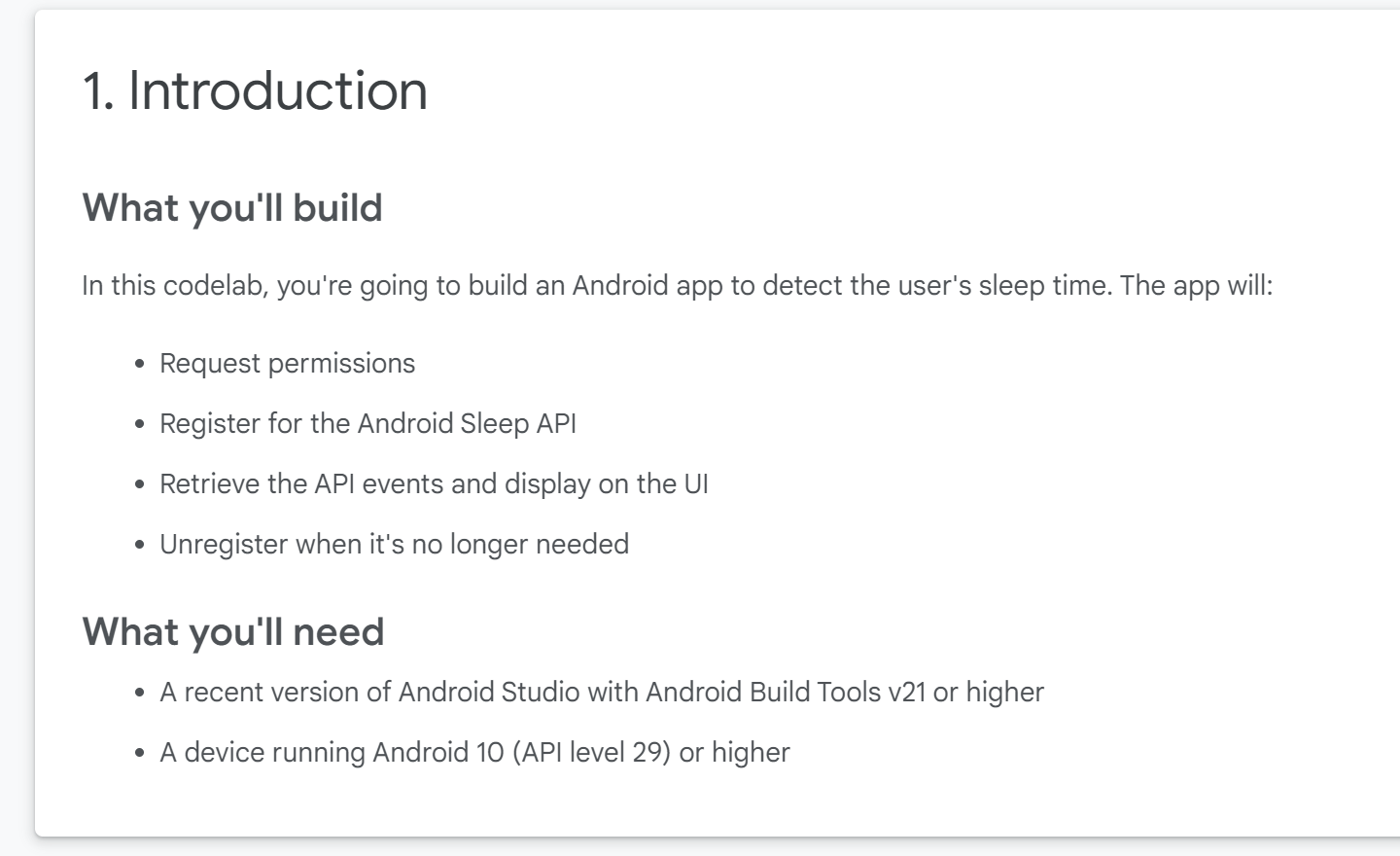


git clone https://github.com/android/codelab-android-sleep/

// TODO: Review play services library required for activity recognition.  
    implementation 'com.google.android.gms:play-services-location:18.0.0'

<!-- TODO: Add activity recognition and receive boot complete permissions. -->  
<!-- Required for 29+. -->  
<uses-permission android:name="android.permission.ACTIVITY\_RECOGNITION" />  
<uses-permission android:name="android.permission.RECEIVE\_BOOT\_COMPLETED" />

<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
         package="com.android.example.sleepcodelab">  
...  
<!-- TODO: Add activity recognition and receive boot complete permissions. -->  
<!-- Required for 29+. -->  
<uses-permission android:name="android.permission.ACTIVITY\_RECOGNITION" />  
<uses-permission android:name="android.permission.RECEIVE\_BOOT\_COMPLETED" />  
  ...  
</manifest>

// TODO: Review Activity Recognition permission checking.  
private fun activityRecognitionPermissionApproved(): Boolean {  
   return PackageManager.PERMISSION\_GRANTED == ActivityCompat.checkSelfPermission(  
           this,  
           Manifest.permission.ACTIVITY\_RECOGNITION  
   );  
}

// TODO: Enable/Disable sleep tracking and ask for permissions if needed.  
fun onClickRequestSleepData(view: View) {  
   if (activityRecognitionPermissionApproved()) {  
       if (subscribedToSleepData) {  
           unsubscribeToSleepSegmentUpdates(applicationContext, sleepPendingIntent)  
       } else {  
           subscribeToSleepSegmentUpdates(applicationContext, sleepPendingIntent)  
       }  
   } else {  
       requestPermissionLauncher.launch(permission.ACTIVITY\_RECOGNITION)  
   }  
}

// TODO: Create a PendingIntent for Sleep API events  
sleepPendingIntent =  
   SleepReceiver.createSleepReceiverPendingIntent(context = applicationContext)

// TODO: Request Sleep API updates  
val task = ActivityRecognition.getClient(context).requestSleepSegmentUpdates(  
   pendingIntent,  
   // Registers for both SleepSegmentEvent and SleepClassifyEvent data.  
   SleepSegmentRequest.getDefaultSleepSegmentRequest()  
)  
  
task.addOnSuccessListener {  
   mainViewModel.updateSubscribedToSleepData(true)  
   Log.d(TAG, "Successfully subscribed to sleep data.")  
}  
task.addOnFailureListener { exception ->  
   Log.d(TAG, "Exception when subscribing to sleep data: $exception")  
}

// TODO: Request Sleep API upon boot complete  
val subscribedToSleepData = repository.subscribedToSleepDataFlow.first()  
if (subscribedToSleepData) {  
   subscribeToSleepSegmentUpdates(  
       context = context,  
       pendingIntent = SleepReceiver.createSleepReceiverPendingIntent(context)  
   )  
}

// TODO: Extract sleep information from PendingIntent.  
if (SleepSegmentEvent.hasEvents(intent)) {  
   val sleepSegmentEvents: List<SleepSegmentEvent> =  
       SleepSegmentEvent.extractEvents(intent)  
   addSleepSegmentEventsToDatabase(repository, sleepSegmentEvents)  
} else if (SleepClassifyEvent.hasEvents(intent)) {  
   val sleepClassifyEvents: List<SleepClassifyEvent> =  
       SleepClassifyEvent.extractEvents(intent)  
   addSleepClassifyEventsToDatabase(repository, sleepClassifyEvents)  
}

