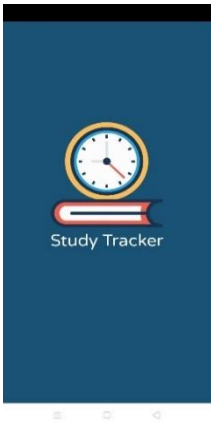
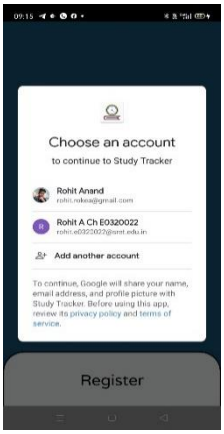
Course: CSE 230 Mobile Programming
Topic: Study Tracker App
Objective: To Help individuals and teams set goals and track their progress towards those goals

Splash Screen



Login

Google Sign In

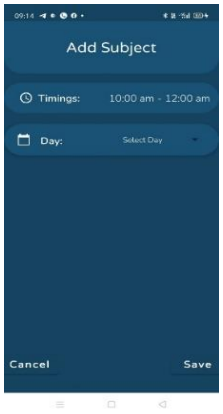


Sign Up

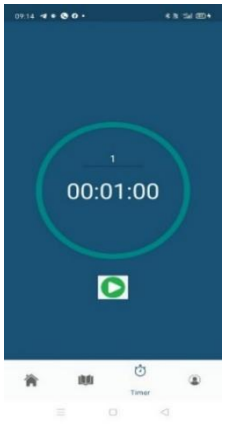
Home Page

Book Page

Add New Subject



Timer Page



Profile

TEAM MEMBERS

Rohit A Ch

E0320022

Kumaresh N M

E0320004

Adhithyan B

E0320005

Dhrish S Kumar

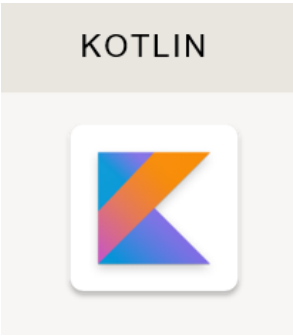
E0320005

Introduction:

Study tracker helps individuals and teams can use this tool to define goals, measure their progress toward those goals, and review their progress using a visual depiction. It also helps to enhance learner agency—knowledge of oneself as a learner—and increase the rate of learning.

Tools & Technology:

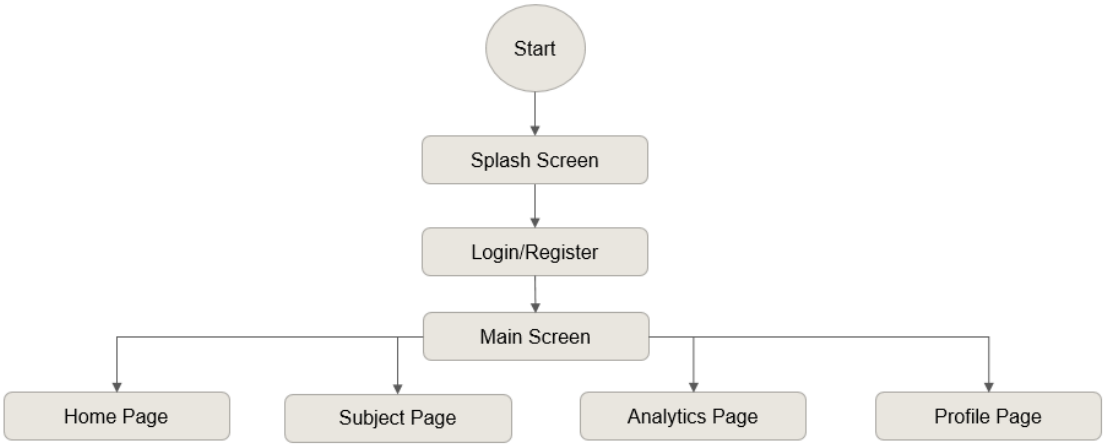
Android Studio is a unified development environment that allows you to create apps for Android phones, tablets, Android Wear, Android TV, and Android Auto. Structured code modules allow you to break down your project into functional parts that you can create, test, and debug independently.



JetBrains created Kotlin, a modern general-purpose programming language. Its full Java compatibility and succinct syntax make it a desirable language for web development, Android development, and other applications.

Firebase is a Google-backed app development platform that allows developers to create apps for iOS, Android, and the web. Firebase delivers analytics tracking, reporting, and app issue fixes, as well as marketing and product experimentation capabilities.

Flow Chart:



Code:

activity_main.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.coordinatorlayout.widget.CoordinatorLayout
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=".MainActivity">
    <LinearLayout
        android:id="@+id/firstlayout"
        android:layout_width="match_parent"
        android:layout_height="670dp"
        android:orientation="vertical">
    </LinearLayout>
    <include layout="@layout/fragment_blank" />
</androidx.coordinatorlayout.widget.CoordinatorLayout>
```

MainActivity.kt:

```
package com.example.myapplication

import android.content.ContentValues
import android.content.Context
import android.content.SharedPreferences
import android.os.Bundle
import android.text.Editable
import android.text.TextWatcher
import android.util.Log
import android.util.Patterns
import android.view.View
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity
import androidx.constraintlayout.widget.ConstraintLayout
import androidx.fragment.app.commit
import androidx.fragment.app.replace
import com.example.myapplication.data.User
import com.google.android.gms.auth.api.identity.BeginSignInRequest
import com.google.android.gms.auth.api.identity.SignInClient
import com.google.android.gms.auth.api.signin.GoogleSignInClient
import com.google.android.material.bottomsheet.BottomSheetBehavior
import com.google.firebase.auth.FirebaseAuth
import com.google.firebase.database.DataSnapshot
import com.google.firebase.database.DatabaseError
import com.google.firebase.database.DatabaseReference
import com.google.firebase.database.ValueEventListener
import com.google.firebase.database.ktx.database
import com.google.firebase.ktx.Firebase
import kotlinx.android.synthetic.main.activity_add_subject.*
import kotlinx.android.synthetic.main.fragment_blank.*
import kotlinx.android.synthetic.main.fragment_loginpage.*
import java.sql.Date
import java.sql.Time
import java.text.DateFormat
import java.text.SimpleDateFormat


class MainActivity : AppCompatActivity() {
    lateinit var oneTapClient: SignInClient
    lateinit var signInRequest: BeginSignInRequest
    lateinit var mGoogleSignInClient: GoogleSignInClient
    val Req_Code: Int = 123
    private lateinit var firebaseAuth: FirebaseAuth
    private var database: DatabaseReference = Firebase.database.reference
    var user_count:Long =0
```

```
    var count:Long?=null
// var sharedPreferences:SharedPreferences= this.getSharedPreferences("login",
Context.MODE_PRIVATE)
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        fname.addTextChangedListener(fWatcher)
        lname.addTextChangedListener(lWatcher)
        email_id.addTextChangedListener(emailWatcher)
        pwd.addTextChangedListener(pdWatcher)
        repwd.addTextChangedListener(rpdWatcher)
        //createRequest()
        register.setOnClickListener{
            val existUser = database.child ("users")
            Log.w(ContentValues.TAG, existUser.toString())
            existUser.addListenerForSingleValueEvent(object : ValueEventListener {
                override fun onDataChange(snapshot: DataSnapshot) {
                    if(snapshot.exists()){
                        user_count=snapshot.childrenCount+1
                        }
                    else{
                        user_count=0+1
                    }
                    var user_name="user$user_count"
                    writeNewUser(user_name,(fname.text).toString(),(lname.text).toString(),(email_id.text).toStr
ing(),(pwd.text.toString()))
                }
                override fun onCancelled(databaseError: DatabaseError) {
                    // Getting Post failed, log a message
                    Log.w(ContentValues.TAG, "loadUser:onCancelled", databaseError.toException())
                    // ...
                }
            })
        }
        val manager= supportFragmentManager
        manager.commit{
            setReorderingAllowed(true)
            replace<Login>(R.id.firstlayout)
        }
        lateinit var bottomSheetBehavior: BottomSheetBehavior<ConstraintLayout>
            bottomSheetBehavior = BottomSheetBehavior.from(bottomSheet)
            bottomSheetBehavior.addBottomSheetCallback(object :
                BottomSheetBehavior.BottomSheetCallback() {
                override fun onSlide(bottomSheet: View, slideOffset: Float) {
                    // handle onSlide
                }
```

```
            override fun onStateChanged(bottomSheet: View, newState: Int) {
               when (newState) {
BottomSheetBehavior.STATE_COLLAPSED -> {
                     fname.text.clear()
                     lname.text.clear()
                     email_id.text.clear()
                     pwd.text.clear()
                     repwd.text.clear()
                     Toast.makeText(this@MainActivity, "Swipe Up to Register",
Toast.LENGTH_SHORT).show()}
                  BottomSheetBehavior.STATE_EXPANDED -> Toast.makeText(this@MainActivity,
"Swipe Down to Login", Toast.LENGTH_SHORT).show()
                  //BottomSheetBehavior.STATE_DRAGGING -> Toast.makeText(this@MainActivity,
"STATE_DRAGGING", Toast.LENGTH_SHORT).show()
                  //BottomSheetBehavior.STATE_SETTLING -> Toast.makeText(this@MainActivity,
"STATE_SETTLING", Toast.LENGTH_SHORT).show()
                  //BottomSheetBehavior.STATE_HIDDEN -> Toast.makeText(this@MainActivity,
"STATE_HIDDEN", Toast.LENGTH_SHORT).show()
                  //else -> Toast.makeText(this@MainActivity, "OTHER_STATE",
Toast.LENGTH_SHORT).show()
                  }
               }
            })
         fun bundle():Boolean{
            val intent = android.content.Intent(this, HomeActivity::class.java)
            var bundle:Bundle?= intent.extras
            var str=bundle!!.getBoolean("ISLOGGEDIN")
            if(str){
               startActivity(intent)
               finish()
            }
            else{
               val i = android.content.Intent(this, MainActivity::class.java)
               startActivity(i)
               finish()
            }
            return str
         }
}
  private val fWatcher = object : TextWatcher{
      override fun afterTextChanged(s: Editable?) {
      }
      override fun beforeTextChanged(s: CharSequence?, start: Int, count: Int, after: Int) {
      }
      override fun onTextChanged(s: CharSequence?, start: Int, before: Int, count: Int) {
         if (s != null) {
```

```
            if(s.length ==0){
                fname.setError("First Name is Required")
            }
        }
    }
}
  private val lWatcher = object : TextWatcher{
      override fun afterTextChanged(s: Editable?) {
      }
      override fun beforeTextChanged(s: CharSequence?, start: Int, count: Int, after: Int) {
      }
      override fun onTextChanged(s: CharSequence?, start: Int, before: Int, count: Int) {
          if (s != null) {
              if(s.length ==0){
                  lname.setError("Last Name is Required")
              }
          }
      }
  }
  private val emailWatcher = object : TextWatcher{
      override fun afterTextChanged(s: Editable?) {
      }
      override fun beforeTextChanged(s: CharSequence?, start: Int, count: Int, after: Int) {
      }
      override fun onTextChanged(s: CharSequence?, start: Int, before: Int, count: Int) {
          if (s != null) {
              if(!((s.contains("@"))&&(Patterns.EMAIL_ADDRESS.matcher(s).matches()))){
                  email_id.setError("Not a vaild email")
              }
              if(s.length ==0){
                  email_id.setError("Email is Required")
              }
          }
      }
  }
  private val pdWatcher = object : TextWatcher{
      override fun afterTextChanged(s: Editable?) {
      }
      override fun beforeTextChanged(s: CharSequence?, start: Int, count: Int, after: Int) {
      }
      override fun onTextChanged(s: CharSequence?, start: Int, before: Int, count: Int) {
          if (s != null) {
              if(s.length ==0){
                  pwd.setError("Password is Required")
              }
          }
      }
```

```
    }
    private val rpdWatcher = object : TextWatcher{
        override fun afterTextChanged(s: Editable?) {


 }
        override fun beforeTextChanged(s: CharSequence?, start: Int, count: Int, after: Int) {
        }
        override fun onTextChanged(s: CharSequence?, start: Int, before: Int, count: Int) {
            if (s != null) {
                if(s.length01:39 PM



==0){
                    repwd.setError("Must re-enter password")
                }
            }
        }
    }
    lateinit var usernameInDB:String
    fun preference(username:String){
        var sharedPreferences:SharedPreferences= getSharedPreferences("login",
Context.MODE_PRIVATE)?:return
        with (sharedPreferences.edit()) {
            putString("Username", username)
            apply()
        }
    }
    fun writeNewUser(userId: String, firstname: String, lastname: String, email: String,password:String) {
        val user = User(firstname,lastname, email,password)
        database.child("users").child(userId).setValue(user)
    }
}
```

HomeActivity.kt:

```
package com.example.myapplication

import android.app.Notification
import android.app.NotificationChannel
import android.app.NotificationManager
import android.app.PendingIntent
import android.content.ContentValues
import android.content.Context
import android.content.Intent
import android.graphics.BitmapFactory
import android.graphics.Color
import android.os.Build
import android.os.Bundle
import android.util.Log
import android.widget.RemoteViews
import androidx.appcompat.app.AppCompatActivity
import androidx.fragment.app.Fragment
import com.example.myapplication.Adapter.SubjectAdapter
import com.google.firebase.database.DataSnapshot
import com.google.firebase.database.DatabaseError
import com.google.firebase.database.DatabaseReference
import com.google.firebase.database.ValueEventListener
import com.google.firebase.database.ktx.database
import com.google.firebase.ktx.Firebase
import kotlinx.android.synthetic.main.activity_home.*
import kotlinx.android.synthetic.main.fragment_home.*
import java.text.SimpleDateFormat
import java.util.*


class HomeActivity : AppCompatActivity() {
    lateinit var notificationManager: NotificationManager
    lateinit var notificationChannel: NotificationChannel
    lateinit var builder: Notification.Builder
    private val channelId = "i.apps.notifications"
    private val description = "Test notification"
    var sub_count:Int=0
    private var database: DatabaseReference = Firebase.database.reference
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_home)
        val home_frag=Home()
        val subjects_frag=Subjects()
        val analytics_frag=Analytics()
        val profile_frag=Profile()
        setCurrentFragment(home_frag)
```

```
        bottomNavigationView.setOnNavigationItemSelectedListener {
            when(it.itemId){
                R.id.home-> setCurrentFragment(home_frag)
                R.id.book->setCurrentFragment(subjects_frag)
                R.id.analytics->setCurrentFragment(analytics_frag)
                R.id.profile->setCurrentFragment(profile_frag)
            }
            true
        }
    }
    private fun setCurrentFragment(fragment: Fragment) {
        supportFragmentManager.beginTransaction().apply{
            replace(R.id.fragment_layout,fragment)
            commit()
        }
    }
}
```

fragment_home.xml:

```
<?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"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    tools:context=".MainActivity">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical">
        <androidx.cardview.widget.CardView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_margin="10dp"
            app:cardCornerRadius="20dp">
            <androidx.constraintlayout.widget.ConstraintLayout
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:background="@drawable/home_fragment_card_edge"
                android:padding="10dp">
                <ImageView
                    android:layout_width="78dp"
```

```
            android:layout_height="67dp"
            android:layout_marginStart="4dp"
            android:src="@drawable/study"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent"
            app:layout_constraintVertical_bias="0.709" />
        <TextView
            android:id="@+id/home_day_textview"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginStart="96dp"
            android:layout_marginTop="40dp"
            android:text="Thurday, 23"
            android:textColor="?attr/colorOnSurface"
            android:textSize="20sp"
            android:textStyle="bold"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent" />
        <AnalogClock
            android:id="@+id/analogClock2"
            android:layout_width="130dp"
            android:layout_height="129dp"
            android:layout_marginEnd="4dp"
            app:layout_constraintEnd_toEndOf="parent"
            tools:ignore="MissingConstraints"
            tools:layout_editor_absoluteY="-10dp" />
      </androidx.constraintlayout.widget.ConstraintLayout>
    </androidx.cardview.widget.CardView>
  <ListView
    android:id="@+id/subjects_time"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"/>
  </LinearLayout>
</FrameLayout>
```

HomeActivity.kt-Fragment:

```
package com.example.myapplication

import android.app.Notification
import android.app.NotificationChannel
import android.app.NotificationManager
import android.content.ContentValues
import android.content.ContentValues.TAG
import android.content.Intent
import android.os.Bundle
import android.util.Log
import androidx.fragment.app.Fragment
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import com.example.myapplication.Adapter.SubjectAdapter
import com.google.firebase.database.DataSnapshot
import com.google.firebase.database.DatabaseError
import com.google.firebase.database.DatabaseReference
import com.google.firebase.database.ValueEventListener
import com.google.firebase.database.ktx.database
import com.google.firebase.ktx.Firebase
import kotlinx.android.synthetic.main.fragment_home.*
import kotlinx.android.synthetic.main.fragment_subjects.*
import java.text.SimpleDateFormat
import java.util.*


// TODO: Rename parameter arguments, choose names that match
// the fragment initialization parameters, e.g. ARG_ITEM_NUMBER


private const val ARG_PARAM1 = "param1"
private const val ARG_PARAM2 = "param2"
/**
 * A simple [Fragment] subclass.
 * Use the [Home.newInstance] factory method to
 * create an instance of this fragment.
 */


class Home : Fragment() {
    lateinit var notificationManager: NotificationManager
    lateinit var notificationChannel: NotificationChannel
    lateinit var builder: Notification.Builder
    private val channelId = "i.apps.notifications"
    private val description = "Test notification"
    // TODO: Rename and change types of parameters
```

```
    private var param1: String? = null
    private var param2: String? = null
    var sub_count:Int=0
    private var database: DatabaseReference = Firebase.database.reference
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        arguments?.let {
            param1 = it.getString(ARG_PARAM1)
            param2 = it.getString(ARG_PARAM2)
        }
    }


    override fun onCreateView(inflater: LayoutInflater, container: ViewGroup?,
                              savedInstanceState: Bundle?): View? {
        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.fragment_home, container, false)
    }


    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
        super.onViewCreated(view, savedInstanceState)
        val formatter = SimpleDateFormat("EEEEEEEEEEEEEE, dd")
        val date = Date()
        home_day_textview.text=formatter.format(date)
        val day=SimpleDateFormat("EEEE").format(date)
        Log.d(TAG,day)
        val getsubjects =
database.child("users").child("user1").child("subjects").orderByChild("day").equalTo(day)
        //Log.w(ContentValues.TAG, existUser.toString())
       getsubjects.addListenerForSingleValueEvent(object : ValueEventListener {
            override fun onDataChange(snapshot: DataSnapshot) {
                if(snapshot.exists()){
                    sub_count=Integer.parseInt((snapshot.childrenCount).toString())
                    var subjects = arrayOfNulls<String>(sub_count)
                    var days = arrayOfNulls<String>(sub_count)
                    var start= arrayOfNulls<String>(sub_count)
                    var end= arrayOfNulls<String>(sub_count)
                    var i=0
                    for (sub in snapshot.children){
                        Log.d(TAG,(sub.child("day").getValue()).toString())
                        subjects[i]=(sub.child("name").getValue()).toString()
                        days[i]=(sub.child("day").getValue()).toString()
                        start[i]=(sub.child("start").getValue()).toString()
                        end[i]=(sub.child("end").getValue()).toString()
                        i++
                    }
                    var adapter= SubjectAdapter(this@Home.requireActivity(),subjects,start,end)
```

```
                subjects_time.adapter=adapter
            }
            else{
            }
        }



override fun onCancelled(databaseError: DatabaseError) {
            // Getting Post failed, log a message
            Log.w(ContentValues.TAG, "loadUser:onCancelled", databaseError.toException())
            // ...
        }
    })
  }
  companion object {
    /**
     * Use this factory method to create a new instance of
     * this fragment using the provided parameters.
     *
     * @param param1 Parameter 1.
     * @param param2 Parameter 2.
     * @return A new instance of fragment Home.
     */
          @JvmStatic fun newInstance(param1: String, param2: String) =
          Home().apply {
            arguments = Bundle().apply {
              putString(ARG_PARAM1, param1)
              putString(ARG_PARAM2, param2)
            }
          }
  }
}
```

Subjects.kt –Fragment:

```
package com.example.myapplication

import android.content.ContentValues
import android.content.Intent
import android.os.Bundle
import android.util.Log
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.ArrayAdapter
import androidx.fragment.app.Fragment
import androidx.fragment.app.FragmentManager
import androidx.fragment.app.FragmentTransaction
import com.example.myapplication.Adapter.GridAdapter
import com.example.myapplication.Adapter.SubjectAdapter
import com.google.firebase.database.DataSnapshot
import com.google.firebase.database.DatabaseError
import com.google.firebase.database.DatabaseReference
import com.google.firebase.database.ValueEventListener
import com.google.firebase.database.ktx.database
import com.google.firebase.ktx.Firebase
import kotlinx.android.synthetic.main.fragment_home.*
import kotlinx.android.synthetic.main.fragment_subjects.*
// TODO: Rename parameter arguments, choose names that match
// the fragment initialization parameters, e.g. ARG_ITEM_NUMBER
private const val ARG_PARAM1 = "param1"
private const val ARG_PARAM2 = "param2"
/**
 * A simple [Fragment] subclass.
 * Use the [Subjects.newInstance] factory method to
 * create an instance of this fragment.
 */
class Subjects() : Fragment() {
    var sub_count:Int=0
    private var database: DatabaseReference = Firebase.database.reference
    // TODO: Rename and change types of parameters
    private var param1: String? = null
    private var param2: String? = null
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        arguments?.let {
            param1 = it.getString(ARG_PARAM1)
            param2 = it.getString(ARG_PARAM2)
        }
    }
```

```
override fun onCreateView(
    inflater: LayoutInflater, container: ViewGroup?,
    savedInstanceState: Bundle?
): View? {
    // Inflate the layout for this fragment
    return inflater.inflate(R.layout.fragment_subjects, container, false)
}
override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
    super.onViewCreated(view, savedInstanceState)
    add.setOnClickListener{
        var i = Intent(this@Subjects.context,AddSubject::class.java)
        startActivity(i)
    }
    val getsubjects = database.child("users").child("user1").child("subjects")
    //Log.w(ContentValues.TAG, existUser.toString())
    getsubjects.addListenerForSingleValueEvent(object : ValueEventListener {
        override fun onDataChange(snapshot: DataSnapshot) {
            if(snapshot.exists()){
                sub_count=Integer.parseInt((snapshot.childrenCount).toString())
                var subjects = arrayOfNulls<String>(sub_count)
                var days = arrayOfNulls<String>(sub_count)
                var start= arrayOfNulls<String>(sub_count)
                var end= arrayOfNulls<String>(sub_count)
                var i=0
                for (sub in snapshot.children){
                    Log.d(ContentValues.TAG,(sub.child("day").getValue()).toString())
                    subjects[i]=(sub.child("name").getValue()).toString()
                    days[i]=(sub.child("day").getValue()).toString()
                    start[i]=(sub.child("start").getValue()).toString()
                    end[i]=(sub.child("end").getValue()).toString()
                    i++
                }
                var adapter= this@Subjects.context?.let { GridAdapter(it,subjects) }
                subjects_grid.adapter=adapter
            }
            else{
            }
        }
        override fun onCancelled(databaseError: DatabaseError) {
            // Getting Post failed, log a message
            Log.w(ContentValues.TAG, "loadUser:onCancelled", databaseError.toException())
            // ...
        }
    })
}
companion object {
    /**
```

```
     * Use this factory method to create a new instance of
     * this fragment using the01:44 PM



provided parameters.
     *
     * @param param1 Parameter 1.
     * @param param2 Parameter 2.
     * @return A new instance of fragment Subjects.
     */
    // TODO: Rename and change types and number of parameters
    @JvmStatic
    fun newInstance(param1: String, param2: String) =
        Subjects().apply {
            arguments = Bundle().apply {
                putString(ARG_PARAM1, param1)
                putString(ARG_PARAM2, param2)
            }
        }
  }
}
```

Results & Conclusion:

- It provides evidence for 'high quality leadership and management'.
- It effectively monitors learners and helps evidence safeguarding in action (the Personal Development, Behaviour and Attitudes agenda).
- It supports the priority of ensuring excellent 'outcomes for learners' and supports the Quality of Education agenda.
- It improves overall effectiveness by ensuring that all levels of the organisation can be tracked and support provided where required.