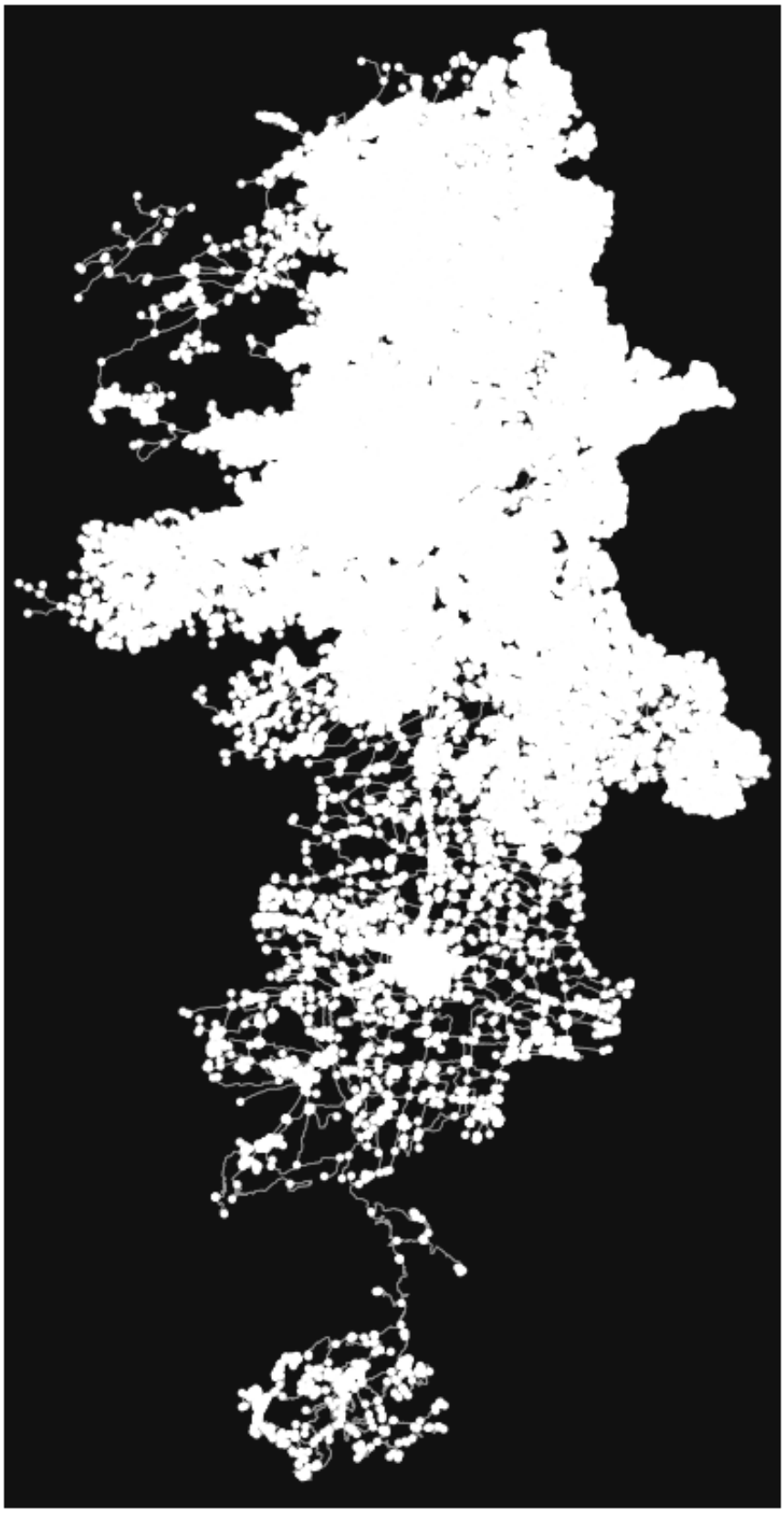
In [13]:
```
import os
import osmnx as ox
import networkx as nx
import matplotlib.pyplot as plt

cache_folder = '/Users/prakateessh/Downloads/App_building'
os.makedirs(cache_folder, exist_ok=True)

ox.settings.cache_folder = cache_folder
ox.settings.use_cache = True

def main():
    place_name = "Coimbatore, Tamil Nadu, India"
    graph = ox.graph_from_place(place_name, network_type="drive")
    fig, ax = ox.plot_graph(graph, node_size=10, edge_linewidth=0.5
    save_path = os.path.join(cache_folder, "coimbatore_graph.graphm
    ox.save_graphml(graph, filepath=save_path)

    print("Road network data for Coimbatore downloaded and saved!")

if __name__ == "__main__":
    main()
```



```
Road network data for Coimbatore downloaded and saved!
```

In [14]:
```
graph = ox.load_graphml(os.path.join(cache_folder, "coimbatore_grap
fig, ax = ox.plot_graph(graph, node_size=20, edge_linewidth=0.5, fi
```

In [20]:
```
from IPython.display import Image, display
display(Image("/Users/prakateessh/Downloads/App_building/sample.png
```

Updated User Inputs

User	Starting Point	Destination
User 1	Gandhipuram Bus Stand, Coimbatore	PSG College of Technology, Coimbatore
User 2	Peelamedu, Coimbatore	Coimbatore Junction Railway Station
User 3	RS Puram, Coimbatore	Brookefields Mall, Coimbatore
User 4	Hope College, Coimbatore	TIDEL Park, Coimbatore

In [17]:
```python
from geopy.geocoders import Nominatim

def get_location_coordinates(place):
    """Convert a place name into latitude & longitude."""
    geolocator = Nominatim(user_agent="ride_pooling")
    location = geolocator.geocode(place)
    return (location.latitude, location.longitude) if location else

num_users = int(input("Enter number of users (3-7): "))
while num_users < 3 or num_users > 7:
    num_users = int(input("Invalid! Enter a number between 3 and 7:

user_locations = []

for i in range(num_users):
    start_location = input(f"User {i+1} - Enter starting location:
    end_location = input(f"User {i+1} - Enter destination location:

    start_coords = get_location_coordinates(start_location)
    end_coords = get_location_coordinates(end_location)

    if start_coords and end_coords:
        user_locations.append({
            "user": i+1,
            "start": start_coords,
            "end": end_coords
        })
    else:
        print(f"Could not find coordinates for User {i+1}. Try agai

print("\nUser Locations:")
for user in user_locations:
    print(f"User {user['user']}: Start {user['start']} → Destinatio
```

```
User Locations:
User 1: Start (11.016481500000001, 76.96931866253846) → Destination
(11.0246833, 77.00284245647313)
User 2: Start (11.0269577, 76.9945813) → Destination (10.99756815, 7
6.96636565958987)
User 3: Start (11.0080177, 76.9501661) → Destination (11.0088739, 7
6.9593964)
User 4: Start (11.0257673, 77.0166097) → Destination (11.0321578, 7
7.02014326784214)
```

In [18]:
```python
import osmnx as ox
import networkx as nx
```

```
cache_folder = '/Users/prakateessh/Downloads/App_building'
graph = ox.load_graphml(os.path.join(cache_folder, "coimbatore_grap

for user in user_locations:
    user["start_node"] = ox.distance.nearest_nodes(graph, X=user["s
    user["end_node"] = ox.distance.nearest_nodes(graph, X=user["end

print("\nMapped Graph Nodes:")
for user in user_locations:
    print(f"User {user['user']}: Start Node {user['start_node']} →
```

```
Mapped Graph Nodes:
User 1: Start Node 803667700 → End Node 2166441938
User 2: Start Node 11371026522 → End Node 5692827144
User 3: Start Node 330044437 → End Node 12377519461
User 4: Start Node 2168236583 → End Node 9047069357
```

In [22]:
```
best_route = None
shortest_distance = float('inf')

all_routes = []

for user in user_locations:
    route = nx.shortest_path(graph, source=user["start_node"], targ
    all_routes.append(route)


    route_length = nx.path_weight(graph, route, weight="length")
    if route_length < shortest_distance:
        shortest_distance = route_length
        best_route = route

print("\nBest Route (Fuel-efficient & Shortest):")
print(best_route)
```

```
Best Route (Fuel-efficient & Shortest):
[330044437, 330042763, 330042758, 802057906, 330044447, 330041139, 5
660759948, 330041054, 11465245434, 802037911, 2966170585, 296617059
1, 11998263647, 11998263631, 11998263627, 12377519472, 1482488730, 2
96413047, 296412913, 9929068469, 12377519462, 12377519464, 917943010
4, 12377519460, 12377519461]
```

In [24]:
```
import osmnx as ox
import networkx as nx
import folium

cache_folder = '/Users/prakateessh/Downloads/App_building'
graph = ox.load_graphml(os.path.join(cache_folder, "coimbatore_grap


for user in user_locations:
    user["start_node"] = ox.distance.nearest_nodes(graph, X=user["s
    user["end_node"] = ox.distance.nearest_nodes(graph, X=user["end


all_routes = []
```

```
shortest_distance = float('inf')
best_route = None

for user in user_locations:
    route = nx.shortest_path(graph, source=user["start_node"], targ
    all_routes.append(route)


    route_length = nx.path_weight(graph, route, weight="length")
    if route_length < shortest_distance:
        shortest_distance = route_length
        best_route = route


map_center = (11.0168, 76.9558)
m = folium.Map(location=map_center, zoom_start=13)

def plot_route(route, color, weight):
    route_coords = [(graph.nodes[node]['y'], graph.nodes[node]['x']
    folium.PolyLine(route_coords, color=color, weight=weight, opaci

for route in all_routes:
    plot_route(route, color='gray', weight=3)

plot_route(best_route, color='red', weight=5)

for user in user_locations:
    folium.Marker(location=user["start"], popup=f"User {user['user'
    folium.Marker(location=user["end"], popup=f"User {user['user']}

map_path = os.path.join(cache_folder, "ride_pooling_map.html")
m.save(map_path)
print(f"Map saved! Open {map_path} to view it.")
```

```
Map saved! Open /Users/prakateessh/Downloads/App_building/ride_pooli
ng_map.html to view it.
```