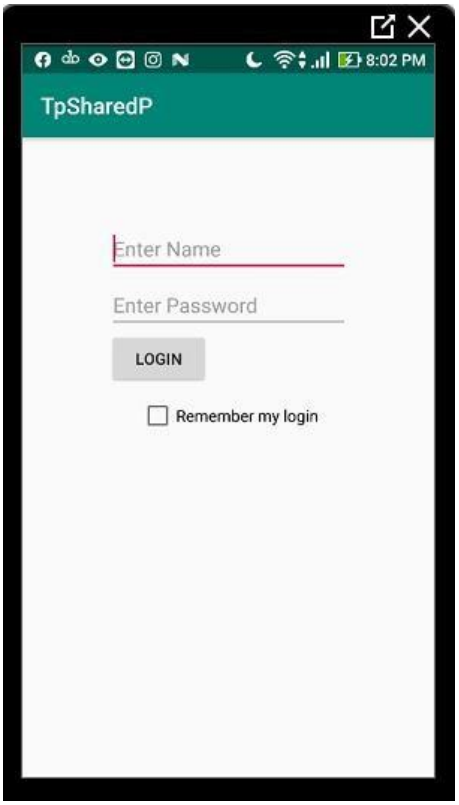
TP N°5

Atelier Shared Preferences

Il s'agit de réaliser l'application suivante :



Partie I (SharedPrefences):

1. Créer un projet nommé TPSharedP
2. Remplir le code xml de l'interface principale de l'application contenant deux EditText, un Button et un chekbox.
3. Dans la classe principale déclarer les objets suivants :
4. Pour réaliser cette application utiliser ajouter la classe kotlin suivante à votre projet :

```
class SharedPreference(val context: Context) {
//Déclaration de mon fichier shared preferences
private val PREFS_NAME = "MonFichierShared"
   val sharedPref: SharedPreferences =
context.getSharedPreferences(PREFS_NAME, Context.MODE_PRIVATE)
```



```
//Fonction pour sauvegarder une valeur string dans mon fichier shared
fun save(KEY_NAME: String, text: String) {
       val editor: SharedPreferences.Editor = sharedPref.edit()
editor.putString(KEY_NAME, text)
       editor!!.commit()
   }

//Fonction pour sauvegarder une valeur int dans mon fichier shared
fun save(KEY_NAME: String, value: Int) {
       val editor: SharedPreferences.Editor = sharedPref.edit()
editor.putInt(KEY_NAME, value)
       editor.commit()
   }

//Fonction pour sauvegarder une valeur booléenne dans mon fichier shared
fun save(KEY_NAME: String, status: Boolean) {       val editor:
SharedPreferences.Editor = sharedPref.edit()
editor.putBoolean(KEY_NAME, status!!)
       editor.commit()
   }

  //Fonction pour lire une valeur string depuis mon fichier shared
fun getValueString(KEY_NAME: String): String? {
       return sharedPref.getString(KEY_NAME, null)
   }

//Fonction pour lire une valeur int depuis mon fichier shared
fun getValueInt(KEY_NAME: String): Int {
       return sharedPref.getInt(KEY_NAME, 0)
   }

//Fonction pour lire une valeur booléenne depuis mon fichier shared  fun
getValueBoolien(KEY_NAME: String, defaultValue: Boolean): Boolean {
return sharedPref.getBoolean(KEY_NAME, defaultValue)
   }

//Fonction pour effacer les valeurs de mon fichier shared
fun clearSharedPreference() {
       val editor: SharedPreferences.Editor = sharedPref.edit()
       //sharedPref = PreferenceManager.getDefaultSharedPreferences(context);
editor.clear()
       editor.commit()
   }
```

```
//Fonction pour effacer une valeur de mon fichier shared à partir de sa clé
    fun removeValue(KEY_NAME: String) {
        val editor: SharedPreferences.Editor = sharedPref.edit()
        editor.remove(KEY_NAME)
        editor.commit()
    }
}
```

5. Utliser les methodes de cette classe pour realizer l'application demandée

6. Ajouter un menu dans l'action bar « couleur de thème » qui permet de choisir une couleur de background du bouton et la sauvegarder dans les preférences.

Partie I (DataStore):

Réaliser la même application avec DataStore au lieu de SharedPreferences en s'inspirant du code suivant :

```
// 1. INITIALISATION
    private val Context.monDataStore by preferencesDataStore("mes_donnees")

// 2. ÉCRIRE
suspend fun sauvegarder(valeur: String) {
    dataStore.edit { prefs ->
        prefs[stringPreferencesKey("ma_cle")] = valeur
    }
}

// 3. LIRE
fun lire(): Flow<String> {
    return dataStore.data.map { prefs ->
        prefs[stringPreferencesKey("ma_cle")] ?: ""
    }
}

// 4. UTILISER
        // Sauvegarder au clic
            lifecycleScope.launch { sauvegarder(texte) }
      // Afficher la valeur
            lifecycleScope.launch { lire().collect { afficher(it) } }
```