--Portfolio Project 02--

------------------------------Vector Geoprocessing Tools-------------------------------------

Import Basic Libraries

In [3]:

```
import geopandas as gbd
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

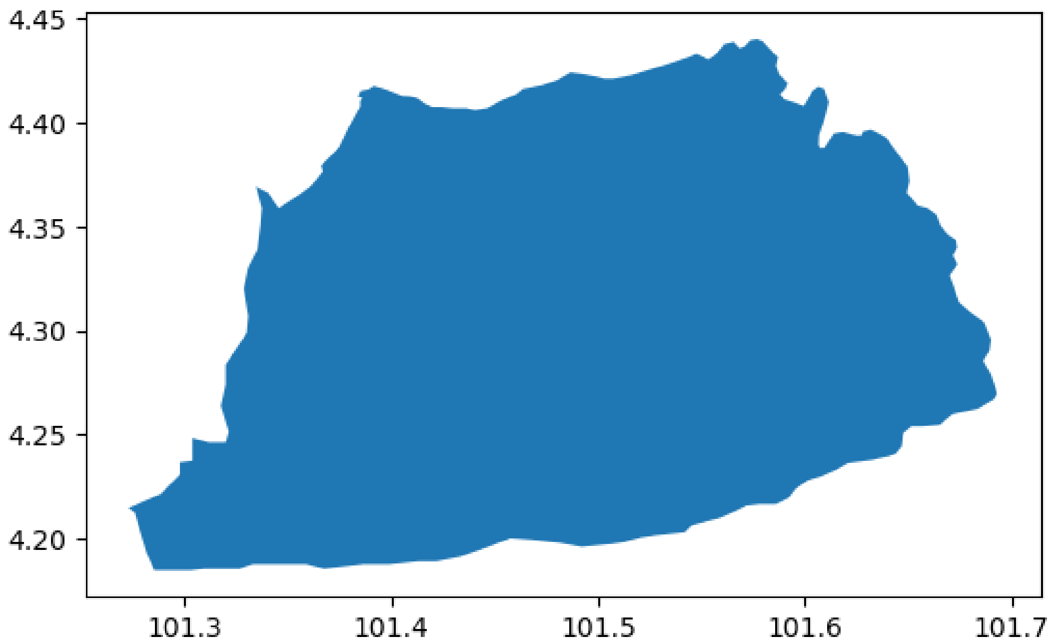
Load Shape Files (shp)

In [21]:

```
sa1 = gbd.read_file('D:/QGIS/QGIS dataset/QGIS 3d/RainFall data set (In
```

In [20]:

```
sa1.plot()
plt.show()
```

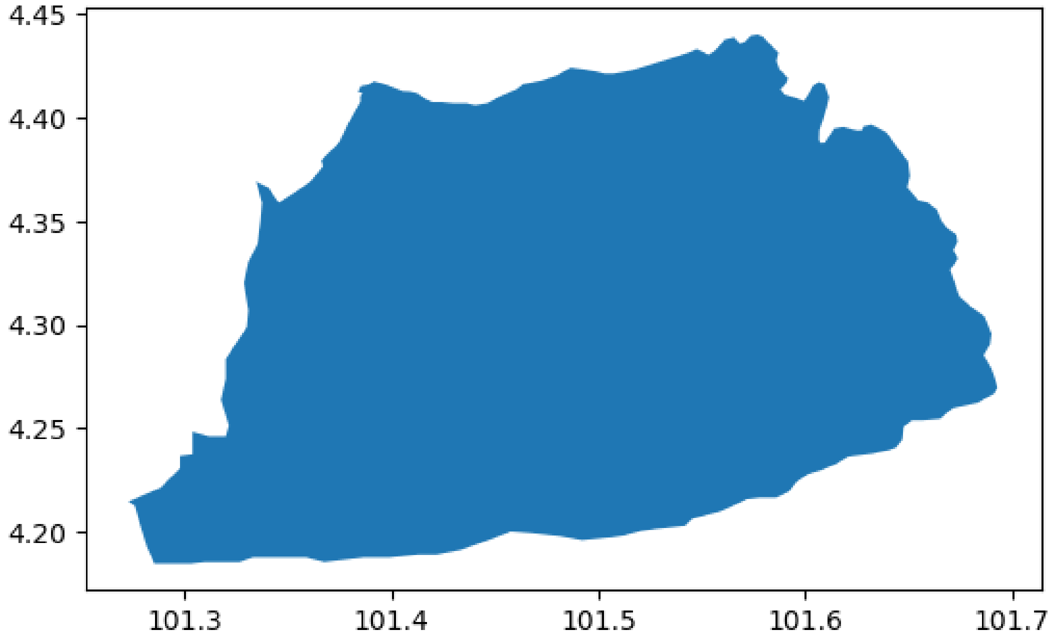


In [9]:

```
sa2= gbd.read_file('D:/QGIS/QGIS dataset/QGIS 3d/RainFall data set (Int
```

In [10]:

```
sa2.plot()
plt.show()
```

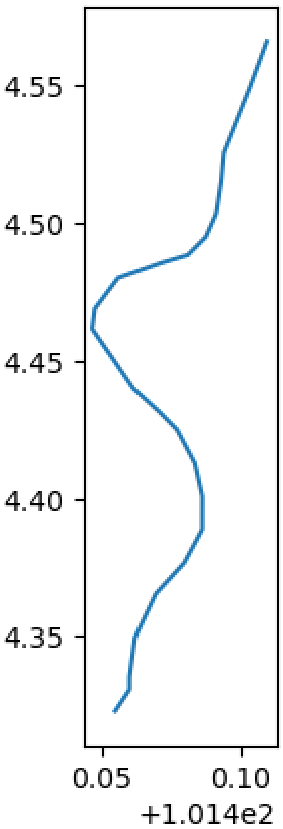


In [11]:

```
river = gbd.read_file('D:/QGIS/QGIS dataset/QGIS 3d/RainFall data set (
```

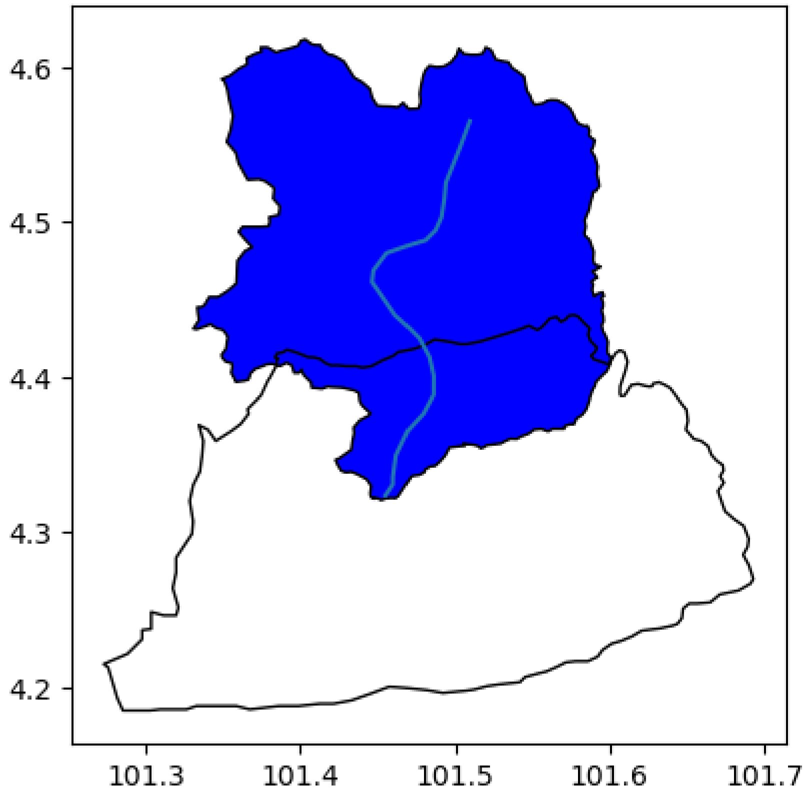
In [12]:

```
river.plot()
plt.show()
```



In [25]:
```
fig, ax = plt.subplots()
sa1.plot(ax = ax, color = 'blue', edgecolor = 'black')
sa2.plot(ax = ax, color = 'none', edgecolor = 'black')
river.plot(ax = ax)
```

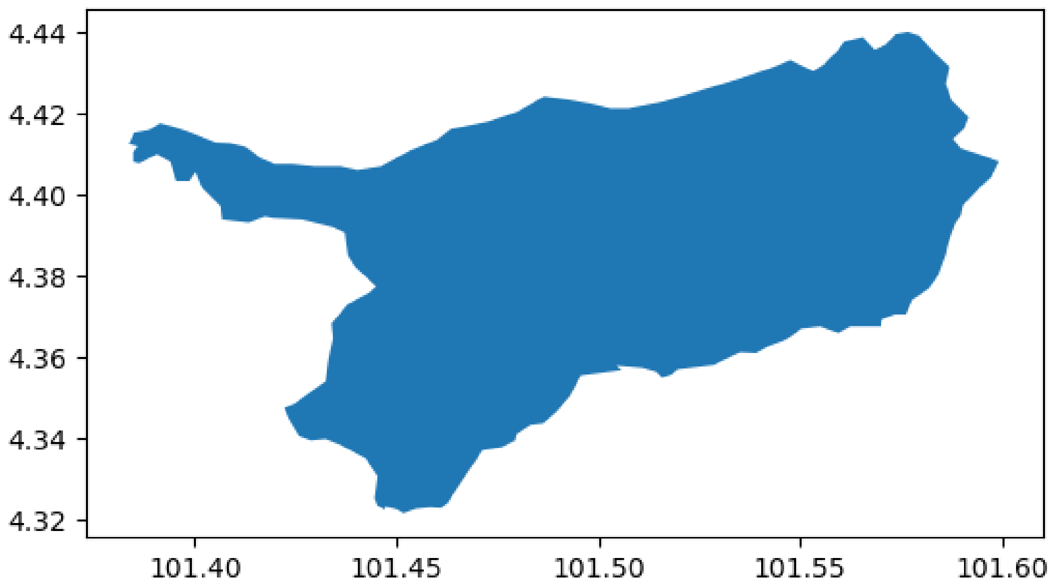
Out[25]: <Axes: >



Intersection of Polygons

In [27]:
```
intersection = gbd.overlay(sa1, sa2, how='intersection')
intersection.plot()
```

Out[27]: <Axes: >



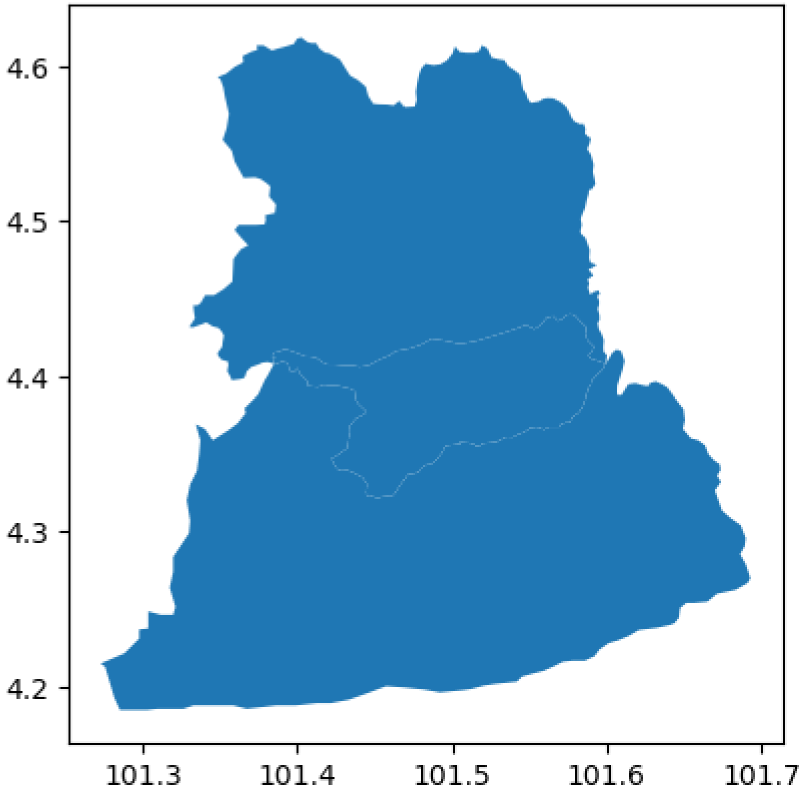
In [32]:
```
intersection.head()
```

Out[32]:

	desc_1	desc_2	geometry
0	Study_Area_1	Study_Area_2	POLYGON ((101.44456 4.32501, 101.44529 4.33050...

Union of Polygons

In [30]:
```
union = gbd.overlay(sa1, sa2, how='union')
union.plot()
plt.show()
```



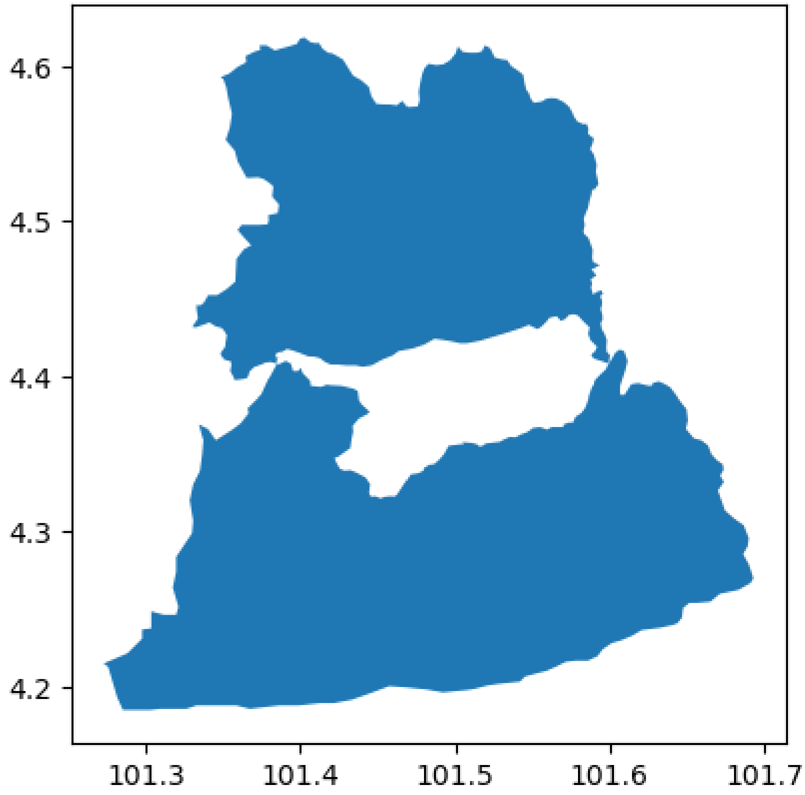
In [31]:
```
union.head()mm
```

Out[31]:

	desc_1	desc_2	geometry
0	Study_Area_1	Study_Area_2	POLYGON ((101.44456 4.32501, 101.44529 4.33050...
1	Study_Area_1	NaN	POLYGON ((101.38479 4.40801, 101.38479 4.40801...
2	NaN	Study_Area_2	POLYGON ((101.38479 4.40764, 101.38479 4.40801...

Symmetric Difference of Polygons

In [33]:
```
sy_diff = gbd.overlay(sa1, sa2, how='symmetric_difference')
sy_diff.plot()
plt.show()
```



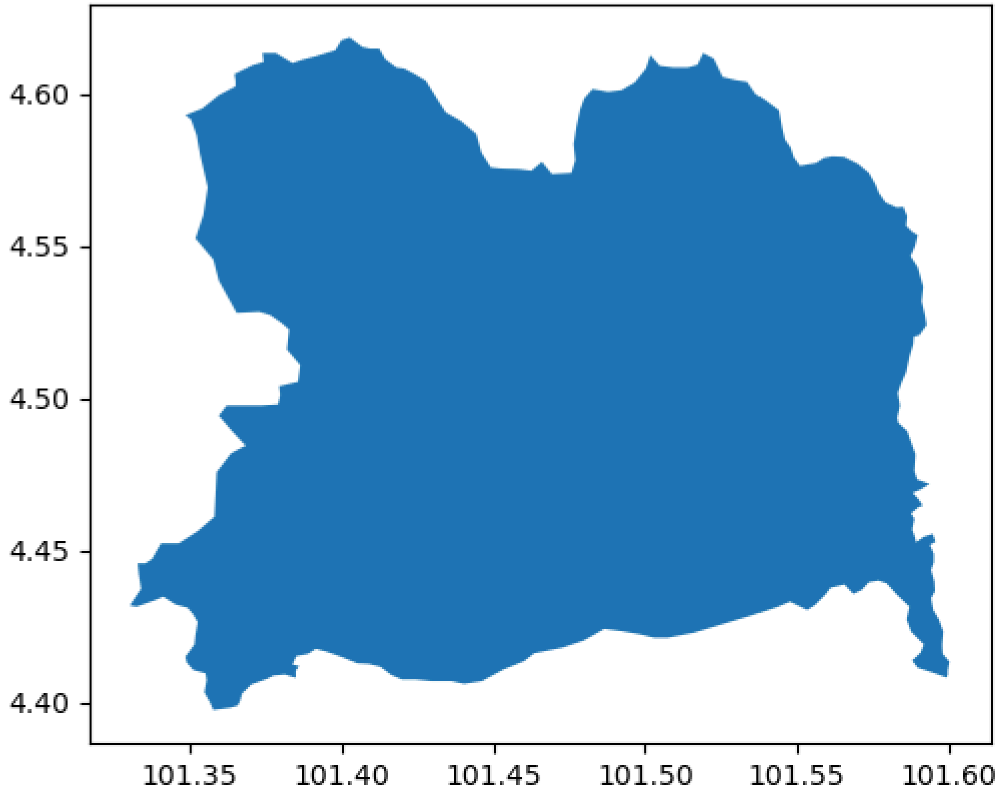
In [34]:
```
sy_diff.head()
```

Out[34]:

	desc_1	desc_2	geometry
0	Study_Area_1	NaN	POLYGON ((101.38479 4.40801, 101.38479 4.40801...
1	NaN	Study_Area_2	POLYGON ((101.38479 4.40764, 101.38479 4.40801...

Difference of Polygons

In [35]:
```
difference = gbd.overlay(sa1,sa2,how='difference')
difference.plot()
plt.show()
```



In [36]:
```
difference.head()
```

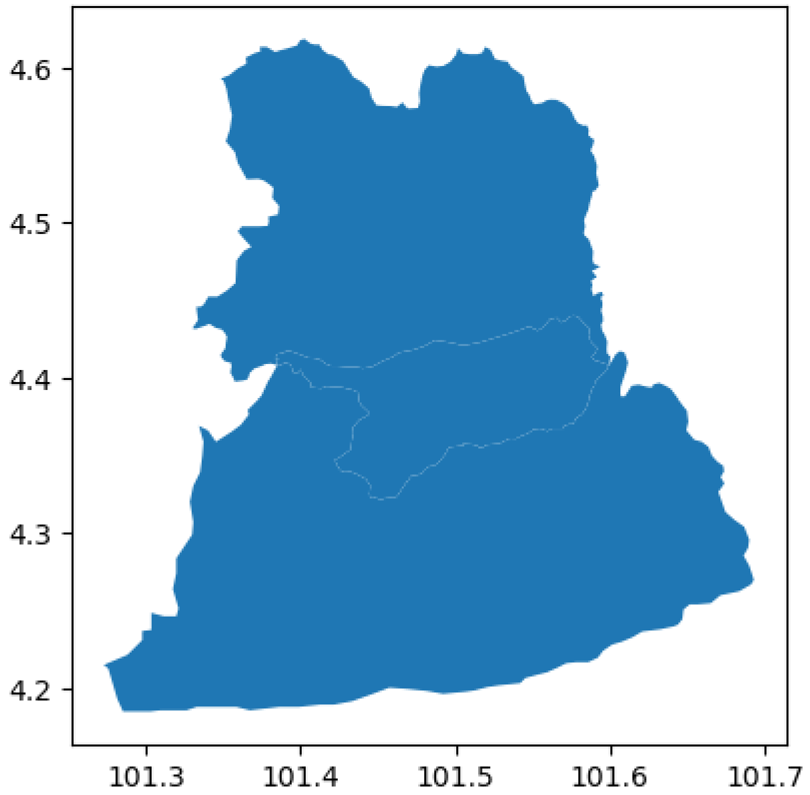
Out[36]:

	desc	geometry
0	Study_Area_1	POLYGON ((101.38479 4.40801, 101.38479 4.40801...

Dissolveing a polygon

In [38]:
```python
union = gbd.overlay(sa1, sa2, how='union')
union.plot()
```

Out[38]: <Axes: >



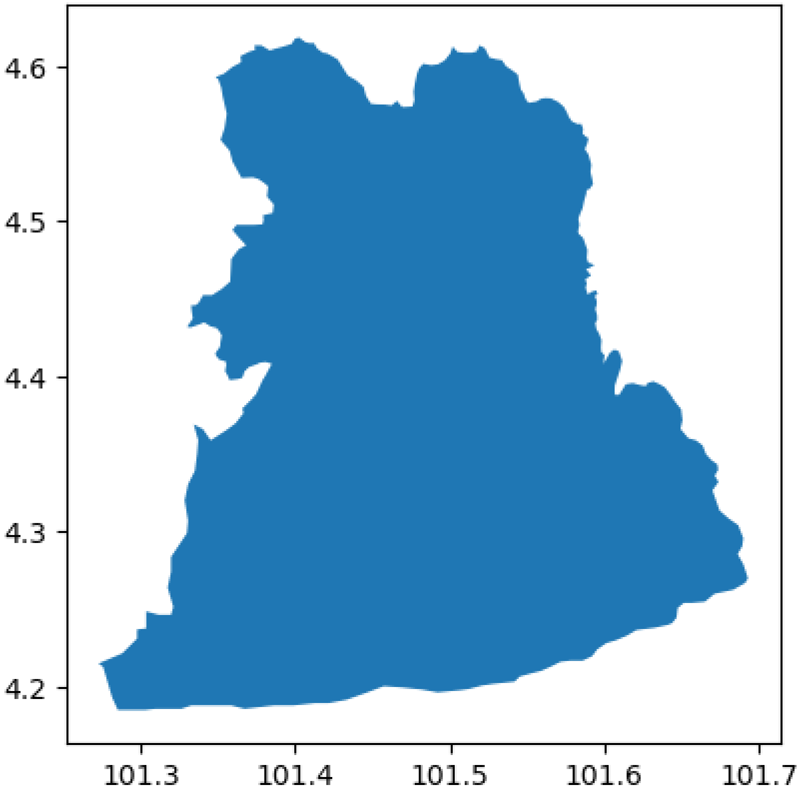
In [39]:
```python
union['common_column'] = 1
```

In [40]:
```python
union.head()
```

Out[40]:

	desc_1	desc_2	geometry	common_column
0	Study_Area_1	Study_Area_2	POLYGON ((101.44456 4.32501, 101.44529 4.33050...	1
1	Study_Area_1	NaN	POLYGON ((101.38479 4.40801, 101.38479 4.40801...	1
2	NaN	Study_Area_2	POLYGON ((101.38479 4.40764, 101.38479 4.40801...	1

In [42]:
```
dissolved_sa = union.dissolve(by = 'common_column')
dissolved_sa.plot()
plt.show()
```




```
dissolved_sa.head()
```

Buffer in Polygons

In [45]:
```
river.crs
```

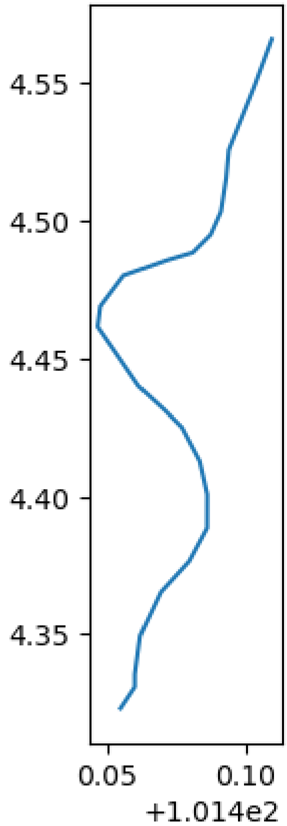
Out[45]:
```
<Geographic 2D CRS: EPSG:4326>
Name: WGS 84
Axis Info [ellipsoidal]:
- Lat[north]: Geodetic latitude (degree)
- Lon[east]: Geodetic longitude (degree)
Area of Use:
- name: World.
- bounds: (-180.0, -90.0, 180.0, 90.0)
Datum: World Geodetic System 1984 ensemble
- Ellipsoid: WGS 84
- Prime Meridian: Greenwich
```

Reprojecting the river GeoPandas GeoDataFrame into a projected CRS

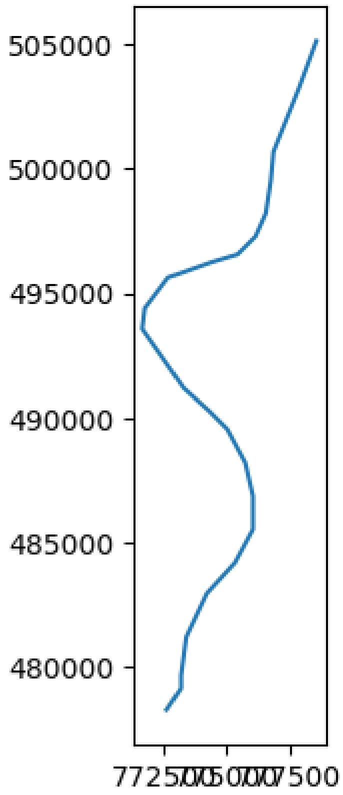
In [47]:
```
river_pro = river.to_crs(epsg=24547)
```

In [48]:
```
river.plot()
```

Out[48]: <Axes: >



In [49]:
```python
river_pro.plot()
```
Out[49]: <Axes: >



In [50]:
```python
river_pro.head()
```
Out[50]:

	id	geometry
0	1	LINESTRING (778620.752 505214.942, 777899.429 ...

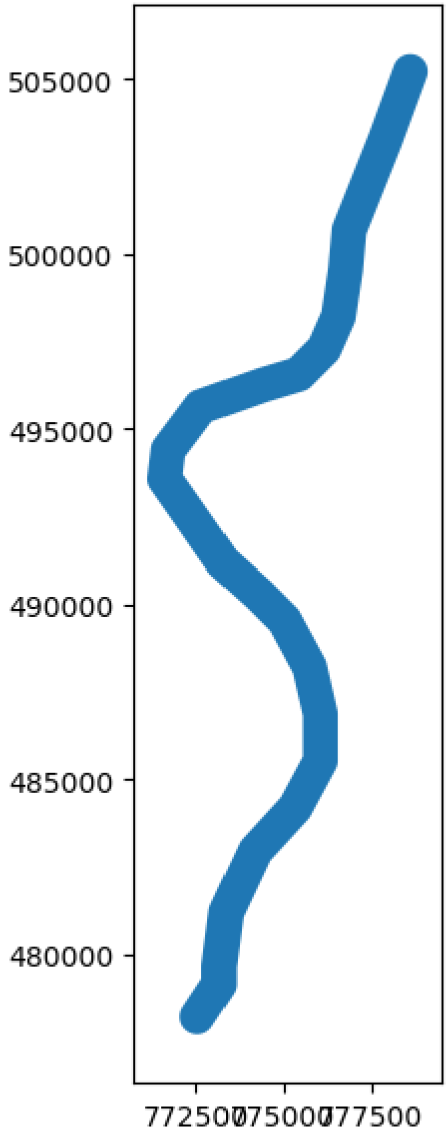
In [51]:
```python
type(river_pro)
```
Out[51]: geopandas.geodataframe.GeoDataFrame

In [53]:
```python
type(river_pro['geometry'])
```
Out[53]: geopandas.geoseries.GeoSeries

In [55]:
```python
buffer_500m = (river_pro['geometry']).buffer(distance = 500)
```

In [57]:
```
buffer_500m.plot(figsize = (7,7))
```

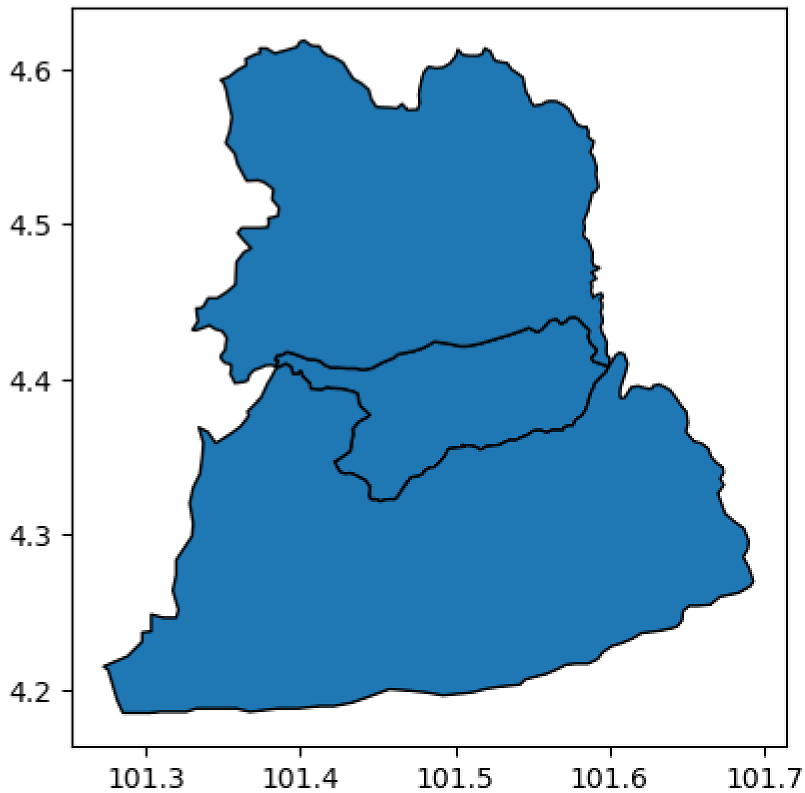
Out[57]: <Axes: >



Obtained the centroid

In [62]:

```
union = gbd.overlay(sa1,sa2,how='union')
union.plot(edgecolor = 'black')
plt.show()
```

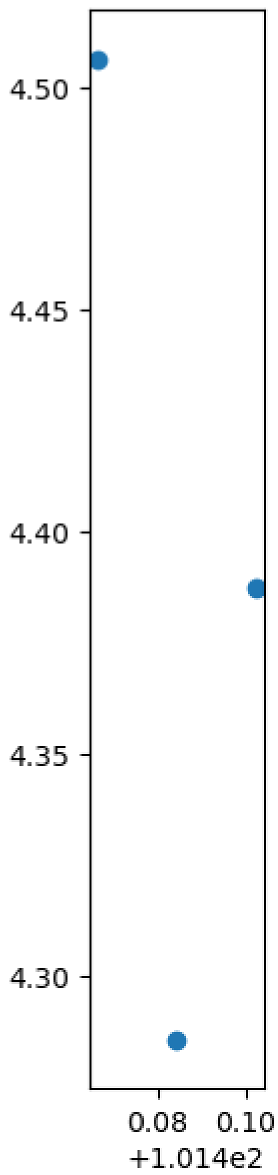


In [64]:
```python
centroid = union['geometry'].centroid
centroid.plot(figsize = (7,7))
```

```
C:\Users\DELL\AppData\Local\Temp\ipykernel_26256\3828846257.py:1: User
Warning: Geometry is in a geographic CRS. Results from 'centroid' are
likely incorrect. Use 'GeoSeries.to_crs()' to re-project geometries to
a projected CRS before this operation.

  centroid = union['geometry'].centroid
```

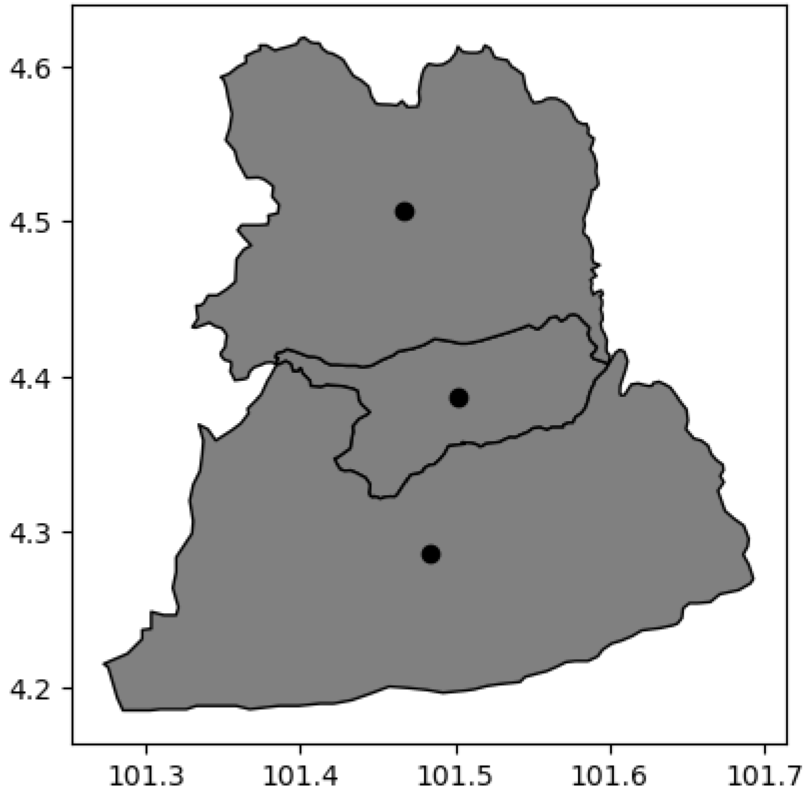
Out[64]: <Axes: >



In [66]:

```
fig1, ax1 = plt.subplots()
union.plot(ax = ax1, color = 'gray', edgecolor = 'black')
centroid.plot(ax = ax1, color = 'black')
```

Out[66]: <Axes: >



In [ ]: