en esta clase se da un proyecto en el cual trabajar, aplicar lo aprendido y cosas nuevas que debemos investigar, por mi parte continúo buscando comprender los contenidos anteriores aunque no pueda ir al dia en la clase

```
class MainActivity : AppCompatActivity(), SearchView.OnQueryTextListener {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        b = ActivityMainBinding.inflate(layoutInflater)
        setContentView(b.root)
        b.svDogs.setOnQueryTextListener(this)
        initRecyclerView()

    }

    private fun initRecyclerView() {
        adapter = DogAdapter(dogImages)
        b.rvDogs.layoutManager = LinearLayoutManager(this)
        b.rvDogs.adapter = adapter
    }

    private fun getRetrofit(): Retrofit {
        return Retrofit.Builder()
            .baseUrl("https://dog.ceo/api/breed/")//url de donde consultaremos sin poner el valor
a consultar
            .addConverterFactory(GsonConverterFactory.create())
            .build()
    }

    private fun searchByName(query: String) {
        CoroutineScope(Dispatchers.IO).launch {
            //corrutina, permite que tod0 lo siguiente se haga en un hilo distinto
            val call =
getRetrofit().create(APIService::class.java).getDogsByBreeds("$query/images")
            //se pasa la url del json+imagenes para consultarlas
            val puppies: DogResponse? = call.body()
            runOnUiThread {//permite volver al hilo principal
                if (call.isSuccessful) {
                    val images = puppies?.images ?: emptyList()
                    dogImages.clear()
                    dogImages.addAll(images)
                    adapter.notifyDataSetChanged()

                } else {
                    showError()
                }
                hideKeyboard()
```

```
            }
        }
    }

    private fun hideKeyboard() {
        val imm = getSystemService(INPUT_METHOD_SERVICE) as InputMethodManager
        imm.hideSoftInputFromWindow(b.viewRoot.windowToken, 0)
        //funcion para esconder el teclado al realizar la busqueda
    }

    private fun showError() {
        Toast.makeText(this, "ha ocurrido un error", Toast.LENGTH_SHORT).show()
    }

    override fun onQueryTextSubmit(query: String?): Boolean {
        if (!query.isNullOrEmpty()){
            searchByName(query.lowercase())

        }
        return true
    }

    override fun onQueryTextChange(newText: String?): Boolean {
        return true
    }

}
```