Phát Triển Ứng Dụng

Báo cáo bài tập Thực Hành

Thông tin sinh viên

- Họ tên: Hà Bảo Anh
- Mã sinh viên: 19442001
- Link github repository: https://github.com/baoanhcr7/19442001_HaBaoAnh_AD_TodoApp.git

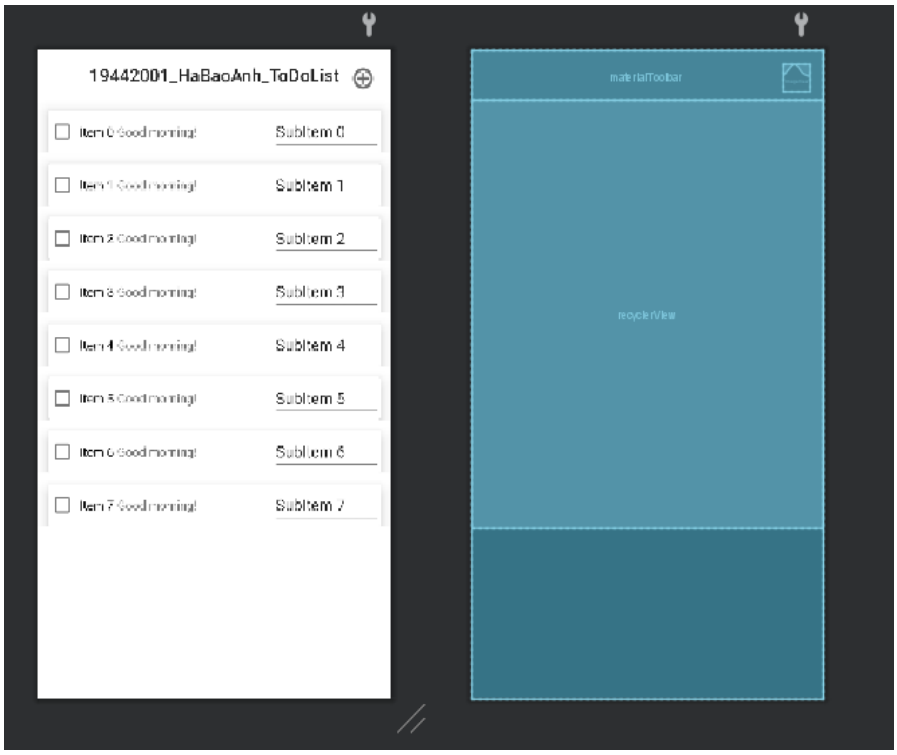
Quá trình thực hiện

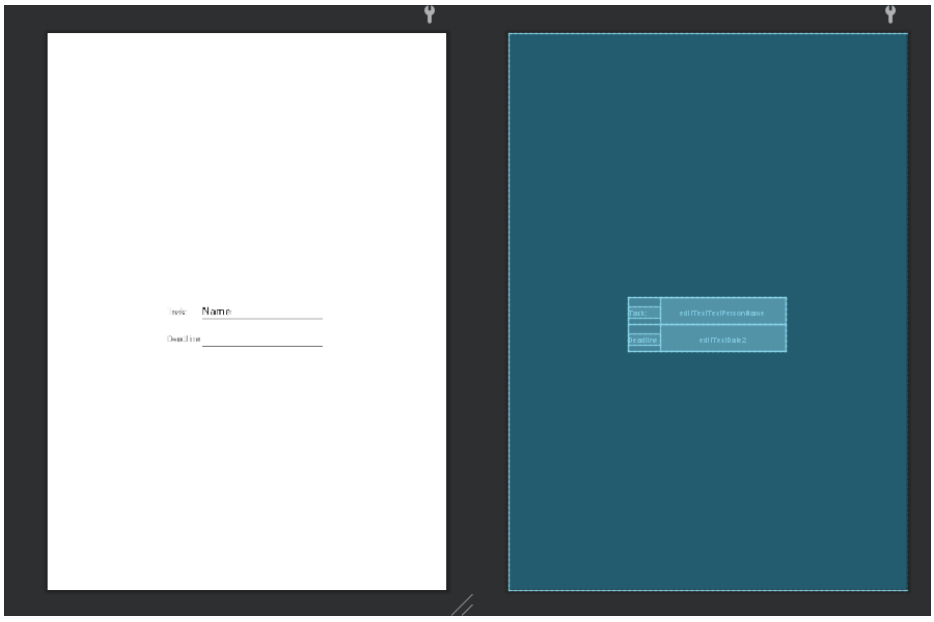
1. Trình bày ý tưởng thực hiện
    - Xây dựng ứng dụng To Do List để lưu thông tin các công việc cần làm. Ứng dụng có chức năng thêm, xóa, cập nhật danh sách các công việc. Thông tin các công việc bao gồm mô tả công việc và trạng thái của công việc đó. Ngoài ra, ứng dụng phải kết nối với firebase để lưu trữ các cập nhật của người dùng.

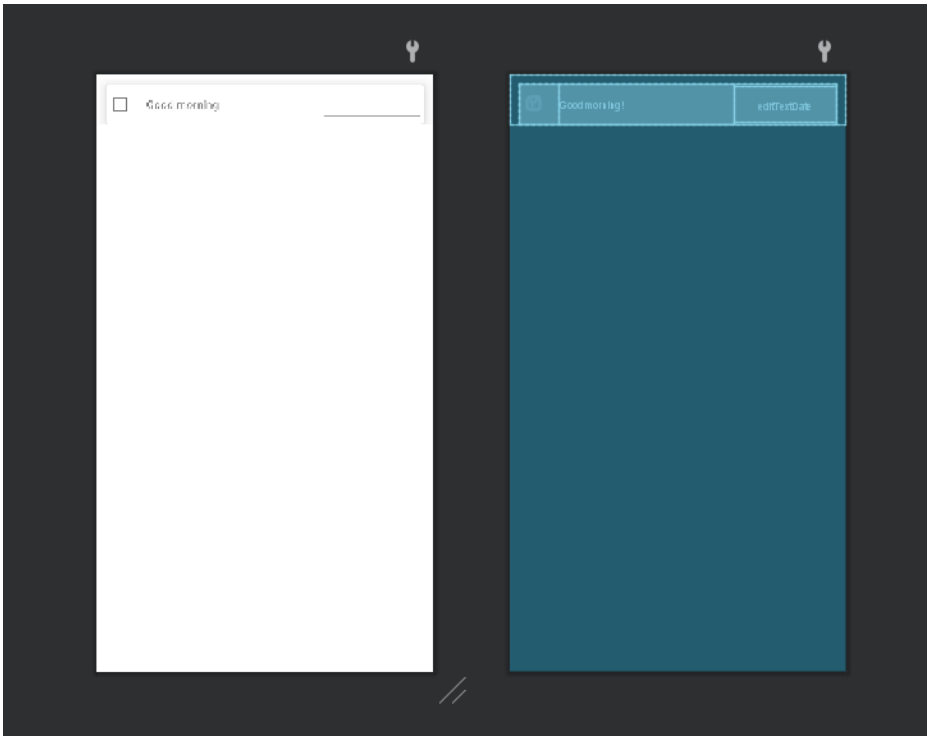
    Về phần layout, cần 3 layout chính:
        - Để hiện thị giao diện chính cho người dùng thao tác.
        - Để quy định layout cho 1 task hiện thị.
        - Để hiện quy định giao diện nhập thông tin

    Các file layout:
        - activity_main.xml: Giao diện chính.
        - layout_a_task.xml: Hiện thị 1 task nhỏ.
        - layout_add_task.xml: Giao diện thêm task qua dialog

2. Giao diện của ứng dụng







- Giao diện chính có thể cho người dùng danh sách công việc hiện tại và timeline
- Có nút thêm công việc vào danh sách

3. Các lớp chức năng chính trong ứng dụng
   - Thêm 1 task sẽ thực hiện bằng việc nhấn vào 1 nút bấm add, mặc định khi mới tạo của 1 task sẽ là *done = false*.

Lớp MainActivity

Khởi tạo Adapter và gọi hàm refresh() khi giao diện vừa khởi tạo

```
class MainActivity : AppCompatActivity() {
    val cal: Calendar = Calendar.getInstance()
    val tasks = ArrayList<Task>()
    lateinit var adapter : TaskAdapter

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        adapter = TaskAdapter( context: this, tasks)
        recyclerView.adapter = adapter
        val format = SimpleDateFormat( pattern: "dd-MM-yyyy", Locale.ROOT)

        refresh()
```

Khởi tạo dialog và gán layout

```
btnAdd.setOnClickListener {  it: View!
    val builder = AlertDialog.Builder( context: this)
        .setTitle("Add a new task")
        .setCancelable(true)
    val view = LayoutInflater.from( context: this).inflate(R.layout.layout_add_task, root,  attachToRoot: false)
```

Hàm refresh()

"Xóa các phần tử trong mảng
Lấy các doc từ firestore đã được cập nhật lại
***Index task** với **done** là **false** thì chỉ hiện thỉ những task chưa hoàn thành"*

```
fun refresh() {
    tasks.clear()
    adapter.notifyDataSetChanged()
    Task.get()
        .addOnSuccessListener { query ->
            val documents = query.documents
            for (doc: DocumentSnapshot in documents) {
                val task = Task(doc)
                if (task.done == false) {
                    tasks.add(task)
                    adapter.notifyItemInserted( position: tasks.size - 1)
                }
            }
        }
}
```

Lớp Task

Truyền dữ liệu đã lấy từ doc của firestore

MainActivity.kt | Task.kt | build.gradle (19442001_HaBaoAnh_AD_HW2) | build.gradle (:app)

```
1   package com.example.a19442001_habaoanh_ad_hw2
2
3   import ...
10
11  class Task() {
12      var id: String? = null
13      var task: String? = null
14      var deadline: Date? = null
15      var done: Boolean? = false
16
17      constructor(task: String, deadline: Date, done: Boolean) : this() {
18          this.task = task
19          this.deadline = deadline
20          this.done = done
21      }
22
23      constructor(doc: DocumentSnapshot) : this() {
24          id = doc.id
25          task = doc.getString( field: "task")
26          deadline = doc.getDate( field: "deadline")
27          done = doc.getBoolean( field: "done")
28      }
```

Hàm Set() đữ liệu

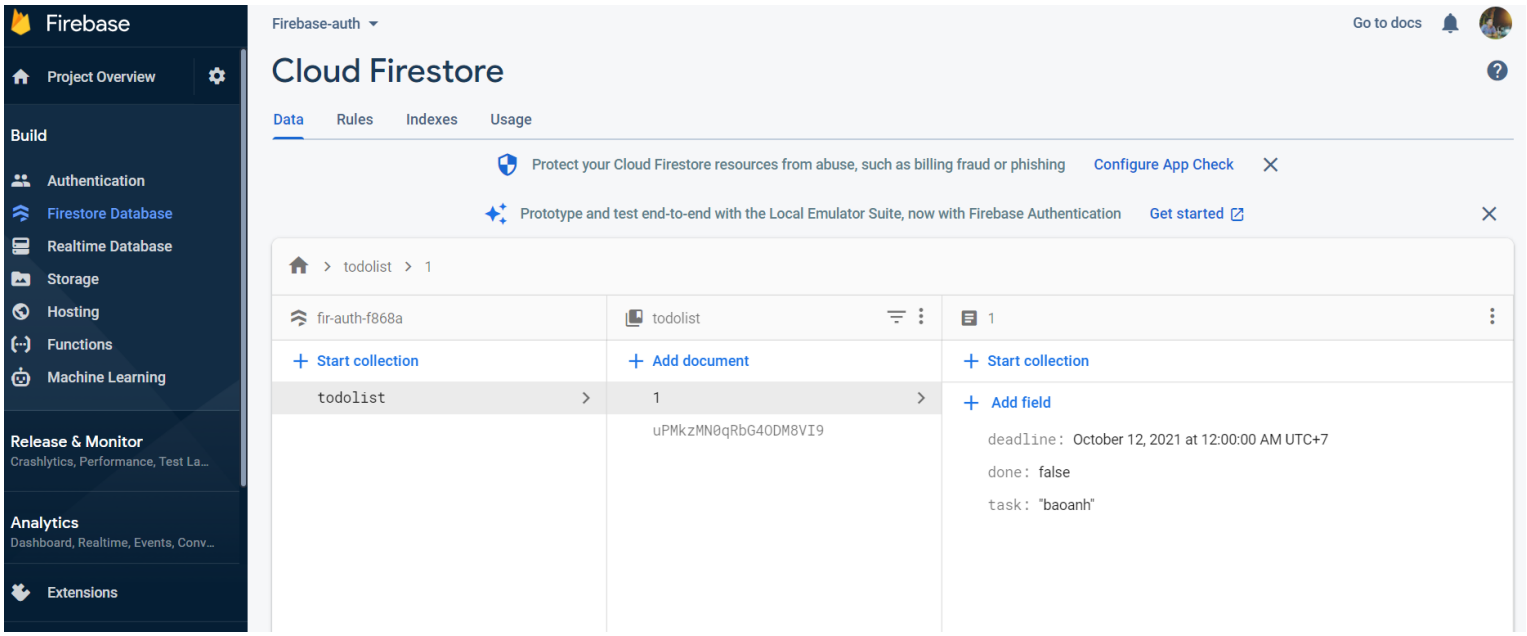
```
fun set() {
    if (id != null) {
        Firebase.firestore.collection( collectionPath: "todolist").document(id!!)
            .set(
                hashMapOf(
                    "task" to task,
                    "deadline" to deadline,
                    "done" to done
                )
            )
    } else {
        Firebase.firestore.collection( collectionPath: "todolist")
            .add(
                hashMapOf(
                    "task" to task,
                    "deadline" to deadline,
                    "done" to done
                )
            )
    }
}
```

Hàm get()

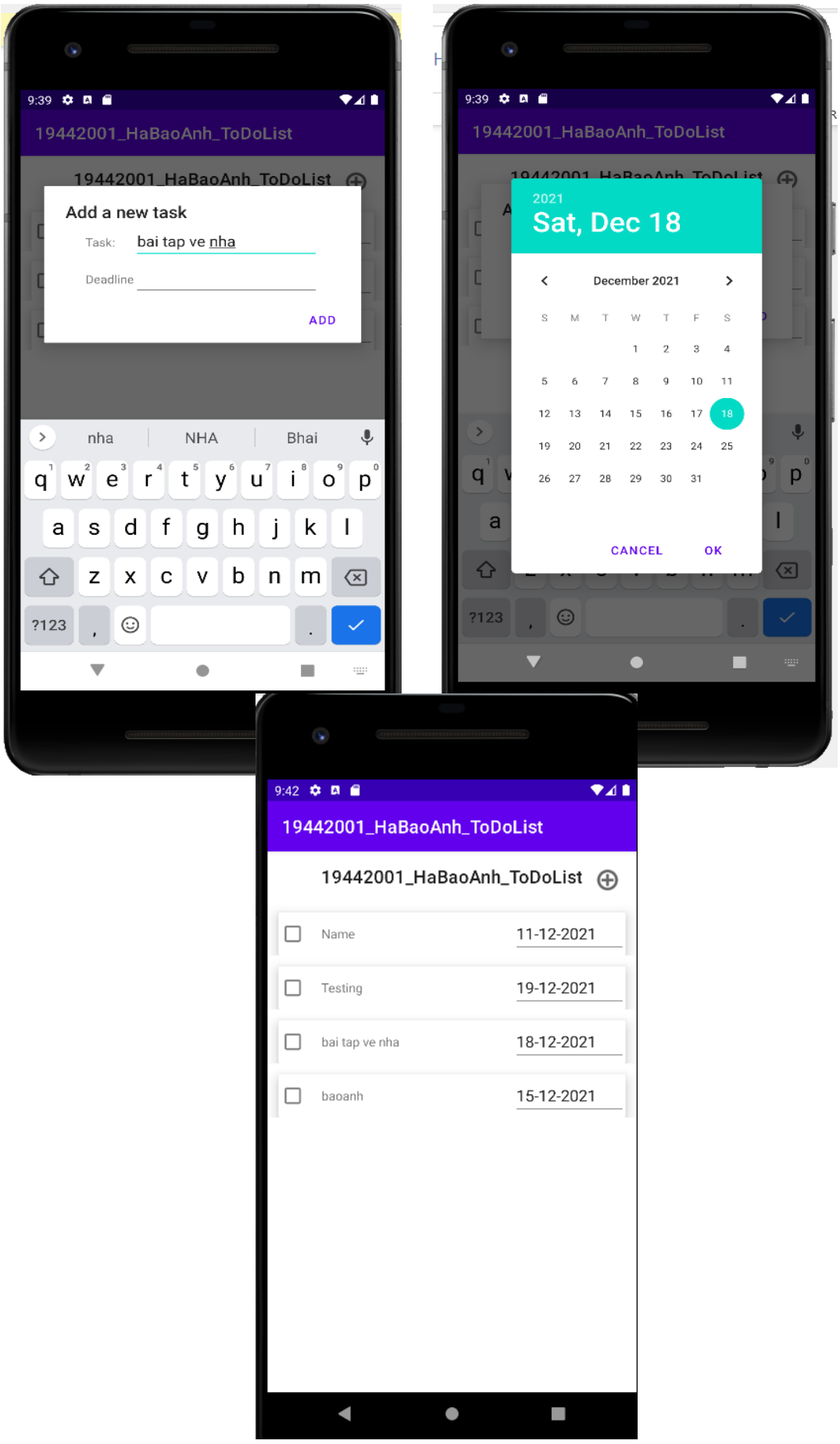
- get dữ liệu từ collection không theo order
- get dữ liệu từ collection order
- get theo ID của document

```
companion object {
    fun get() : Task<QuerySnapshot> {
        return Firebase.firestore.collection( collectionPath: "todolist").get()
    }

    fun getRecent() : Task<QuerySnapshot> {
        // Get 8 newest products
        return Firebase.firestore.collection( collectionPath: "todolist")
            .orderBy( field: "deadline", Query.Direction.DESCENDING)
            .get()
    }

    fun get(id : String) : Task<DocumentSnapshot> {
        return Firebase.firestore.collection( collectionPath: "todolist").document(id).get()
    }
}
```

4. Kết nối dữ liệu với firebase



5. Hiển thị kết quả chạy chương trình



Kết quả thực hiện

1. Làm được:
    - Làm được chức năng thêm
    - Setup thành công firestore.
2. Chưa làm được:
    - Chức năng xóa.
3. Khó khăn gặp phải trong quá trình làm:
    - Học 1 ngôn ngữ mới Kotlin
    - Kiến thức về lập trình ứng dụng và lập trình hướng sự kiện chưa học nên phải tự tìm hiểu.