Experiment No. 8

Aim: To implement Inventory Tracking App in Android.

Requirements: Compatible version of Android Studio.

Theory:

This app is an Inventory tracking app. Demos how to add, update, sell, and delete items from the local database.

This app demonstrated

Room database

Apps that handle non-trivial amounts of structured data can benefit greatly from persisting that data locally. The most common use case is to cache relevant pieces of data so that when the device cannot access the network, the user can still browse that content while they are offline.

The Room persistence library provides an abstraction layer over SQLite to allow fluent database access while harnessing the full power of SQLite. In particular, Room provides the following benefits:

- Compile-time verification of SQL queries.
- Convenience annotations that minimize repetitive and error-prone boilerplate code.
- Streamlined database migration paths.

View Model

Architecture Components provides ViewModel helper class for the UI controller that is responsible for preparing data for the UI. ViewModel objects are automatically retained during

configuration changes so that data they hold is immediately available to the next activity or fragment instance.

Flow

A flow is very similar to an Iterator that produces a sequence of values, but it uses suspend functions to produce and consume values asynchronously. This means, for example, that the flow can safely make a network request to produce the next value without blocking the main thread.

View Binding

View binding is a feature that allows you to more easily write code that interacts with views. Once view binding is enabled in a module, it generates a *binding class* for each XML layout file present in that module. An instance of a binding class contains direct references to all views that have an ID in the corresponding layout.

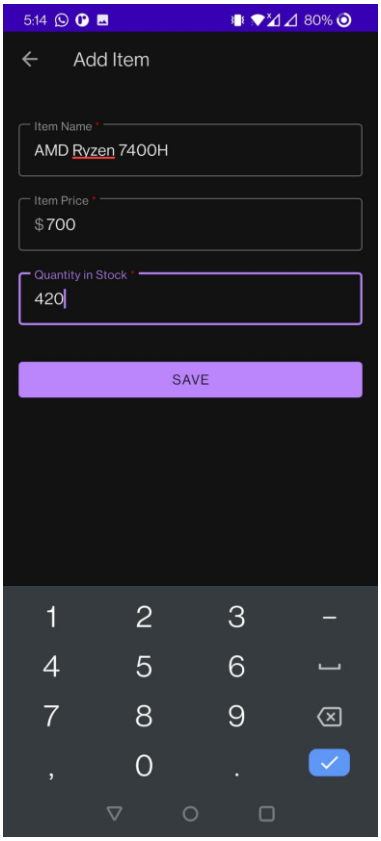
Navigation

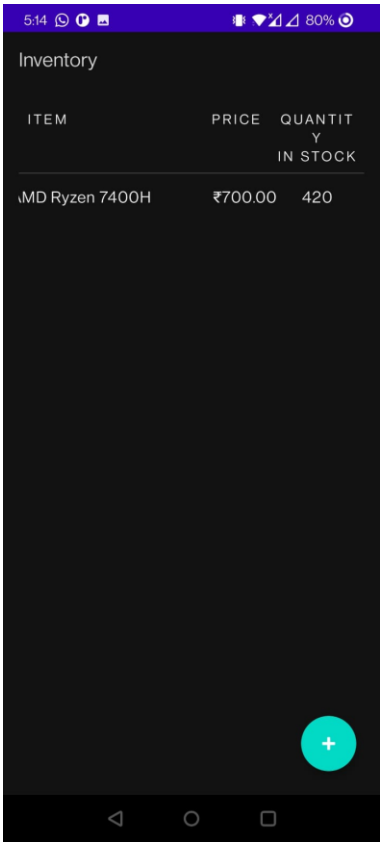
Navigation refers to the interactions that allow users to navigate across, into, and back out from the different pieces of content within your app. Android Jetpack's Navigation component helps you implement navigation, from simple button clicks to more complex patterns, such as app bars and the navigation drawer. The Navigation component also ensures a consistent and predictable user experience by adhering to an established set of principles.

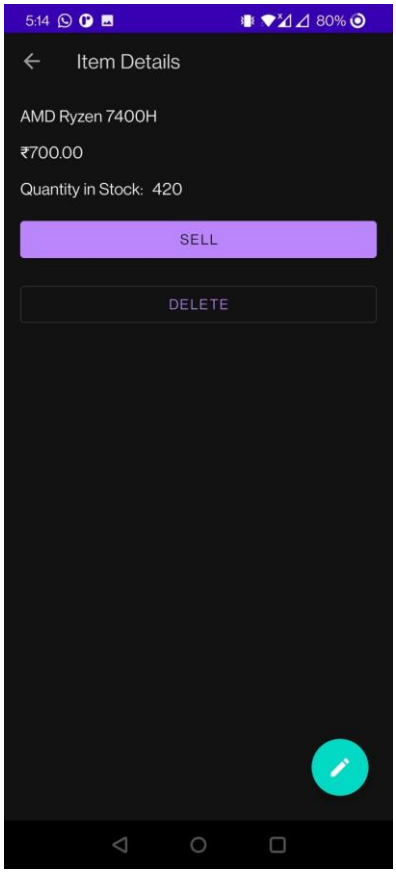
Code:

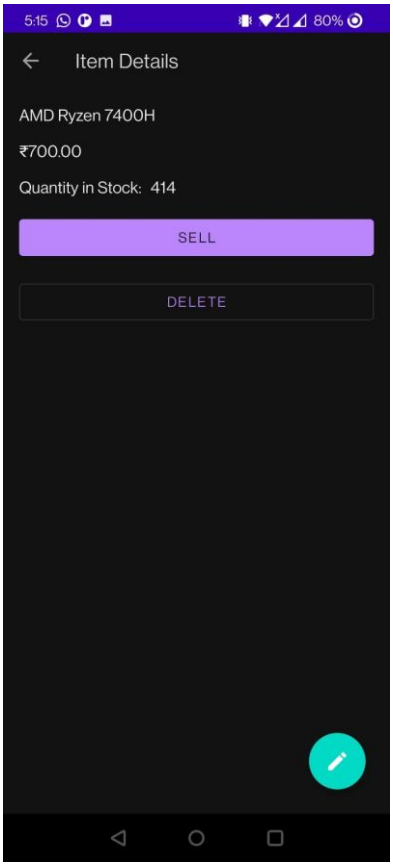
```
class MainActivity : AppCompatActivity(R.layout.activity_main) {

    private lateinit var navController: NavController

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)

        // Retrieve NavController from the NavHostFragment
        val navHostFragment = supportFragmentManager
            .findFragmentById(R.id.nav_host_fragment) as NavHostFragment
        navController = navHostFragment.navController
        // Set up the action bar for use with the NavController
        setupActionBarWithNavController(this, navController)
    }

    /**
     * Handle navigation when the user chooses Up from the action bar.
     */
    override fun onSupportNavigateUp(): Boolean {
        return navController.navigateUp() || super.onSupportNavigateUp()
    }
}
```

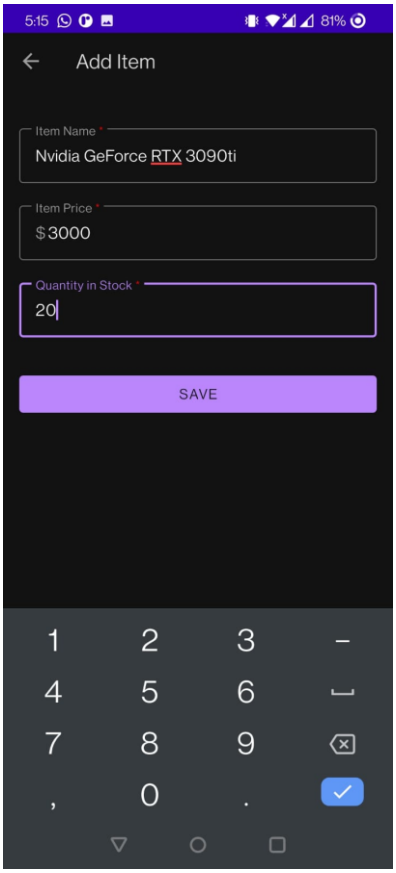
Output:

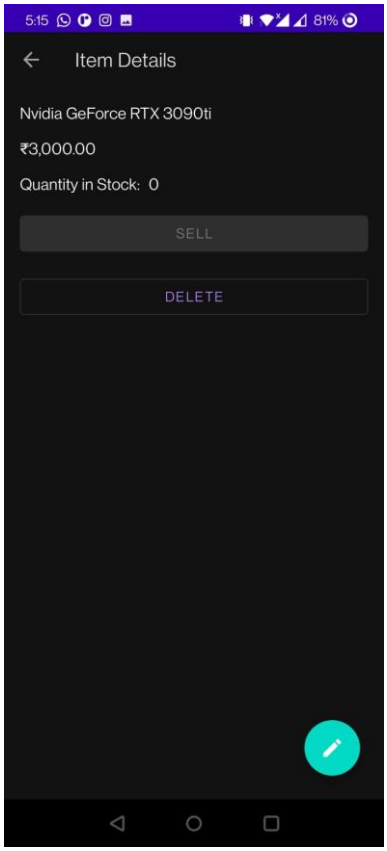












Conclusion: We have successfully implemented Inventory Tracking App in Android.