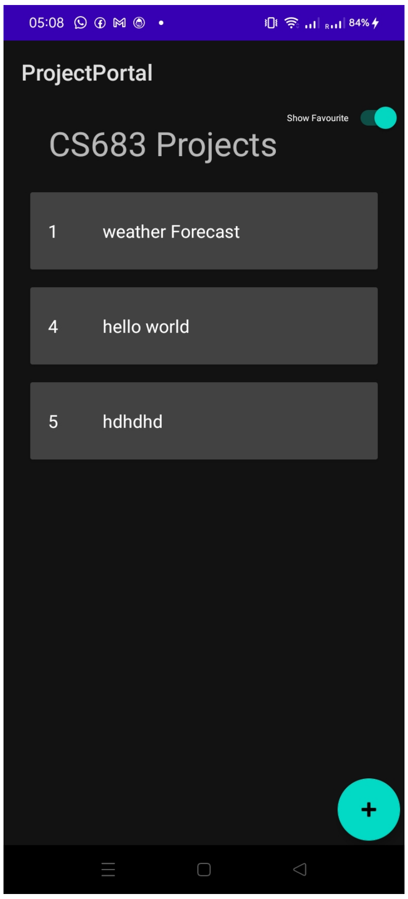
Lab5

GithubRepo The repo contains the lab5 code and doc too.

- Implemented Room Dao whose methods are wraped around couroutine scopes and the methods are made suspend.

```
@Dao
interface ProjectDao {
    @Insert(onConflict = OnConflictStrategy.REPLACE)
    suspend fun addProject(project: Project)

    @Delete
    fun delProject(project: Project)

    @Update
    suspend fun editProject(project: Project)

    @Query("SELECT count(*) From projects")
    fun count(): Int

    @Query("SELECT * FROM projects")
    suspend fun getAllProjects(): List<Project>

    @Query("SELECT * FROM projects where favourite=1")
    suspend fun getAllFavouriteProjects(): List<Project>

    @Query("SELECT * FROM projects where id = :projId")
    suspend fun searchProjectById(projId: Long): Project

    @Query("SELECT * FROM projects WHERE title like :projTitle ")
    fun searchProjectsByTitle(projTitle:String): List<Project>

    @Update
    suspend fun updateProject(project:Project)
}
```

- The project fragment list fragment implements viewLifecycleOwner.lifecycleScope.async job which awaits in the launch scope to update the adapter project data when the show favourite button is pressed.



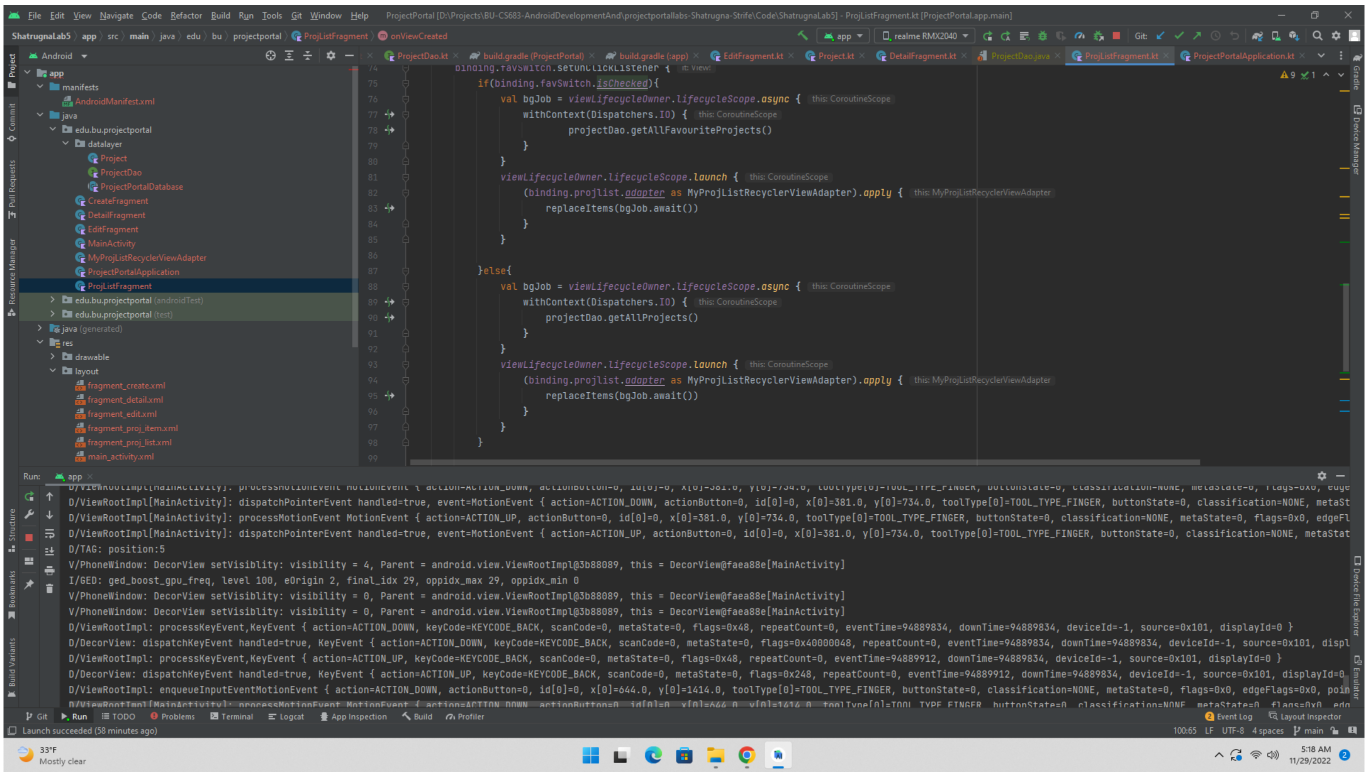
CS683 Projects

1	weather Forecast
4	hello world
5	hdhdhd

```
binding.projlist.apply{
            layoutManager = when {
                columnCount <= 1 -> LinearLayoutManager(context)
                else -> GridLayoutManager(context, columnCount)
            }
            viewLifecycleOwner.lifecycleScope.launch{
                    adapter = if(binding.favSwitch.isChecked)
MyProjListRecyclerViewAdapter(projectDao.getAllFavouriteProjects() as
MutableList<Project>) else MyProjListRecyclerViewAdapter(
                        projectDao.getAllProjects() as MutableList<Project>
                    )
            }
//              adapter = if(binding.favSwitch.isChecked)
MyProjListRecyclerViewAdapter(projectDao.getAllFavouriteProjects() as
MutableList<Project>) else MyProjListRecyclerViewAdapter(
//                  projectDao.getAllProjects() as MutableList<Project>
//              )
        }
        binding.addProject.setOnClickListener {

view.findNavController().navigate(R.id.action_projListRecycleViewFragment_to_creat
eFragment)
        }
        binding.favSwitch.setOnClickListener {
            if(binding.favSwitch.isChecked){
                val bgJob = viewLifecycleOwner.lifecycleScope.async {
                    withContext(Dispatchers.IO) {
                            projectDao.getAllFavouriteProjects()
                    }
                }
                viewLifecycleOwner.lifecycleScope.launch {
                    (binding.projlist.adapter as
MyProjListRecyclerViewAdapter).apply {
                        replaceItems(bgJob.await())
                    }
                }

            }else{
                val bgJob = viewLifecycleOwner.lifecycleScope.async {
                    withContext(Dispatchers.IO) {
                        projectDao.getAllProjects()
                    }
                }
                viewLifecycleOwner.lifecycleScope.launch {
                    (binding.projlist.adapter as
MyProjListRecyclerViewAdapter).apply {
                        replaceItems(bgJob.await())
                    }
                }
            }

            editor.putInt("fav",  if (binding.favSwitch.isChecked) 1 else 0)
            editor.apply()
            editor.commit()
        }
```

- In rest of the fragments the db calls are wrapped around launch scope of viewLifecycleOwner.lifecycleScope.

```
viewLifecycleOwner.lifecycleScope.launch {
    withContext(Dispatchers.IO) {
        projectDao.addProject(project)
    }
}
```



```
        binding.favSwitch.setOnClickListener {
            if(binding.favSwitch.isChecked){
                val bgJob = viewLifecycleOwner.lifecycleScope.async {
                    withContext(Dispatchers.IO) {
                            projectDao.getAllFavouriteProjects()
                    }
                }
                viewLifecycleOwner.lifecycleScope.launch {
                    (binding.projlist.adapter as MyProjListRecyclerViewAdapter).apply {
                        replaceItems(bgJob.await())
                    }
                }

            }else{
                val bgJob = viewLifecycleOwner.lifecycleScope.async {
                    withContext(Dispatchers.IO) {
                        projectDao.getAllProjects()
                    }
                }
                viewLifecycleOwner.lifecycleScope.launch {
                    (binding.projlist.adapter as MyProjListRecyclerViewAdapter).apply {
                        replaceItems(bgJob.await())
                    }
                }
            }
```

```
D/ViewRootImpl[MainActivity]: dispatchPointerEvent handled=true, event=MotionEvent { action=ACTION_DOWN, actionButton=0, id[0]=0, x[0]=381.0, y[0]=734.0, toolType[0]=TOOL_TYPE_FINGER, buttonState=0, classification=NONE, metaSt
D/ViewRootImpl[MainActivity]: processMotionEvent MotionEvent { action=ACTION_UP, actionButton=0, id[0]=0, x[0]=381.0, y[0]=734.0, toolType[0]=TOOL_TYPE_FINGER, buttonState=0, classification=NONE, metaState=0, flags=0x0, edgeFl
D/ViewRootImpl[MainActivity]: dispatchPointerEvent handled=true, event=MotionEvent { action=ACTION_UP, actionButton=0, id[0]=0, x[0]=381.0, y[0]=734.0, toolType[0]=TOOL_TYPE_FINGER, buttonState=0, classification=NONE, metaStat
D/TAG: position:5
V/PhoneWindow: DecorView setVisiblity: visibility = 4, Parent = android.view.ViewRootImpl@3b88089, this = DecorView@faea88e[MainActivity]
I/GED: ged_boost_gpu_freq, level 100, eOrigin 2, final_idx 29, oppidx_max 29, oppidx_min 0
V/PhoneWindow: DecorView setVisiblity: visibility = 0, Parent = android.view.ViewRootImpl@3b88089, this = DecorView@faea88e[MainActivity]
V/PhoneWindow: DecorView setVisiblity: visibility = 0, Parent = android.view.ViewRootImpl@3b88089, this = DecorView@faea88e[MainActivity]
D/ViewRootImpl: processKeyEvent,KeyEvent { action=ACTION_DOWN, keyCode=KEYCODE_BACK, scanCode=0, metaState=0, flags=0x48, repeatCount=0, eventTime=94889834, downTime=94889834, deviceId=-1, source=0x101, displayId=0 }
D/DecorView: dispatchKeyEvent handled=true, KeyEvent { action=ACTION_DOWN, keyCode=KEYCODE_BACK, scanCode=0, metaState=0, flags=0x40000048, repeatCount=0, eventTime=94889834, downTime=94889834, deviceId=-1, source=0x101, displ
D/ViewRootImpl: processKeyEvent,KeyEvent { action=ACTION_UP, keyCode=KEYCODE_BACK, scanCode=0, metaState=0, flags=0x48, repeatCount=0, eventTime=94889912, downTime=94889834, deviceId=-1, source=0x101, displayId=0 }
D/DecorView: dispatchKeyEvent handled=true, KeyEvent { action=ACTION_UP, keyCode=KEYCODE_BACK, scanCode=0, metaState=0, flags=0x248, repeatCount=0, eventTime=94889912, downTime=94889834, deviceId=-1, source=0x101, displayId=0
D/ViewRootImpl: enqueueInputEventMotionEvent { action=ACTION_DOWN, actionButton=0, id[0]=0, x[0]=644.0, y[0]=1414.0, toolType[0]=TOOL_TYPE_FINGER, buttonState=0, classification=NONE, metaState=0, flags=0x0, edgeFlags=0x0, poin
```

Launch succeeded (58 minutes ago)

Shatrugna Rao Korukanti
U43517028