```
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>

<application
    android:usesCleartextTraffic="true" />
```

```
app/
└── src/main/assets/maps/offline_map.tif
```

GeoTiffUtils.kt

```
/**
 * Copies GeoTIFF from assets to app-specific external storage.
 * This is required because ArcGIS needs a real file path.
 */
fun copyGeoTiffFromAssets(
    context: Context,
    fileName: String
): String {

    // Folder: /Android/data/your.package/files/maps/
    val outputDir = File(context.getExternalFilesDir(null), "maps")

    if (!outputDir.exists()) {
        outputDir.mkdirs()
    }

    val outFile = File(outputDir, fileName)

    // Copy only once
    if (!outFile.exists()) {
        context.assets.open("maps/$fileName").use { input ->
            FileOutputStream(outFile).use { output ->
                input.copyTo(output)
            }
        }
    }

    return outFile.absolutePath
}
```

GeoTiffMapFactory.kt

```
/**
 * Creates an ArcGISMap using an offline GeoTIFF raster.
 */
fun createGeoTiffMap(tifPath: String): ArcGISMap {

    // Load raster from file
    val raster = Raster(tifPath)

    // Wrap raster inside a layer
    val rasterLayer = RasterLayer(raster)

    // Create empty map (no basemap, fully offline)
    val map = ArcGISMap()

    // Add raster as operational layer
    map.operationalLayers.add(rasterLayer)

    return map
}
```

GeoTiffMapScreen.kt

```
@Composable
fun GeoTiffMapScreen(tifPath: String) {

    val context = LocalContext.current

    AndroidView(
        modifier = Modifier.fillMaxSize(),
        factory = {

            // Create ArcGIS MapView
            val mapView = MapView(context)

            // Create offline raster map
            val map = createGeoTiffMap(tifPath)
            mapView.map = map

            // Enable gestures (zoom, pan)
            mapView.interactionOptions.isZoomEnabled = true
            mapView.interactionOptions.isPanEnabled = true

            // ——————————————————————————————
            // SHOW CURRENT LOCATION DOT
            // ——————————————————————————————

            val locationDisplay = mapView.locationDisplay
```

```
            // Use Android GPS (offline)
            locationDisplay.autoPanMode =
                LocationDisplay.AutoPanMode.RECENTER

            // Start GPS tracking
            locationDisplay.startAsync()

            // ──────────────────────────────────────
            // ZOOM TO GEO-TIFF EXTENT
            // ──────────────────────────────────────

            map.operationalLayers.first().addDoneLoadingListener {

                val rasterLayer =
                    map.operationalLayers.first() as RasterLayer

                // Zoom map to raster boundaries
                mapView.setViewpointGeometryAsync(
                    rasterLayer.fullExtent,
                    50.0 // padding
                )
            }

            mapView
        }
    )
}

class MainActivity : ComponentActivity() {

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

        // Copy GeoTIFF once and get file path
        val tifPath = copyGeoTiffFromAssets(
            context = this,
            fileName = "offline_map.tif"
        )

        setContent {
            MaterialTheme {
                GeoTiffMapScreen(tifPath)
            }
        }
    }
}
```

GeoTiffUtils.kt

```
// Copies GeoTIFF from assets to app storage (run once)
fun copyGeoTiffFromAssets(context: Context, fileName: String): String {

    val outputDir = File(context.getExternalFilesDir(null), "maps")
    if (!outputDir.exists()) outputDir.mkdirs()

    val outFile = File(outputDir, fileName)

    // Copy only once
    if (!outFile.exists()) {
        context.assets.open("maps/$fileName").use { input ->
            FileOutputStream(outFile).use { output ->
                input.copyTo(output)
            }
        }
    }

    return outFile.absolutePath
}
```

LocationDisplayHelper.kt

```
fun enableLocationDisplay(
    mapView: MapView,
    onLocationUpdate: (Location) -> Unit
) {
    val locationDisplay = mapView.locationDisplay

    // GPS-based data source (offline)
    locationDisplay.autoPanMode =
        LocationDisplay.AutoPanMode.RECENTER

    // Listen for continuous location updates
    locationDisplay.addLocationChangedListener { event ->
        event.location?.let {
            onLocationUpdate(it)
        }
    }

    // Start GPS updates (continuous)
    locationDisplay.startAsync()
}
```

GeoTiffMapScreen.kt

```
@Composable
fun GeoTiffMapScreen(tifPath: String) {

    val context = LocalContext.current

    var accuracy by remember { mutableStateOf(0.0) }
    var speed by remember { mutableStateOf(0.0) }
    var latitude by remember { mutableStateOf(0.0) }
    var longitude by remember { mutableStateOf(0.0) }

    Box(Modifier.fillMaxSize()) {

        AndroidView(
            modifier = Modifier.fillMaxSize(),
            factory = {

                val mapView = MapView(context)

                // Load GeoTIFF
                val raster = Raster(tifPath)
                val rasterLayer = RasterLayer(raster)

                val map = ArcGISMap()
                map.operationalLayers.add(rasterLayer)

                mapView.map = map

                // Zoom to raster extent
                rasterLayer.addDoneLoadingListener {
                    mapView.setViewpointGeometryAsync(
                        rasterLayer.fullExtent, 50.0
                    )
                }

                // Enable GPS location display
                enableLocationDisplay(mapView) { location ->

                    val point = location.position

                    latitude = point.y
                    longitude = point.x

                    accuracy = location.horizontalAccuracy ?: 0.0
                    speed = location.velocity ?: 0.0
```

```
            }

            mapView
        }
    )

    // 🔵 Location info overlay
    Column(
        modifier = Modifier
            .align(Alignment.TopStart)
            .padding(16.dp)
            .background(
                Color.Black.copy(alpha = 0.6f),
                RoundedCornerShape(8.dp)
            )
            .padding(12.dp)
    ) {
        Text("Lat: $latitude", color = Color.White)
        Text("Lon: $longitude", color = Color.White)
        Text("Accuracy: ${accuracy} m", color = Color.White)
        Text("Speed: ${"%.2f".format(speed)} m/s", color = Color.White)
    }
  }
}

class MainActivity : ComponentActivity() {

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

    val tifPath = copyGeoTiffFromAssets(
        this,
        "offline_map.tif"
    )

    setContent {
        MaterialTheme {
            GeoTiffMapScreen(tifPath)
        }
    }
  }

  override fun onPause() {
    super.onPause()
    ArcGISRuntimeEnvironment.pause()
  }

  override fun onResume() {
```

```
        super.onResume()
        ArcGISRuntimeEnvironment.resume()
    }
}
```