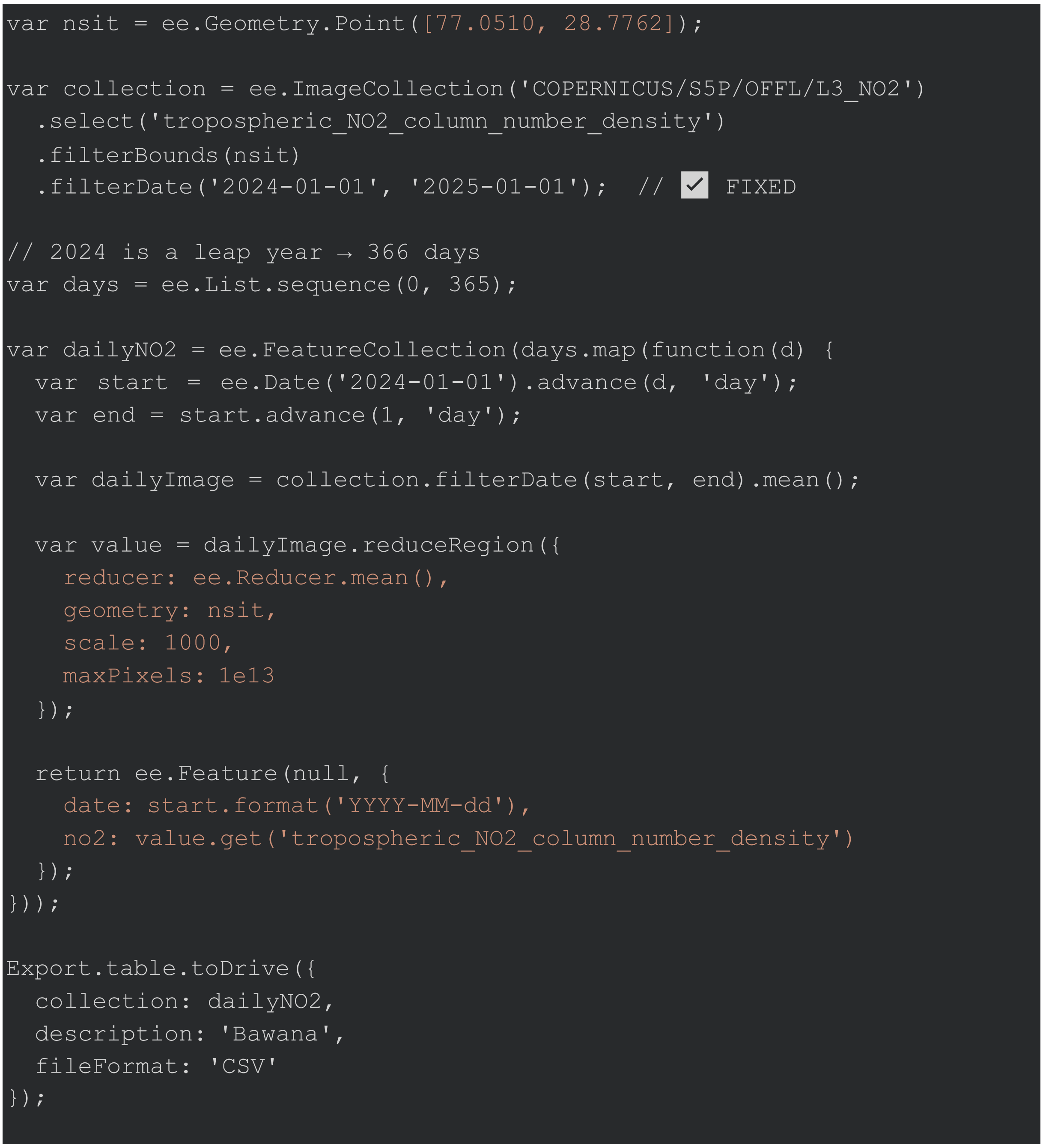
GEE SCRIPT USED

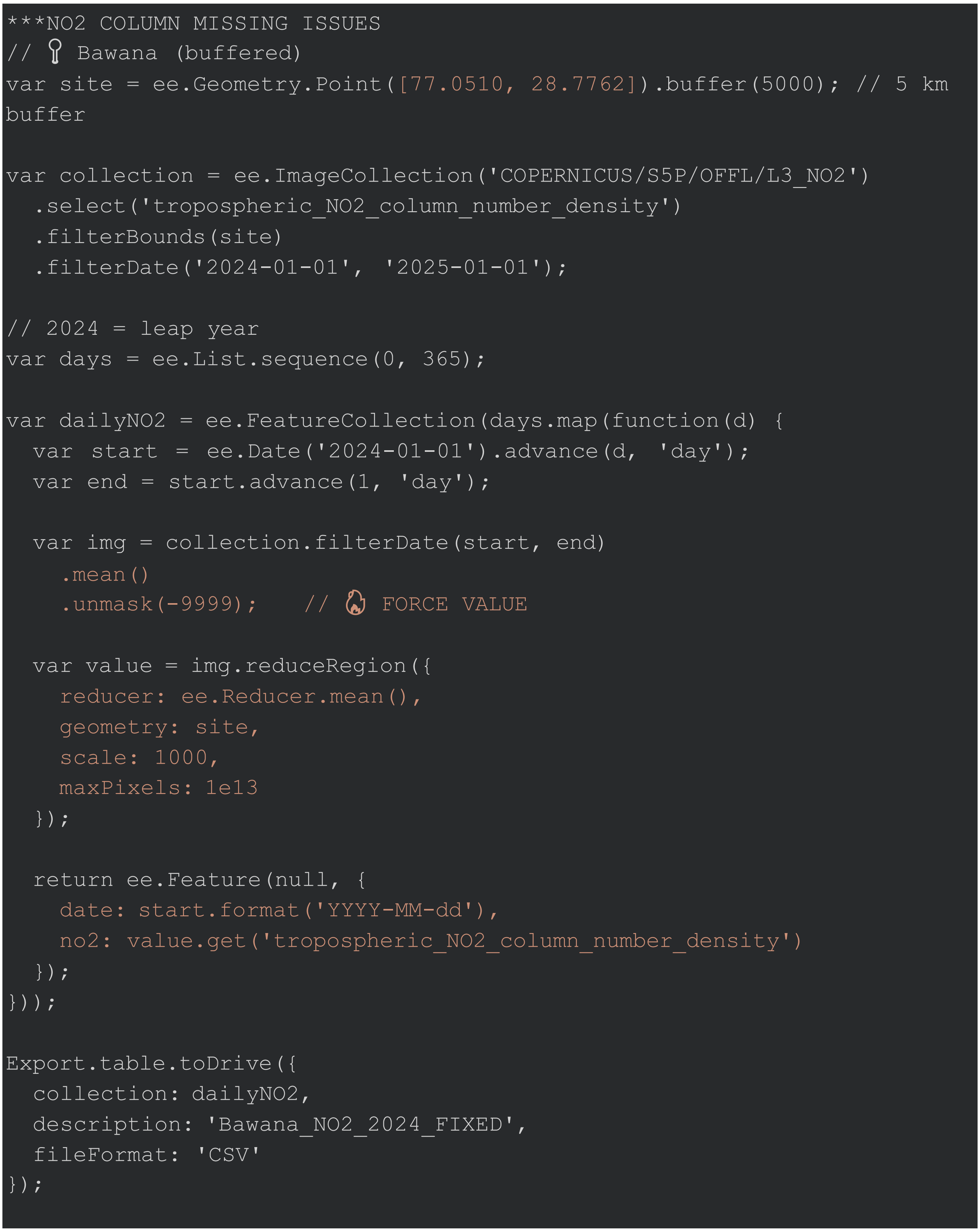
1. Satellite Tropospheric NO_2

a) In case of normal Extraction



```
var nsit = ee.Geometry.Point([77.0510, 28.7762]);

var collection = ee.ImageCollection('COPERNICUS/S5P/OFFL/L3_NO2')
  .select('tropospheric_NO2_column_number_density')
  .filterBounds(nsit)
  .filterDate('2024-01-01', '2025-01-01');  // ✅ FIXED

// 2024 is a leap year → 366 days
var days = ee.List.sequence(0, 365);

var dailyNO2 = ee.FeatureCollection(days.map(function(d) {
  var start = ee.Date('2024-01-01').advance(d, 'day');
  var end = start.advance(1, 'day');

  var dailyImage = collection.filterDate(start, end).mean();

  var value = dailyImage.reduceRegion({
    reducer: ee.Reducer.mean(),
    geometry: nsit,
    scale: 1000,
    maxPixels: 1e13
  });

  return ee.Feature(null, {
    date: start.format('YYYY-MM-dd'),
    no2: value.get('tropospheric_NO2_column_number_density')
  });
}));

Export.table.toDrive({
  collection: dailyNO2,
  description: 'Bawana',
  fileFormat: 'CSV'
});
```

b) When NO2 column in csv is missing



```
***NO2 COLUMN MISSING ISSUES
// 📍 Bawana (buffered)
var site = ee.Geometry.Point([77.0510, 28.7762]).buffer(5000); // 5 km
buffer

var collection = ee.ImageCollection('COPERNICUS/S5P/OFFL/L3_NO2')
  .select('tropospheric_NO2_column_number_density')
  .filterBounds(site)
  .filterDate('2024-01-01', '2025-01-01');

// 2024 = leap year
var days = ee.List.sequence(0, 365);

var dailyNO2 = ee.FeatureCollection(days.map(function(d) {
  var start = ee.Date('2024-01-01').advance(d, 'day');
  var end = start.advance(1, 'day');

  var img = collection.filterDate(start, end)
    .mean()
    .unmask(-9999);   // 🔥 FORCE VALUE

  var value = img.reduceRegion({
    reducer: ee.Reducer.mean(),
    geometry: site,
    scale: 1000,
    maxPixels: 1e13
  });

  return ee.Feature(null, {
    date: start.format('YYYY-MM-dd'),
    no2: value.get('tropospheric_NO2_column_number_density')
  });
}));

Export.table.toDrive({
  collection: dailyNO2,
  description: 'Bawana_NO2_2024_FIXED',
  fileFormat: 'CSV'
});
```

2)NIGHT TIME DATASET VIIRS

```
// ================================
// 1. PUSA (IARI) Location
// ================================
var pusa = ee.Geometry.Point([77.2410, 28.6286]);

// ================================
// 2. Load VIIRS Night-Time Lights
// ================================
var viirs = ee.ImageCollection(
  'NOAA/VIIRS/DNB/MONTHLY_V1/VCMCFG'
)
.select('avg_rad')
.filterBounds(pusa)
.filterDate('2024-01-01', '2025-01-01');

// ================================
// 3. Create Monthly Time Series
// ================================
var months = ee.List.sequence(1, 12);

var monthlyNTL = ee.FeatureCollection(
  months.map(function(m) {
    var start = ee.Date.fromYMD(2024, m, 1);
    var end = start.advance(1, 'month');

    var image = viirs.filterDate(start, end).mean();

    var value = image.reduceRegion({
      reducer: ee.Reducer.mean(),
      geometry: pusa,
      scale: 500,
      maxPixels: 1e13
    });

    return ee.Feature(null, {
      station: 'ITO',
      year: 2024,
      month: m,
      night_light: value.get('avg_rad')
    });
  })
);

// ==================================
// 4. Export to Google Drive
// ==================================
Export.table.toDrive({
  collection: monthlyNTL,
  description: 'ITO_VIIRS_NightLights_2024',
  fileFormat: 'CSV'
});
```

4) ERA5 LAND

```
// ==================================
// 1. Station Location
// ==================================
var station = ee.Geometry.Point([77.0719006, 28.5710274]); // NSIT Dwarka

// ==================================
// 2. ERA5-LAND DAILY DATA
// ==================================
var era5 = ee.ImageCollection('ECMWF/ERA5_LAND/DAILY_AGGR')
  .filterBounds(station)
  .filterDate('2024-01-01', '2025-01-01')
  .select([
    'u_component_of_wind_10m',
    'v_component_of_wind_10m',
    'temperature_2m',
    'surface_pressure'
  ]);

// ================================
// 3. Daily Feature Extraction
// ================================
var dailyFeatures = era5.map(function(img) {

  // Wind components
  var u10 = img.select('u_component_of_wind_10m');
  var v10 = img.select('v_component_of_wind_10m');

  // Wind speed calculation
  var windSpeed = u10.pow(2)
    .add(v10.pow(2))
    .sqrt()
    .rename('wind_speed');

  // Convert temperature from K → °C
  var tempC = img.select('temperature_2m')
    .subtract(273.15)
    .rename('temperature_2m_C');

  // Convert surface pressure from Pa → hPa
  var pressure = img.select('surface_pressure')
    .divide(100)
    .rename('surface_pressure_hPa');

  // Combine all bands
  var combined = u10
    .addBands(v10)
    .addBands(windSpeed)
    .addBands(tempC)
    .addBands(pressure);

  // Extract point values
  var values = combined.reduceRegion({
    reducer: ee.Reducer.mean(),
    geometry: station,
    scale: 10000,
    maxPixels: 1e13
  });

  return ee.Feature(null, {
    station_name: 'DWARKA-SECTOR 8',
    latitude: 28.5710274,
    longitude: 77.0719006,
    date: img.date().format('YYYY-MM-dd'),
    u10: values.get('u_component_of_wind_10m'),
    v10: values.get('v_component_of_wind_10m'),
    wind_speed: values.get('wind_speed'),
    temperature_2m_C: values.get('temperature_2m_C'),
    surface_pressure_hPa: values.get('surface_pressure_hPa')
  });
});

// ================================
// 4. Export to Google Drive
// ================================
Export.table.toDrive({
  collection: dailyFeatures,
  description: 'ERA5_LAND_Daily_Met_DWARKA-SECTOR 8_2024',
  fileFormat: 'C
```

6)ERA5 ATMOS

```
// ================================
// 1. Station Details
// ================================
var stationName = 'VIVEK VIHAR';
var lat = 28.672342;
var lon = 77.31526;

var region = ee.Geometry.Point([lon, lat]).buffer(16000);

// ================================
// 2. ERA5 ATMOSPHERIC HOURLY
// ================================
var era5 = ee.ImageCollection('ECMWF/ERA5/HOURLY')
  .filterDate('2024-01-01', '2025-01-01')
  .select([
    'boundary_layer_height',
    'total_cloud_cover'
  ]);

// ================================
// 3. Daily Aggregation
// ===============================
var days = ee.List.sequence(0, 365);

  var dailyMet = ee.FeatureCollection(days.map(function(d) {

    var date = ee.Date('2024-01-01').advance(d, 'day');
    var dailyImgs = era5.filterDate(date, date.advance(1, 'day'));

    var dailyMean = dailyImgs.mean();

    var stats = dailyMean.reduceRegion({
      reducer: ee.Reducer.mean(),
      geometry: region,
      scale: 31000,
      bestEffort: true,
      maxPixels: 1e13
    });

    return ee.Feature(null, {
      station: stationName,
      latitude: lat,
      longitude: lon,
      date: date.format('YYYY-MM-dd'),
      BLH: stats.get('boundary_layer_height'),
      total_cloud_cover: stats.get('total_cloud_cover')
    });
  }));

  // ================================
  // 4. Export CSV
  // ================================
  Export.table.toDrive({
    collection: dailyMet,
    description: 'ERA5_ATMOS_DAILY_BLH_TCC_VIVEK VIHAR_2024',
    fileFormat: 'CSV',
    selectors: [
      'station',
      'latitude',
      'longitude',
      'date',
      'BLH',
      'total_cloud_cover'
    ]
  });
```

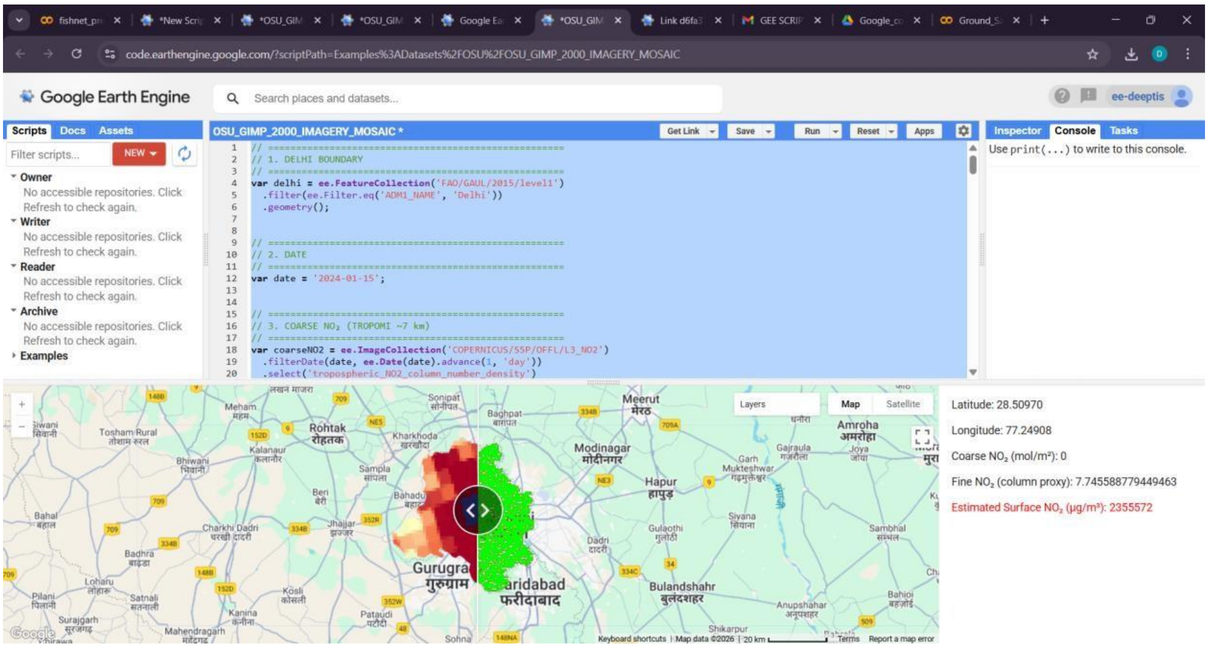
6) FINAL PREDICTION GEE AFTER LOADING GEOTIFF IN ASSET

```
// ==================================================
// 1. DELHI BOUNDARY
// ==================================================
var delhi = ee.FeatureCollection('FAO/GAUL/2015/level1')
  .filter(ee.Filter.eq('ADM1_NAME',  'Delhi'))
  .geometry();



// ==================================================
// 2. DATE
// ==================================================
var date = '2024-01-15';



// ==================================================
// 3. COARSE NO₂ (TROPOMI ~7 km)
// ==================================================
var coarseNO2 = ee.ImageCollection('COPERNICUS/S5P/OFFL/L3_NO2')
  .filterDate(date, ee.Date(date).advance(1, 'day'))
  .select('tropospheric_NO2_column_number_density')
  .mean()
  .clip(delhi)
  .unmask(0);

var coarseVis = {
  min: 0,
  max: 0.00015,
  palette: ['#313695', '#74add1', '#ffffbf', '#f46d43', '#a50026']
};



// =====================================================
// 4. FINE NO₂ (500 m ML OUTPUT – COLUMN PROXY)
// =====================================================
var fineNO2 = ee.Image('projects/ee-deeptis/assets/NO2_500m_2024-01')
  .select('b1')
  .rename('no2_column_proxy')
  .clip(delhi)
  .unmask(0);



// =====================================================
// 5. ML COVERAGE MASK (CRITICAL)
// =====================================================
var mlMask = fineNO2.gt(0);
var fineMasked = fineNO2.updateMask(mlMask);



// =====================================================
// 6. LOG SCALE (VISUAL ONLY)
// =====================================================
var fineLog = fineMasked.log();

var fineVis = {
  min: -1,
  max: 4,
  palette: ['#313695', '#74add1', '#ffffbf', '#f46d43', '#a50026']
};
```

```
// ==========================================================
// ==========================================================
// 7. ERA5 PBL HEIGHT (CORRECT DATASET)
// ==========================================================

// ERA5 HOURLY has boundary_layer_height
var pblRaw = ee.ImageCollection('ECMWF/ERA5/HOURLY')
  .filterDate(date, ee.Date(date).advance(1, 'day'))
  .select('boundary_layer_height')
  .mean()                  // daily mean PBL
  .clip(delhi);

// Mask unrealistic shallow PBL
var pbl = pblRaw
  .updateMask(mlMask)
  .updateMask(pblRaw.gte(100))  // >= 100 m
  .rename('pbl_height_m');




// ==========================================================
// 8. PBL-CORRECTED SURFACE NO₂ (ESTIMATED)
// ==========================================================
var MOLAR_MASS_NO2 = 46; // g/mol

var surfaceNO2 = fineMasked
  .divide(pbl)              // mol/m³
  .multiply(MOLAR_MASS_NO2) // g/m³
  .multiply(1e6)           // µg/m³
  .rename('surface_no2_est');



// =====================================================
// 9. COLUMN-BASED ALERT THRESHOLDS (PERCENTILES)
// =====================================================
var percentiles = fineMasked.reduceRegion({
  reducer: ee.Reducer.percentile([50, 75, 90]),
  geometry: delhi,
  scale: 500,
  maxPixels: 1e13
});

var p75 = ee.Number(percentiles.get('no2_column_proxy_p75'));
var p90 = ee.Number(percentiles.get('no2_column_proxy_p90'));

var columnHigh = fineMasked.gt(p75);
var columnSevere = fineMasked.gt(p90);



// =====================================================
// 10. SURFACE-LEVEL ALERT SYSTEM (ESTIMATED)
// =====================================================
var dangerSurface = surfaceNO2.gt(150); // µg/m³ (estimated)



// =====================================================
// 11. SPLIT MAP VIEW
// =====================================================
var leftMap = ui.Map();
var rightMap = ui.Map();

leftMap.addLayer(coarseNO2, coarseVis, 'Coarse NO₂ (~7 km)');
rightMap.addLayer(fineLog, fineVis, 'Fine NO₂ (~500 m ML, log)');
rightMap.addLayer(surfaceNO2, {
  min: 0,
  max: 200,
  palette: ['#2c7bb6', '#abd9e9', '#ffffbf', '#fdae61', '#d7191c']
}, 'Estimated Surface NO₂ (µg/m³)');
rightMap.addLayer(
dangerSurface.updateMask(dangerSurface),
  { palette: ['red'] },
  '🚨 Danger Alert (Estimated)'
);

leftMap.centerObject(delhi, 9);
rightMap.centerObject(delhi, 9);

var splitPanel = ui.SplitPanel({
  firstPanel: leftMap,
  secondPanel: rightMap,
  wipe: true
});

ui.root.widgets().reset([splitPanel]);


// ====================================================
// 12. INFO PANEL
// ====================================================
var infoPanel = ui.Panel({ style: { width: '340px', padding: '8px' } });

infoPanel.add(ui.Label({
  value: '🛰 NO₂ Inspector',
  style: { fontWeight: 'bold', fontSize: '16px' }
}));

infoPanel.add(ui.Label(
  'Column NO₂ → ML Downscaling → PBL-corrected Surface Estimate\n' + '⚠
  Surface values are estimated',
  { whiteSpace: 'pre' }
));

ui.root.widgets().add(infoPanel);



// ====================================================
// 13. CLICK INSPECTION
// ====================================================
function inspectPixel(coords) {

  infoPanel.clear();
  infoPanel.add(ui.Label('Lat: ' + coords.lat.toFixed(5)));
  infoPanel.add(ui.Label('Lon: ' + coords.lon.toFixed(5)));
  infoPanel.add(ui.Label('Loading...'));

  var point = ee.Geometry.Point([coords.lon, coords.lat]);

  var coarseSample = coarseNO2.sample(point, 1000).first();
  var fineSample   = fineNO2.sample(point, 500).first();
  var surfaceSample = surfaceNO2.sample(point, 500).first();

  ee.Dictionary({
    coarse: coarseSample,
    fine: fineSample,
    surface: surfaceSample
  }).evaluate(function(res) {

    infoPanel.clear();
    infoPanel.add(ui.Label('Latitude: ' + coords.lat.toFixed(5)));
    infoPanel.add(ui.Label('Longitude: ' + coords.lon.toFixed(5)));

    infoPanel.add(ui.Label(
      'Coarse NO₂ (mol/m²): ' +
      (res.coarse ? res.coarse.properties.tropospheric_NO2_column_number_density : 'No data')
    ));

    infoPanel.add(ui.Label(
      'Fine NO₂ (column proxy): ' +
      (res.fine ? res.fine.properties.no2_column_proxy : 'No ML prediction')
    ));

    infoPanel.add(ui.Label(
      'Estimated Surface NO₂ (µg/m³): ' +
      (res.surface ? res.surface.properties.surface_no2_est : 'No estimate'),
      { color: (res.surface && res.surface.properties.surface_no2_est > 150) ? 'red' : 'black' }
    ));
  });
}
```

```
leftMap.onClick(inspectPixel);
rightMap.onClick(inspectPixel);



// ============================================================
// 14. ML COVERAGE (VALIDATION)
// ============================================================
rightMap.addLayer(
  mlMask.updateMask(mlMask),
  { palette: ['00FF00'] },
  'ML Prediction Coverage'
);
```

PREDICTED OUTPUT



```
// ============================================================
// 1. DELHI BOUNDARY
// ============================================================
var delhi = ee.FeatureCollection('FAO/GAUL/2015/level1')
  .filter(ee.Filter.eq('ADM1_NAME', 'Delhi'))
  .geometry();


// ============================================================
// 2. DATE
// ============================================================
var date = '2024-01-15';


// ============================================================
// 3. COARSE NO2 (TROPOMI ~7 km)
// ============================================================
var coarseNO2 = ee.ImageCollection('COPERNICUS/S5P/OFFL/L3_NO2')
  .filterDate(date, ee.Date(date).advance(1, 'day'))
  .select('tropospheric_NO2_column_number_density')
```

Use print(...) to write to this console.

No accessible repositories. Click Refresh to check again.

```
Latitude: 28.50970
Longitude: 77.24908
Coarse NO2 (mol/m²): 0
Fine NO2 (column proxy): 7.745588779449463
Estimated Surface NO2 (µg/m³): 2355572
```