Nasdaq are the two largest exchanges in the world, based on the total market capitalization of all the companies listed on the exchange. The number of U.S. stock exchanges has grown in recent years, with IEX Group becoming the 13th one in August 2016.

Vision and Scope

- We have a vision to use the application as a competitive and informative tool for students of the market where they can hone their skills and apply their principles to trade in a virtual environment.
- We also hope to add other features like daily highlights of the market, allow users at a certain locations to create chat groups and add notifications to user when their favorite stocks are in the news
- This app hopes to help people make money and feel confident and not feel that markets move randomly but understand patterns and expert guidance.

++