# Android Programming Deliverable Documentation

**Name:Chege Mbae**

**Reg No: ICT-04-2800-18.**

This is a simple app that servers as note taker. As expected in this unit, I applied all the learned skills in the coursework and I finally implemented this application.

I have branded it as Event Logger. This is since the application stores the event a user writes. One can view past written event and also edit them as one likes.

A new event can also be added. All events are displayed to the main-screen.

## How it works

The app launches the main-screen and within it, is a list of previously written event, if there is no previous event the screen is empty.

Also within the main-screen is a floatingButton that acts a link to the Activity involved in creating new event.

The floatingButton opens an activity that picks input from user and forwards it to database for storage.

The events are stored in SQLITE database.

### Main Screen

Its appearance is as displayed below, For a fresh install it is empty(no previous events are rendered).

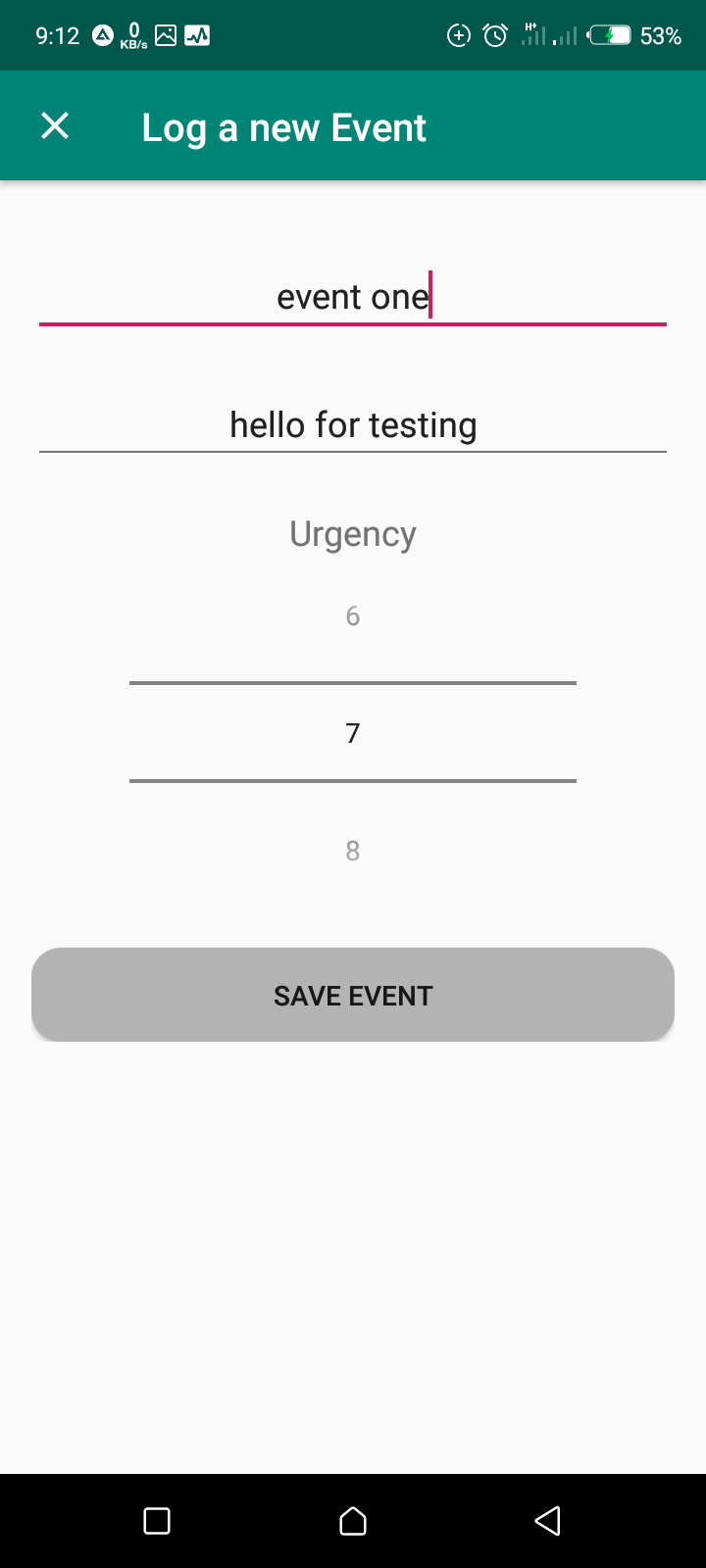


### Add Event Screen

Its appearance is as displayed below,

Has three parameters which only two must be filled, that is for Event Body and Event Title, for Event urgency the it has a default value assigned to it.

The Button at the end invokes the function that saves the written event.



### Edit Event Screen

Its appearance is as displayed below, it uses similar display (xml file) with Add Event Screen.

Has three parameters which only two must be filled, that is for Event Body and Event Title, for Event urgency the it has a default value assigned to it.

The three parameter are already filled, since it picks the values of event and populates them to the input fields above.

The Button at the end invokes the function that saves the written event.

## Features within the app

The app has the following feature to itself

* Add New Note
* Edit Note
* Delete Single Event

→ By swiping the event that is to be deleted

* Delete all Events

→ Using the overflow menu and select the “Remove All Events”

* Dark Theme Toggle

→ Using the overflow menu and delete “Dark Theme”

* Real Time updates

→ The app automatically updates without needing a refresh.

## Application Logic

The application uses the MVVM architecture (Model-View-ViewModel). The data source (Database) is the Model. The Activities with UI are the View and the ViewModel is the **Event\_ViewModel**

I used the following architecture components

* Life Cycles

→ Creating UI that responds to lifecycle events.

* ViewModel

→ Stores UI-related data. And schedules asynchronous tasks in the backgound

* Room

→ It access the SQLITE database and performs compile-time checks. Ensure compiled app has no SQLITE related errore.

* Live Data

→ Builds data objects. These objects inform the view of database changes

The Application has one parent(MainActivity) class, the the other activities (Add\_Event, Edit\_Event) are subject to, as show from the manifest file below

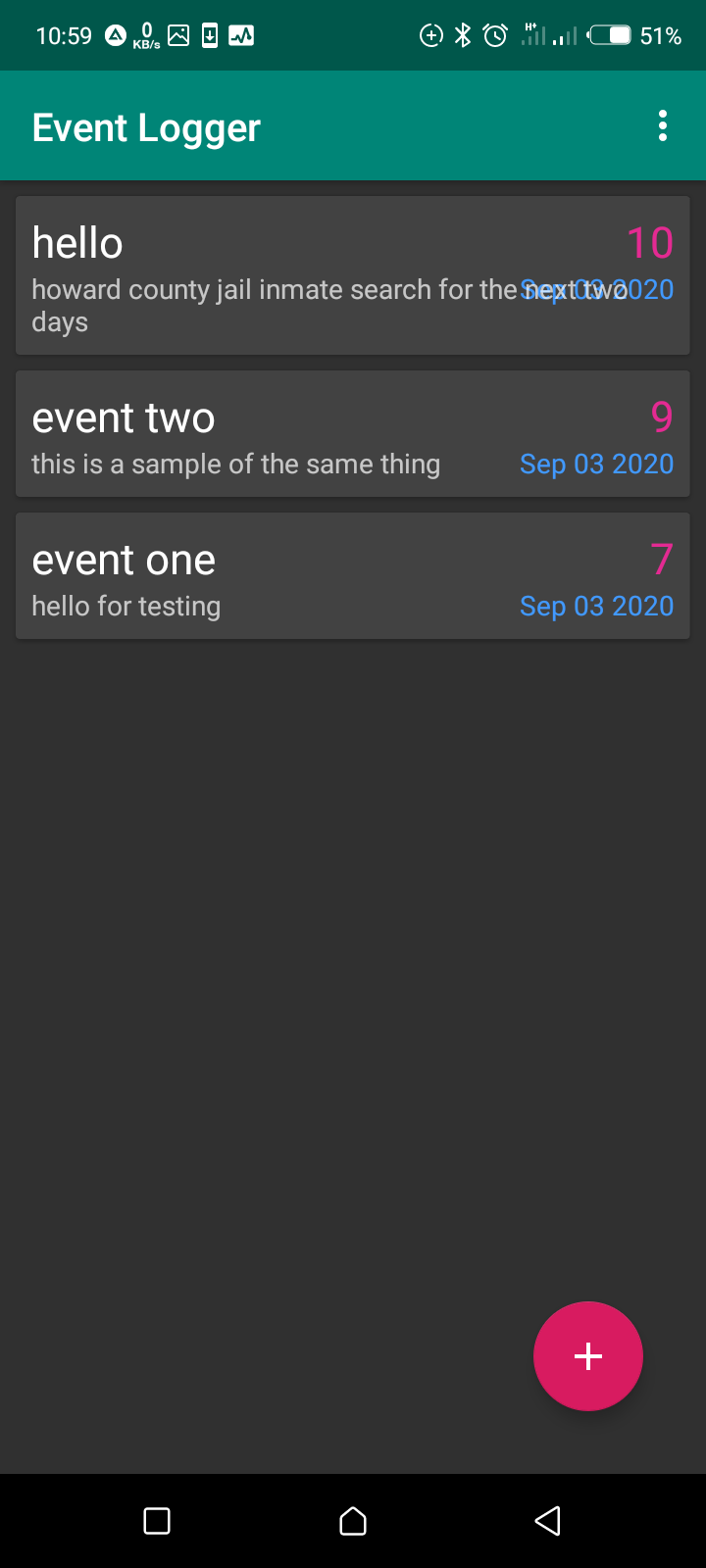
The **MainActivity** has a RecyclerView draws the list of all elements from the data as specified by the adapter class (Event Adapter).

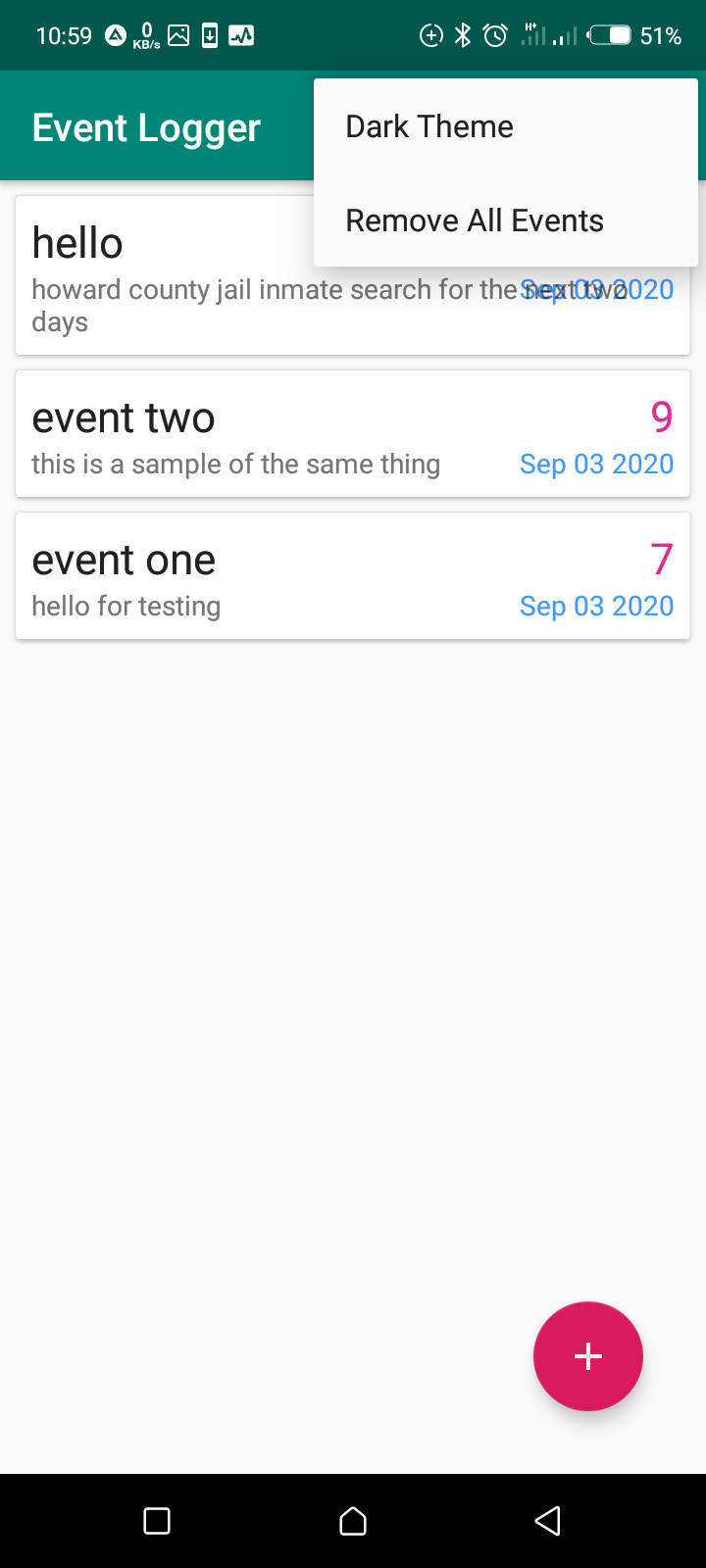
The **Event\_Adapter** this is a class that inherits from RecyclerView.Adapter, it specifies how to populate the Event Note.

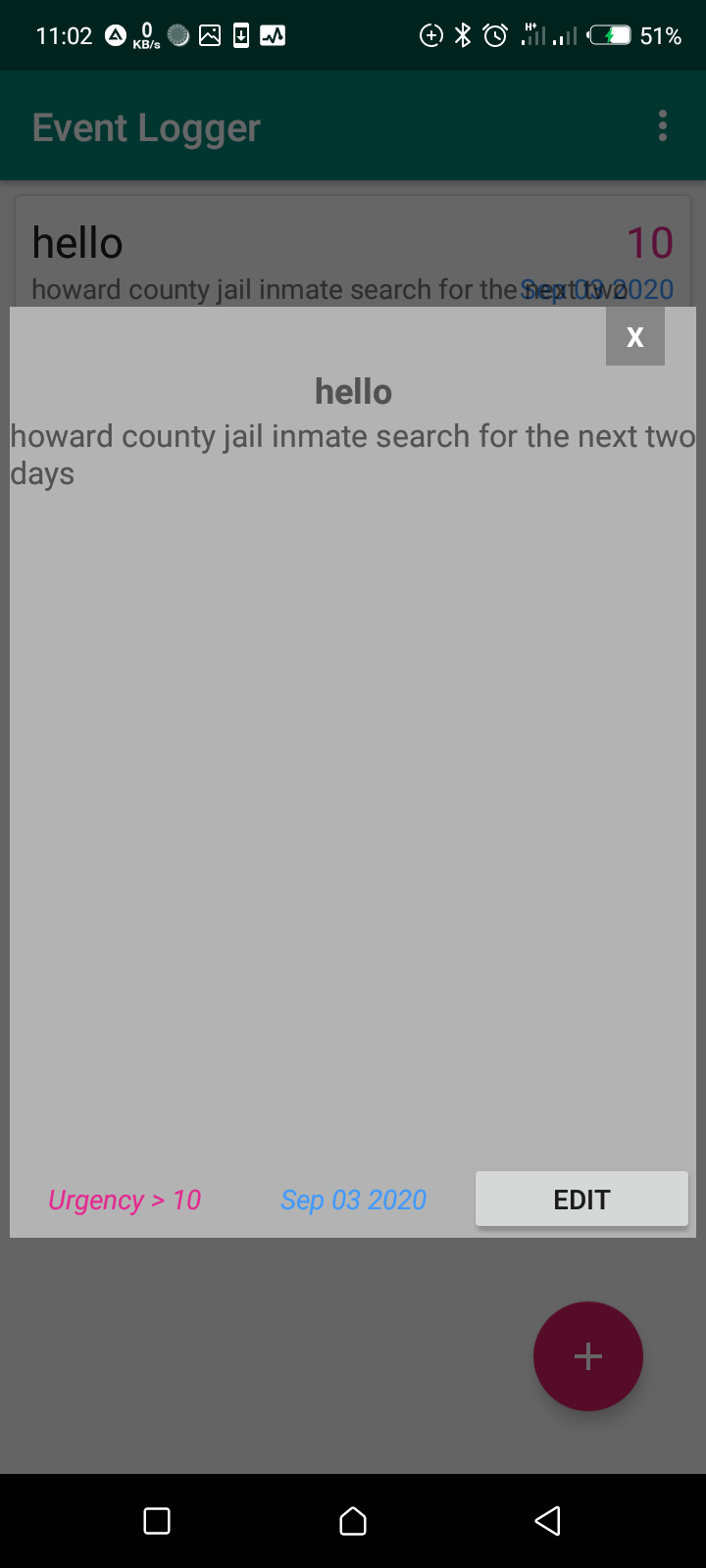
The **Event\_Dao** this is a class from Room persistence library, its abbreviation for Database Access Object. Has the functions/queries to run.

The **Event\_ViewModel** this is a class for holding and preparing all data for the User Interface. This is the source of data for recycler objects.

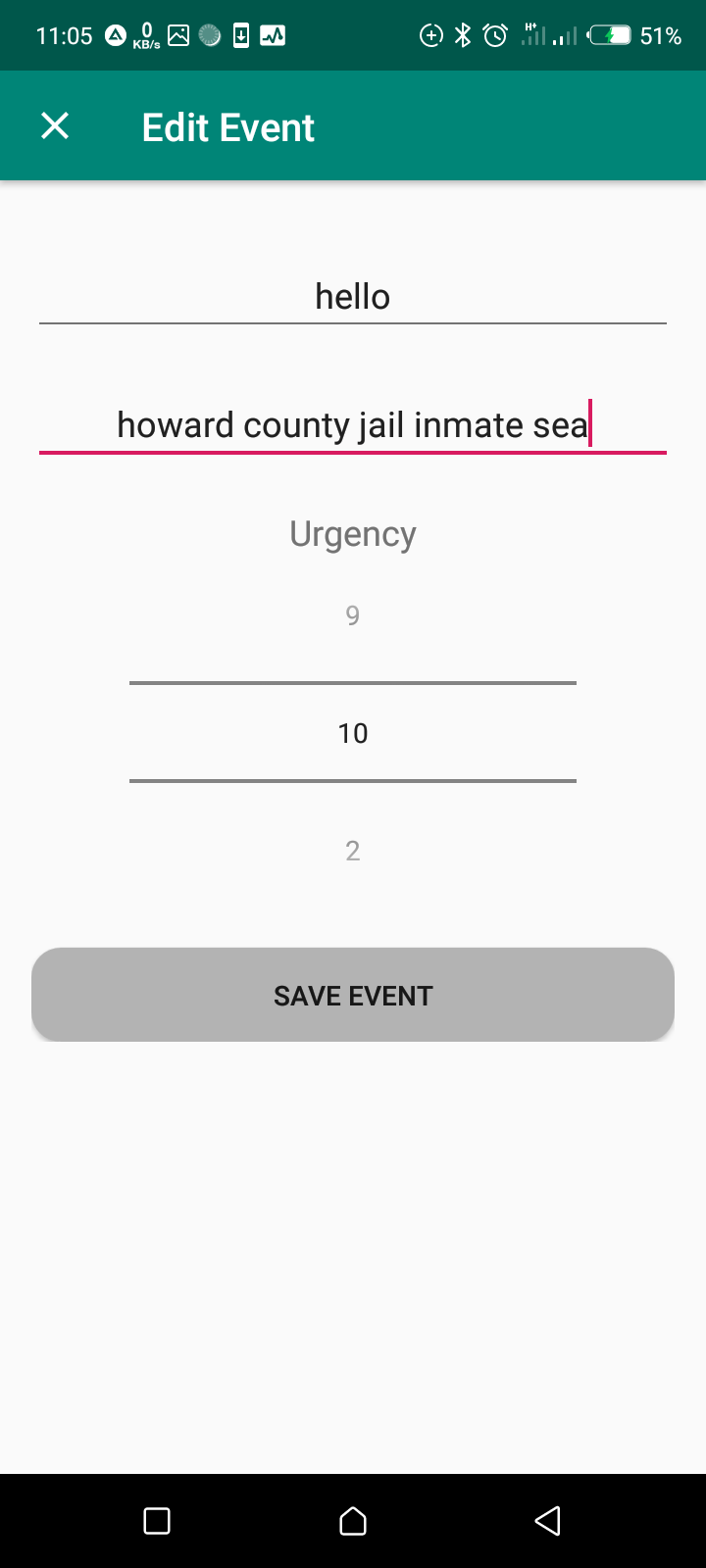
The **Event\_Repo** this is a class is used for mediating between **Event\_ViewModel** and the **Event\_Dao**.

Dark Mode toggle

Overflow menu

Viewing an event

Edit Event



Source code for inspection is here

<https://github.com/Niccher/Event-Taker>

Apk , This Documentation and Compressed Source Code

<https://github.com/Niccher/Event-Taker/Deliverables/>