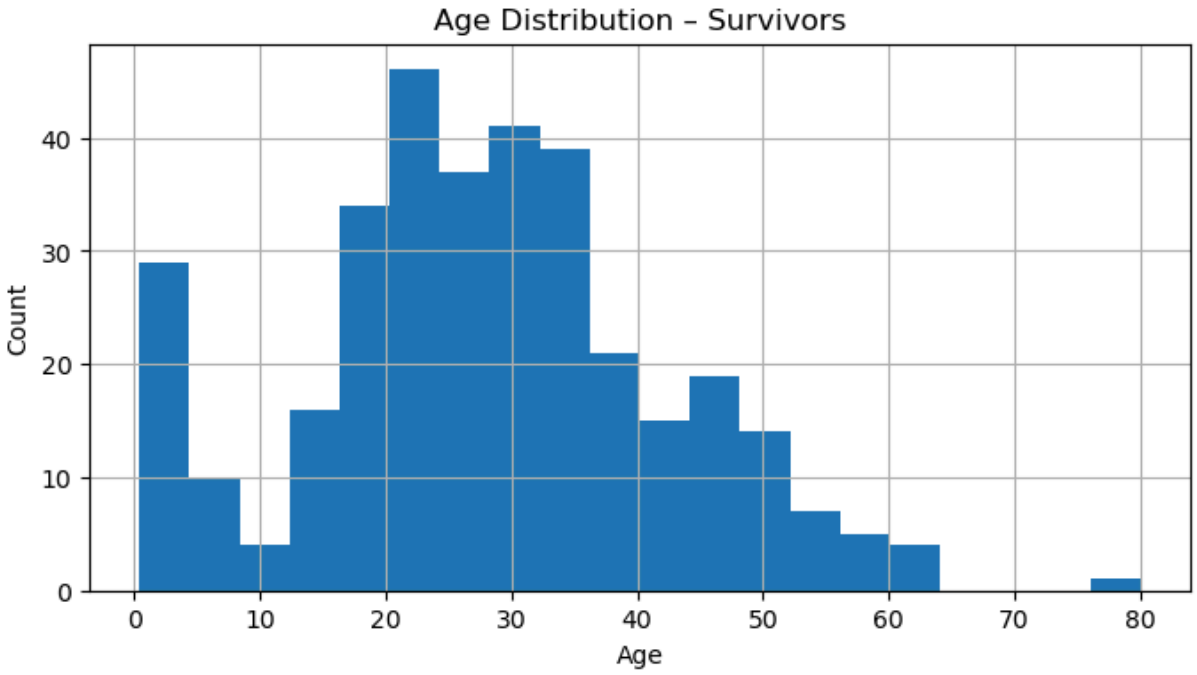
Question 1

In [13]:
```
import pandas as pd
import matplotlib.pyplot as plt

# 1. Read the Titanic data
df = pd.read_csv('titanic.csv')

# 2. Separate passengers by survival status
survived = df[df['Survived'] == 1]
drowned = df[df['Survived'] == 0]

# 2.1 Basic counts
print("Total passengers:       ", len(df))
print("  • Survived:           ", len(survived))
print("  • Drowned:            ", len(drowned))

# 2.2 Gender ratio in each group
print("\nGender ratio among survivors:")
print(survived['Sex'].value_counts(normalize=True))
print("\nGender ratio among victims:")
print(drowned['Sex'].value_counts(normalize=True))

# 2.3 Passenger class distribution
print("\nPassenger class distribution among survivors:")
print(survived['Pclass'].value_counts().sort_index())
print("\nPassenger class distribution among victims:")
print(drowned['Pclass'].value_counts().sort_index())

# 3. Histograms of age distribution
plt.figure(figsize=(8,4))
survived['Age'].dropna().hist(bins=20)
plt.title('Age Distribution – Survivors')
plt.xlabel('Age')
plt.ylabel('Count')
plt.show()

plt.figure(figsize=(8,4))
drowned['Age'].dropna().hist(bins=20)
plt.title('Age Distribution – Victims')
plt.xlabel('Age')
plt.ylabel('Count')
plt.show()

# 4. Women & children first?
wc_surv = survived[(survived['Sex'] == 'female') | (survived['Age'] < 12)]
wc_drown = drowned[(drowned['Sex'] == 'female') | (drowned['Age'] < 12)]
print(f"\nWomen/Children among survivors: {len(wc_surv)}/{len(survived)} = {len(wc_
print(f"Women/Children among victims   : {len(wc_drown)}/{len(drowned)} = {len(wc_d

# 5. First-class priority?
fc_surv = survived[survived['Pclass'] == 1]
fc_drown = drowned[drowned['Pclass'] == 1]
print(f"\nFirst-class among survivors: {len(fc_surv)}/{len(survived)} = {len(fc_sur
print(f"First-class among victims  : {len(fc_drown)}/{len(drowned)} = {len(fc_drown
```

```
Total passengers:        887
  • Survived:            342
  • Drowned:             545

Gender ratio among survivors:
Sex
female    0.681287
male      0.318713
Name: proportion, dtype: float64

Gender ratio among victims:
Sex
male      0.851376
female    0.148624
Name: proportion, dtype: float64

Passenger class distribution among survivors:
Pclass
1    136
2     87
3    119
Name: count, dtype: int64

Passenger class distribution among victims:
Pclass
1     80
2     97
3    368
Name: count, dtype: int64
```



Question 1.

Part 2.

Total records**: 887 passengers**

- Survived: 342
- Drowned: 545

Gender ratios:

- Among survivors, ~68.1% were female, and ~31.9% male.
- Among victims, ~85.1% were male, ~14.9% female.

Passenger class breakdown:

- Survivors: 136 in class 1, 109 in class 2, 97 in class 3.
- Victims: 80 in class 1, 97 in class 2, 368 in class 3.

Part 3

Histograms illustrate that victims skew older and more concentrated in adult ages, while survivors include many younger passengers.

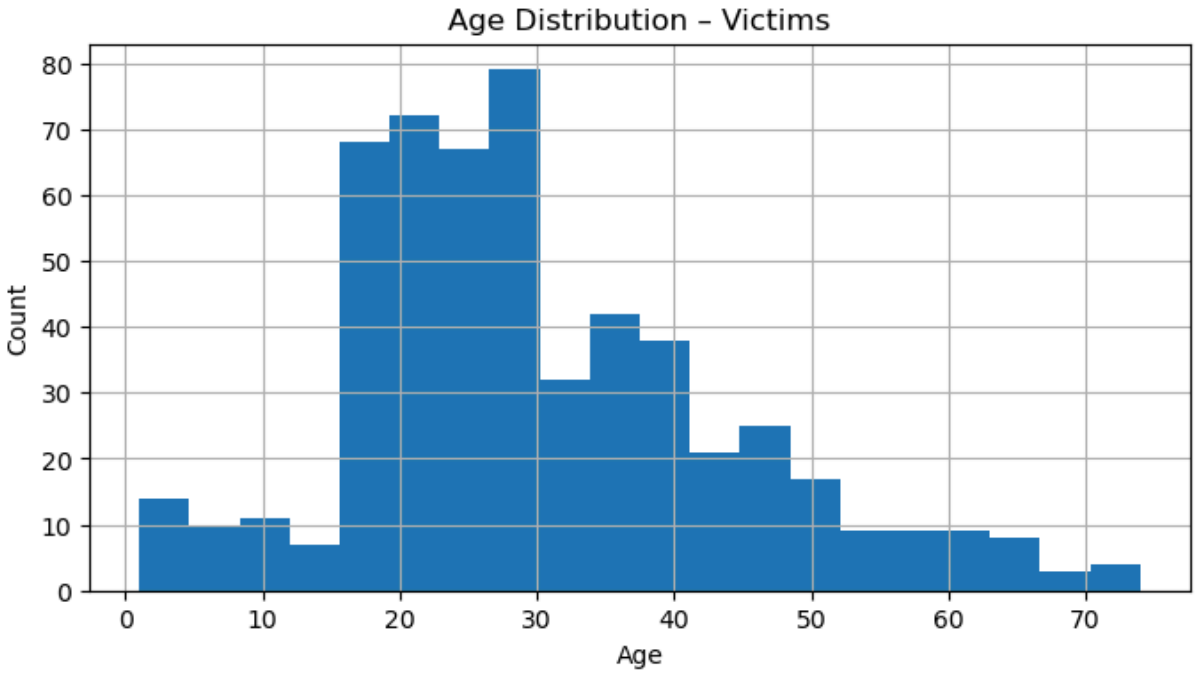
Part 4: Were women and children prioritized?

Yes. The data show that among the 342 survivors, 255 (74.6%) were either women or children under 12, whereas among the 545 who drowned only 99 (18.2%) falls into that same category. That large disparity strongly supports Cameron's "women and children first" narrative.

Part 5: First-class priority?

39.8% of survivors were first-class passengers, compared with 14.7% of victims—clear evidence that first-class passengers had boarding priority.

The conclusion is that poor men are the ones that always take it in the chin, no matter how history evolves. I ran my own code separately to the assignment requirements, and I found that men ages 60+, who were in class 3, were the ones that survived the least (<11%), or shortly the ones left behind, simply because of their gender and wealth status.

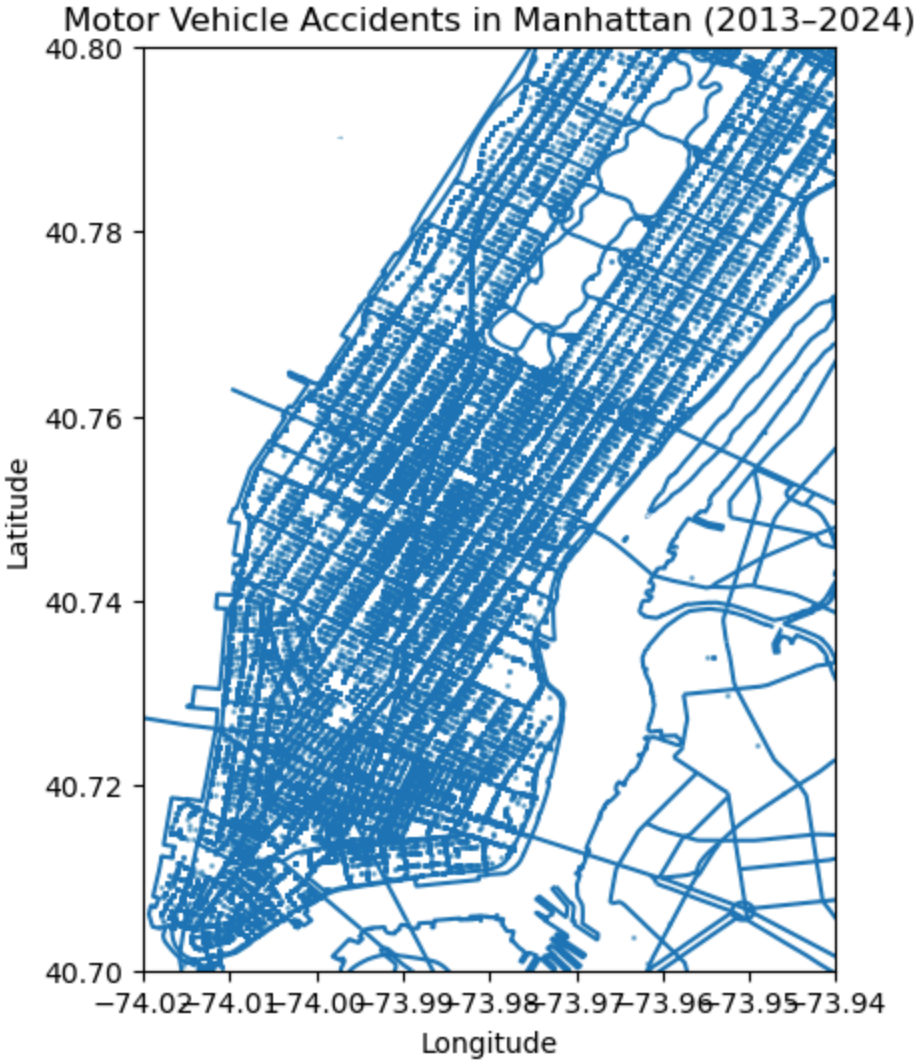


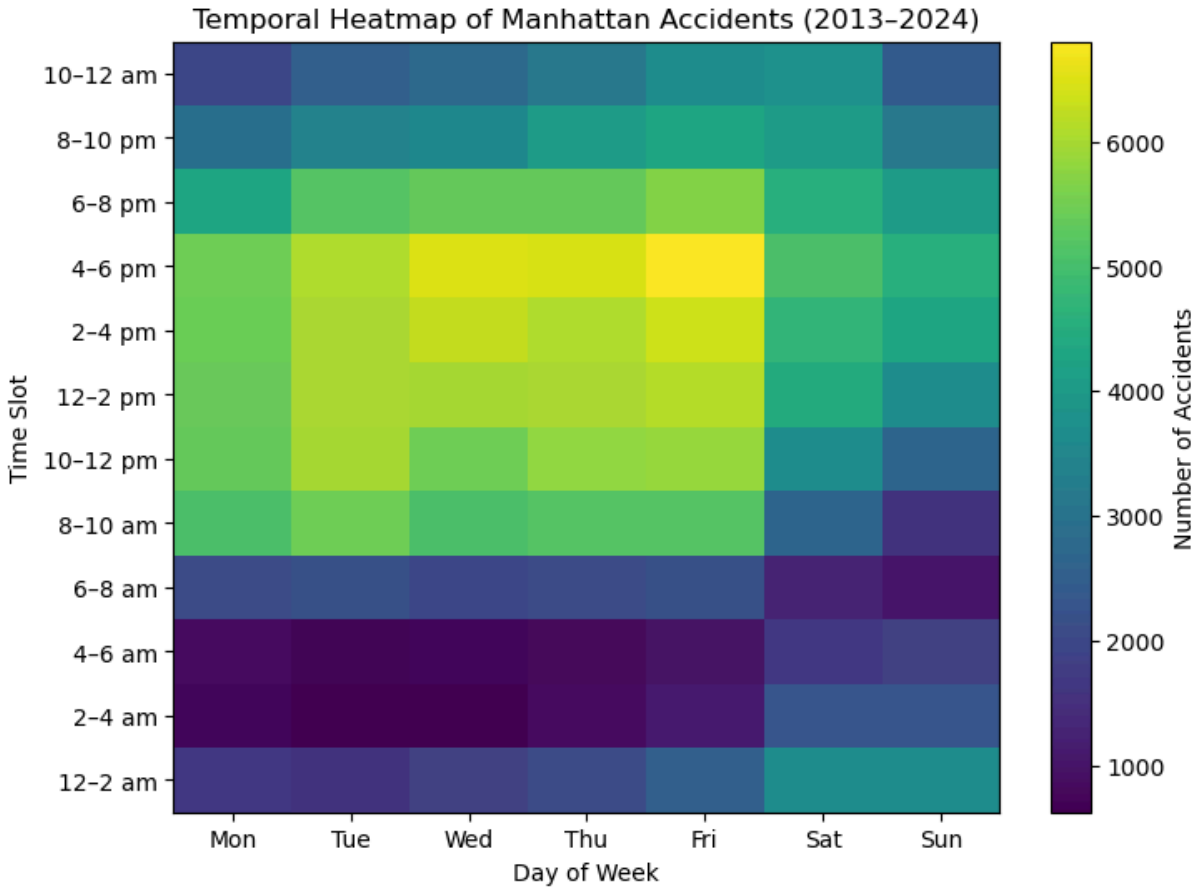

```
Women/Children among survivors: 255/342 = 74.6%
Women/Children among victims   : 99/545 = 18.2%

First-class among survivors: 136/342 = 39.8%
First-class among victims  : 80/545 = 14.7%
```

Question 2 . Part 1

In [25]:
```python
# Part 1: Spatial & Temporal Views for Manhattan Accidents

import pandas as pd
import geopandas as gpd
from shapely import wkt
import matplotlib.pyplot as plt
import numpy as np

# Load & filter collisions
acc = pd.read_csv(
    'Motor-Vehicle-Collisions.csv',
    parse_dates=['CRASH DATE'],
    dtype={'ZIP CODE': str},
    low_memory=False
)

acc = acc[(acc['BOROUGH']=='MANHATTAN') & acc['LATITUDE'].notna() & acc['LONGITUDE'

# --- Spatial View (Figure 3) ---
gdf_acc = gpd.GeoDataFrame(
    acc,
    geometry=gpd.points_from_xy(acc.LONGITUDE, acc.LATITUDE),
    crs='EPSG:4326'
)
streets   = pd.read_csv('dcm-nyc-major-streets.csv')
shoreline = pd.read_csv('nyc-shoreline.csv')
streets['geometry']   = streets['the_geom'].apply(wkt.loads)
shoreline['geometry'] = shoreline['the_geom'].apply(wkt.loads)
gdf_str = gpd.GeoDataFrame(streets,   geometry='geometry', crs='EPSG:4326')
```

```
gdf_sho = gpd.GeoDataFrame(shoreline, geometry='geometry', crs='EPSG:4326')

fig, ax = plt.subplots(figsize=(8,6))
gdf_sho.plot(ax=ax)
gdf_str.plot(ax=ax)
gdf_acc.plot(ax=ax, markersize=1, alpha=0.5)
ax.set_xlim(-74.02, -73.94)
ax.set_ylim(40.70, 40.80)
ax.set_title('Motor Vehicle Accidents in Manhattan (2013–2024)')
ax.set_xlabel('Longitude')
ax.set_ylabel('Latitude')
plt.show()

# --- Temporal Heatmap (Figure 4) ---
acc['Hour'] = acc['CRASH TIME'].str.split(':').str[0].astype(int)
acc['Slot'] = (acc['Hour'] // 2).clip(upper=11)
acc['Day']  = acc['CRASH DATE'].dt.weekday

heat = np.zeros((12,7), dtype=int)
for _, row in acc.iterrows():
    heat[row['Slot'], row['Day']] += 1

fig, ax = plt.subplots(figsize=(8,6))
im = ax.imshow(heat, aspect='auto', origin='lower')
ax.set_xticks(range(7))
ax.set_xticklabels(['Mon','Tue','Wed','Thu','Fri','Sat','Sun'])
ax.set_yticks(range(12))
ax.set_yticklabels([
    "12–2 am","2–4 am","4–6 am","6–8 am","8–10 am","10–12 pm",
    "12–2 pm","2–4 pm","4–6 pm","6–8 pm","8–10 pm","10–12 am"
])
ax.set_xlabel('Day of Week')
ax.set_ylabel('Time Slot')
ax.set_title('Temporal Heatmap of Manhattan Accidents (2013–2024)')
fig.colorbar(im, ax=ax, label='Number of Accidents')
plt.show()
```


Motor Vehicle Accidents in Manhattan (2013–2024)
40.80
40.78
40.76
40.74
40.72
40.70
Latitude
−74.02
−74.01
−74.00
−73.99
−73.98
−73.97
−73.96
−73.95
−73.94
Longitude



Q2: Part 2

In [27]:

```
import pandas as pd

# Load & filter Manhattan collisions with valid coordinates
acc = pd.read_csv(
    'Motor-Vehicle-Collisions.csv',
    parse_dates=['CRASH DATE'],
    low_memory=False
)
acc = acc[(acc['BOROUGH'] == 'MANHATTAN') & acc['LATITUDE'].notna() & acc['LONGITUD

# Part 2: Compute injury/fatality statistics
total_accidents    = len(acc)
persons_injured    = acc['NUMBER OF PERSONS INJURED'].sum()
cyclists_injured   = acc['NUMBER OF CYCLIST INJURED'].sum()
pedestrians_injured= acc['NUMBER OF PEDESTRIANS INJURED'].sum()
motorists_injured  = acc['NUMBER OF MOTORIST INJURED'].sum()

persons_killed     = acc['NUMBER OF PERSONS KILLED'].sum()
cyclists_killed    = acc['NUMBER OF CYCLIST KILLED'].sum()
pedestrians_killed = acc['NUMBER OF PEDESTRIANS KILLED'].sum()
motorists_killed   = acc['NUMBER OF MOTORIST KILLED'].sum()

# Print results
print(f"--- Total No Accidents      = {total_accidents:,}")
print(f"--- No Persons Injured      = {persons_injured:,}")
print(f"--- No Cyclists Injured     = {cyclists_injured:,}")
print(f"--- No Pedestrians Injured  = {pedestrians_injured:,}")
print(f"--- No Motorists Injured    = {motorists_injured:,}\n")
```

```
print(f"--- No Persons Killed       = {persons_killed:,}")
print(f"--- No Cyclists Killed      = {cyclists_killed:,}")
print(f"--- No Pedestrians Killed   = {pedestrians_killed:,}")
print(f"--- No Motorists Killed     = {motorists_killed:,}")
```

```
--- Total No Accidents      = 308,929
--- No Persons Injured      = 64,710.0
--- No Cyclists Injured     = 12,252
--- No Pedestrians Injured  = 20,599
--- No Motorists Injured    = 31,008

--- No Persons Killed       = 313.0
--- No Cyclists Killed      = 33
--- No Pedestrians Killed   = 224
--- No Motorists Killed     = 52
```

Q2: Part 3

In [29]:

```
import pandas as pd

# Chunked load to avoid memory spikes
vehicle_cols = [
    'VEHICLE TYPE CODE 1',
    'VEHICLE TYPE CODE 2',
    'VEHICLE TYPE CODE 3',
    'VEHICLE TYPE CODE 4',
    'VEHICLE TYPE CODE 5'
]
usecols = ['BOROUGH','LATITUDE','LONGITUDE'] + vehicle_cols

veh_counts = {}

for chunk in pd.read_csv(
    'Motor-Vehicle-Collisions.csv',
    usecols=usecols,
    chunksize=200_000,
    low_memory=False
):
    # Filter only Manhattan accidents with valid coords
    m = chunk[
        (chunk['BOROUGH']=='MANHATTAN') &
        chunk['LATITUDE'].notna() &
        chunk['LONGITUDE'].notna()
    ]
    # Tally each vehicle-type column
    for col in vehicle_cols:
        vc = m[col].value_counts(dropna=True)
        for vtype, cnt in vc.items():
            veh_counts[vtype] = veh_counts.get(vtype, 0) + cnt

# Print sorted dictionary
print("Vehicle Type Accident Count")
print("="*40)
for vtype, cnt in sorted(veh_counts.items(), key=lambda x: x[1], reverse=True):
    print(f"{vtype:30} {cnt:,}")
print("="*40)
```

```
Vehicle Type Accident Count
=====================================
PASSENGER VEHICLE              110,389
Sedan                          100,972
Station Wagon/Sport Utility Vehicle 74,825
SPORT UTILITY / STATION WAGON  52,151
TAXI                           42,627
Taxi                           35,000
VAN                            16,203
UNKNOWN                        15,390
Box Truck                      12,566
OTHER                          11,155
Bike                           10,876
SMALL COM VEH(4 TIRES)         9,543
BUS                            9,211
Pick-up Truck                  8,610
LIVERY VEHICLE                 8,076
Bus                            7,733
LARGE COM VEH(6 OR MORE TIRES) 7,218
4 dr sedan                     6,928
PICK-UP TRUCK                  6,728
BICYCLE                        6,069
Van                            4,460
MOTORCYCLE                     2,048
Motorcycle                     1,889
Tractor Truck Diesel           1,455
E-Bike                         1,234
Dump                           1,152
E-Scooter                      1,088
Ambulance                      1,060
AMBULANCE                      977
Garbage or Refuse              962
PK                             933
Convertible                    817
Flat Bed                       780
Moped                          655
Carry All                      567
2 dr sedan                     399
FIRE TRUCK                     394
Tow Truck / Wrecker            349
Tractor Truck Gasoline         299
Motorscooter                   243
Refrigerated Van               216
van                            192
Armored Truck                  182
Tanker                         177
Chassis Cab                    168
SCOOTER                        168
Concrete Mixer                 138
Flat Rack                      133
Pedicab                        128
Beverage Truck                 125
Motorbike                      114
PEDICAB                        106
AMBUL                          106
LIMO                           104
```

3-Door	94
TRUCK	84
School Bus	69
UNK	68
MOPED	63
Stake or Rack	56
Lift Boom	48
FIRE	46
FDNY	45
E-Bik	45
USPS	41
UNKNO	41
Multi-Wheeled Vehicle	38
COM	38
ambul	32
Bulk Agriculture	31
DUMP	30
DELIV	30
TRAIL	30
DELV	29
COMME	27
E-Sco	26
Lunch Wagon	25
TRK	23
ambulance	21
Unknown	21
Pickup with mounted Camper	20
Ambul	20
unkno	20
Open Body	19
AMBU	17
TRACT	17
Fire	17
UTIL	16
unk	16
Scooter	15
BOX T	15
REFG	14
Van Camper	14
bus	14
TRAILER	13
Minibike	13
Pallet	13
truck	13
Snow Plow	12
Trailer	12
Motorized Home	12
moped	12
trail	12
FIRET	12
FOOD	12
fire	12
Truck	11
dump	11
RV	10
BOX TRUCK	10

```
FDNY FIRE                          9
AMB                                9
Fire Truck                         9
unknown                            9
Unk                                9
Fire truck                         9
Glass Rack                         9
UTILI                              9
USPS TRUCK                         8
Tow Truck                          8
TOW T                              8
SANIT                              8
deliv                              8
POSTA                              8
COMMERCIAL                         7
FIRETRUCK                          7
FDNY TRUCK                         7
SCOOT                              7
US PO                              7
MOTOR                              7
firet                              7
ELECT                              7
comme                              7
WAGON                              7
BACKH                              7
ambu                               7
box t                              7
Delivery                           6
STAK                               6
PAS                                6
FDNY AMBUL                         6
GARBAGE TR                         6
Enclosed Body - Nonremovable Enclosure 6
FLAT                               6
SCHOO                              6
FORKL                              6
PASS                               6
CRANE                              6
TRL                                6
schoo                              6
FORD                               6
FREIG                              6
posta                              6
OMT                                6
pick                               6
TOW TRUCK                          5
Firetruck                          5
fire truck                         5
Vanette                            5
TANK                               5
SANITATION                         5
SPRINTER V                         5
PICKU                              5
SUV                                5
TRAC                               5
MACK                               5
```

STREE	5
usps	5
UHAUL	5
sanit	5
horse	5
SELF	5
GARBA	5
Trail	5
DELIVERY T	5
BOOM LIFT	4
box truck	4
MTA BUS	4
Fire Engin	4
com	4
TRACTOR	4
HORSE	4
MTA B	4
tow t	4
UNKOW	4
US Po	4
tow	4
taxi	4
tract	4
STREET SWE	4
FDNY Truck	3
box	3
FOOD CART	3
Commercial	3
scooter	3
PICK UP	3
Self	3
CATER	3
POWER SHOV	3
USPS Truck	3
UKNOWN	3
skateboard	3
Forklift	3
FDNY fire	3
SEDAN	3
Utility	3
CART	3
REVEL MOPE	3
FDNY Fire	3
ELECTRIC M	3
G COM	3
motor	3
ICE C	3
BS	3
Forkl	3
Delv	3
Pick up tr	3
FLATB	3
TOWER	3
Hopper	3
LIGHT	3
SPRIN	3

Garba	3
Sanit	3
MAIL	3
LIMOU	3
PEDIC	3
Deliv	3
GLBEN	3
POWER	3
fdny	3
utili	3
tk	3
Subn	3
GOLF	3
HORSE CARR	3
garba	3
util	3
2 DOO	3
Livestock Rack	3
Flat	3
MINIV	3
self	3
Stree	3
Tow T	3
VAN T	3
Box truck	2
Motorized	2
MINIVAN	2
E Bike w p	2
US POSTAL	2
ELECTRIC S	2
EMS	2
STREET CLE	2
Sprinter V	2
forklift	2
commerical	2
MOTOR SCOO	2
FDNY ENGIN	2
trailer	2
BOX	2
FIRE ENGIN	2
USPS Mail	2
PICK UP TR	2
FDNY Ambul	2
boom lift	2
Usps truck	2
CAT	2
UNICYCLE	2
us postal	2
Excavator	2
USPS MAIL	2
VAN TRUCK	2
GAS SCOOTE	2
SKATEBOARD	2
BOX VAN	2
0	2
Delivery T	2

FDNY Engin	2
FDNY FIRET	2
constructi	2
delivery t	2
POSTAL VEH	2
Wagon	2
Scoot	2
Unkno	2
SEMI	2
City	2
Horse	2
DELIVERY	2
Enclosed Body - Removable Enclosure	2
Pickup	2
Food Cart	2
MOBIL	2
Lift	2
Mobile foo	2
P/SH	2
Sanitation	2
Flatbed	2
4dsd	2
TRANSPORT	2
fork	2
FORK	2
Utili	2
Rescu	2
GRAY	2
rv	2
elect	2
SUBUR	2
GOVER	2
NYPD	2
COMER	2
us po	2
dsny	2
Boom	2
track	2
food	2
delv	2
Crane	2
Tow t	2
YELLO	2
Marke	2
jeep	2
CONST	2
Cargo	2
ENGIN	2
EBIKE	2
COM T	2
ARMOR	2
PSD	2
U-HAU	2
p/sh	2
U HAU	2
COMM	2

FLEET	2
power	2
cemen	2
Box T	2
REFRI	2
wagon	2
ford	2
HINO	2
OML	2
INTL	2
NOT I	2
INTER	2
SCOM	2
CMIX	2
subn	2
PARKE	2
EXCAV	2
MOPD	2
Tractor	2
Skateboard	2
FD AMBULAN	2
SUBURBAN	2
GOLF CART	2
FORK LIFT	2
E-Unicycle	2
NJ transit	1
Government	1
SKYWATCH	1
City vehic	1
FDNY LADER	1
HERTZ RAM	1
SPR	1
Self Insur	1
na	1
Fdny engin	1
UPS TRUCK	1
FDNY Ladde	1
CMIXER	1
PROMASTER	1
YELLOW TAX	1
Motorscoot	1
F550 ESU R	1
FORD SPRIN	1
GOVERNMENT	1
post offic	1
NYC AMBULA	1
FRIEG	1
BOBCAT	1
PICKUP	1
DODGE RAM	1
Pro master	1
Skywatch	1
Ambu	1
Cement tru	1
kick scoot	1
Dump truck	1

UHAL	1
commercial	1
AMBULACE	1
USPS truck	1
USPCS	1
SCOOTER GA	1
Pick up	1
DUMP TRUCK	1
Pass	1
Horse carr	1
Garbage	1
SEDONA	1
NonMotordS	1
VERZION VA	1
MOVING TRU	1
Post offic	1
NYS AMBULA	1
work truck	1
crane boom	1
City Owned	1
AMBULANE	1
gas scoote	1
USPS POSTA	1
Go kart	1
Power shov	1
sedan	1
FED EX	1
PEDI CAB	1
Garbage Tr	1
TOWTRUCK	1
DEL	1
DIRT BIKE	1
WORKH UTIL	1
Razor Scoo	1
POSTAL SER	1
Command Po	1
T350HD	1
REVEL SCOO	1
FOR VAN	1
CADET	1
AMBULETTE	1
U haul mov	1
Tk	1
CAT CATERP	1
FDNY Vehic	1
Push scoot	1
FLAT BED T	1
SUBN TN	1
e scooter	1
DEPARTMENT	1
gas bike	1
MOBILITY S	1
FDNY PICKU	1
COMMAND PO	1
EXCAVATOR	1
Suburban	1

Chevy carg	1
E SKATEBOA	1
BOOM LIFT/	1
dirtbike	1
G AMB	1
gas bicycl	1
dirt bike	1
electric s	1
self insur	1
SKATBOARD	1
comm	1
FDNY RIG	1
WH	1
Food cart	1
WHEELCHAIR	1
NYC FIRE T	1
util truck	1
EMT Truck	1
Flat bed	1
Utility ca	1
MO PED	1
spc	1
3 Wheel Sc	1
CARGO TRAI	1
AMBULAMCE	1
RMP	1
revel	1
BOOM 60FT	1
Trailor	1
UHUAL	1
DEPT VAN #	1
pedicab	1
SKATE	1
PICK-	1
Minicycle	1
Sprin	1
Parce	1
3 WHE	1
POLIC	1
SANTI	1
FLAT/	1
ESU RESCUE	1
cross	1
E REVEL SC	1
GEN AMBUL	1
MOVING VAN	1
UTILITY	1
SWT	1
utility	1
firetruck	1
Bucket Tru	1
BACK HOE	1
FDNY EMS V	1
fdny truck	1
D/V WB	1
PICKUP TRU	1

postal ser	1
ford econo	1
FREIGHTLIN	1
E-BIKE	1
government	1
Ford sprin	1
FUSION	1
NYC FD	1
KUBOT	1
E-SKA	1
E Bik	1
11 PA	1
Chevy	1
LEFT THE S	1
MAN L	1
LADDER TRU	1
GATOR	1
HAND	1
Pedic	1
boom crane	1
2 HOR	1
MTA b	1
Depar	1
Postal tru	1
E-bik	1
ESU REP	1
school bus	1
Razor scoo	1
motor scoo	1
UNKNOWN VE	1
PICK-UP TR	1
uhaul truc	1
DIRTBIKE	1
Pick Truck	1
PALFINGER	1
VEHICLE 2	1
FORKLIFT	1
E-SKATEBOA	1
VENDOR CHA	1
Livery Omn	1
SPINTER VA	1
600AJ	1
NYC DOT	1
BLU BUS	1
PICK RD	1
NYC fire t	1
pickup	1
ambulence	1
delivery	1
Forklift t	1
ESCAVATOR	1
BOB C	1
COURI	1
Unkow	1
EMRGN	1
BROOM	1

work	1
COUPE	1
Bicyc	1
NYC D	1
passe	1
Hrse	1
SEA	1
Barri	1
Log	1
NAVIG	1
NTTRL	1
Mail	1
omm	1
nypd	1
LTRL	1
Const	1
Picku	1
RGS	1
U-Hau	1
Cater	1
UTILITY VA	1
OBJEC	1
SMART	1
Porta	1
FEDEX	1
UTLL	1
COM.	1
HOTDO	1
Jeep	1
GOV'T	1
CONT	1
conv	1
Laund	1
Refri	1
ENCLO	1
conta	1
Fork	1
E BIK	1
PCH	1
White	1
armor	1
omr	1
OIL T	1
Bucketload	1
dp	1
carri	1
Schoo	1
omni	1
3DC-	1
tr	1
ice c	1
SD	1
PRIVA	1
U.S. POSTA	1
Pavin	1
GE/SC	1

sgws	1
E-MOT	1
CHART	1
Backh	1
Semi	1
BK	1
BULLD	1
Food	1
SELF-	1
UPS T	1
VAN W	1
Pick-	1
BOBCA	1
013	1
GMC V	1
nissa	1
TRLPM	1
DELV.	1
scaff	1
WHITE	1
mot s	1
LP	1
VAN/T	1
frieg	1
Veh L	1
B5-44	1
UNKNOW	1
Limou	1
Tract	1
trac	1
Subur	1
KP160	1
ROOD	1
UTLIT	1
REPAI	1
del	1
fedx	1
Renta	1
picku	1
STERL	1
SCOMM	1
trl	1
Budge	1
Shove	1
Posta	1
winne	1
WORK	1
PET	1
1	1
VESPA	1
unkow	1
pedic	1
mail	1
uhaul	1
WINNE	1
semi-	1

tower	1
mack	1
Uhaul	1
white	1
5X8 T	1
NT TR	1
VAV	1
const	1
LADDER 34	1
SE	1
Truc	1
DLVR	1
Tank	1
RD/S	1
boxtr	1
UK	1
TANKE	1
12 PA	1
NYC F	1
fed e	1
POST	1
big r	1
mac 1	1
bulld	1
JUNST	1
GENIE	1
ES	1
UNK T	1
NYC S	1
U.S.P	1
trk	1
hino	1
nyc s	1
DUMPT	1
Comme	1
YLL P	1
Carri	1
ROAD	1
Tow	1
oml	1
ACCEE	1
stak	1
TRLR	1
OFFIC	1
road	1
2000	1
MOVIN	1
amb	1
FED E	1
stree	1
OMT/T	1
GREEN	1
U-HAL	1
NISSA	1
scoot	1
Flatb	1

PC	1
WHIT	1
vAN	1
3 Whe	1
REFR	1
RV/Tr	1
nyu s	1
CONTR	1
Pick	1
couri	1
26 ft	1
US MA	1
Road	1
CARGO	1
HOSRE	1
CON E	1
DODGE	1
HD To	1
Com	1
OMNIB	1
DEIV	1
SC	1
REFRG	1
DEI V	1
LCOM	1
US po	1
Ice C	1
NYFD	1
freig	1
MI/FU	1
Gene	1
2dr	1
skate	1
omnib	1
ACCES	1
TL	1
ECONO	1
LEFT	1
Delie	1
DB	1
GMC	1
wg	1
FD LA	1
post	1
SKID	1
QMZ	1
Rolli	1
P/U	1
PASSE	1
isuzu	1
engin	1
srf	1
GENUI	1
fre	1
SHORT	1
School bus	1

GAS MOPED	1
MC	1
Scooter ga	1
Fdny Fire	1
E SCOOTER	1
PETIT CAB	1
tlc	1
X Amb	1
Pickup tru	1
NYS Ambula	1
Van (Trans	1
Work truck	1
Usps mail	1
In Line Sk	1
FD TRUCK	1
Tractor tr	1
Fdny ems	1
PEPSI DELI	1
utv	1
UTV	1
ELECTRIC C	1
Ford	1
Fork Lift	1
standing s	1
Small Bus	1
Pick-up tr	1
Workcart	1
FDNY truck	1
Push cart	1
FOGLIFT	1
FireTruck	1
Fdny truck	1
AMBLUANCE	1
TF	1
ARCIMOTO	1
NYCHA FORK	1
E scooter	1
Sprinter v	1
E Bike	1
Access A R	1
Siting mop	1
SCHOOL BUS	1
FOOD VENDE	1
FD APPARAT	1
FedEx Truc	1
E-SCOOTER	1
BOBCAT/FOR	1
left scene	1
White box	1
FD Ladder	1
garbage tr	1
F550	1
ONE WHEEL	1
box van	1
street swe	1
COMMERICIA	1

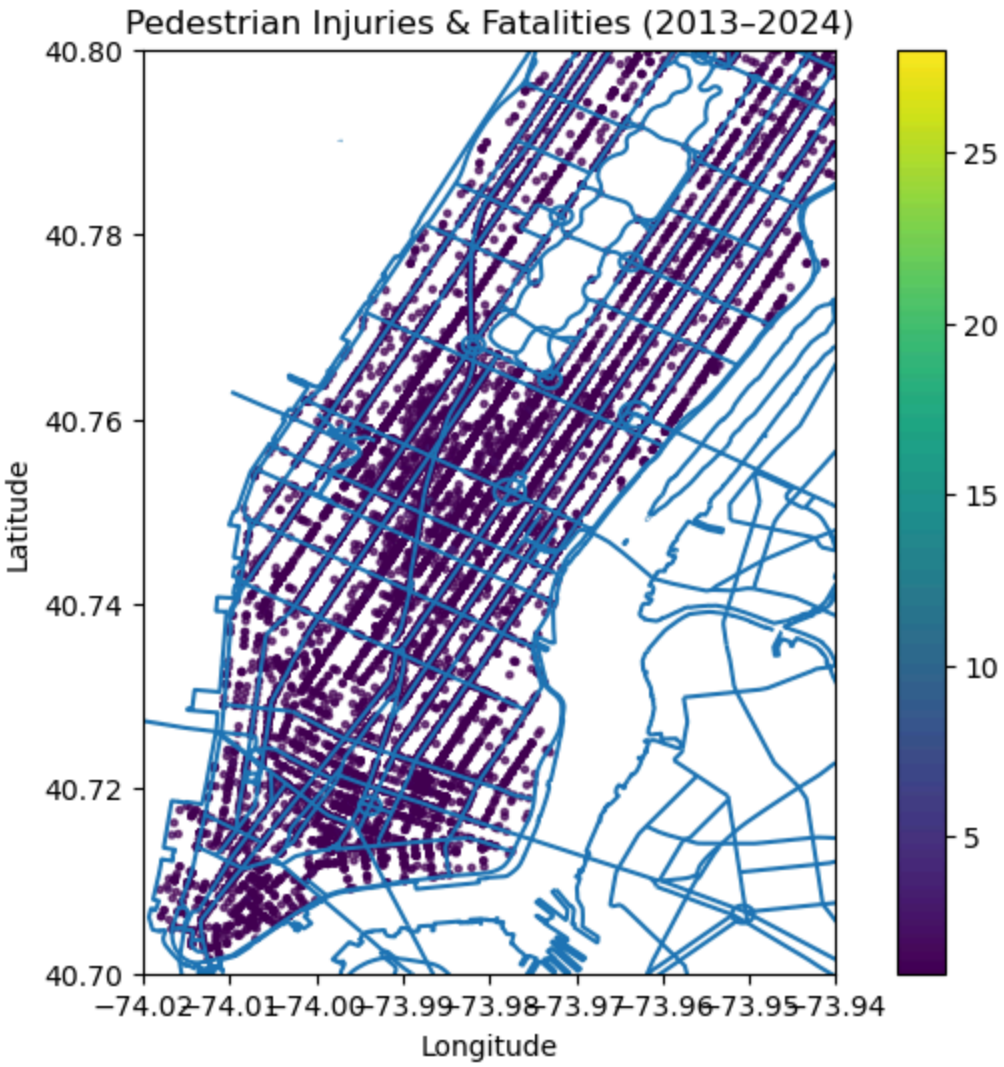
```
E-bike                          1
Street swe                      1
PEDAL BIKE                      1
Pick up Tr                      1
Boom Lift                       1
HORSE DRAW                      1
MINI VAHN                       1
Pickup tow                      1
e-scooter                       1
Passenger                       1
VAN (TRANS                      1
Garbage tr                      1
Van Truck                       1
Rmp                             1
Limo                            1
FORD F-150                      1
PASSENGER                       1
HORSE TRAI                      1
FORD AMBUL                      1
Golf cart                       1
Tow truck                       1
tow truck                       1
JEEP                            1
unicycle                        1
KME                             1
Seagrave T                      1
RED                             1
Horse Carr                      1
US Postal                       1
ROAD SWEEP                      1
MOTORIZED                       1
Bulldozer                       1
Fdny fire                       1
MECHANICAL                      1
usps truck                      1
FDNY SUPER                      1
NYPD Tow t                      1
UNKOWM                          1
CARGO VAN                       1
COMM VAN                        1
push scoot                      1
Rds                             1
CONSTRUCTI                      1
Left scene                      1
Moving van                      1
FORD SUPER                      1
========================================
```

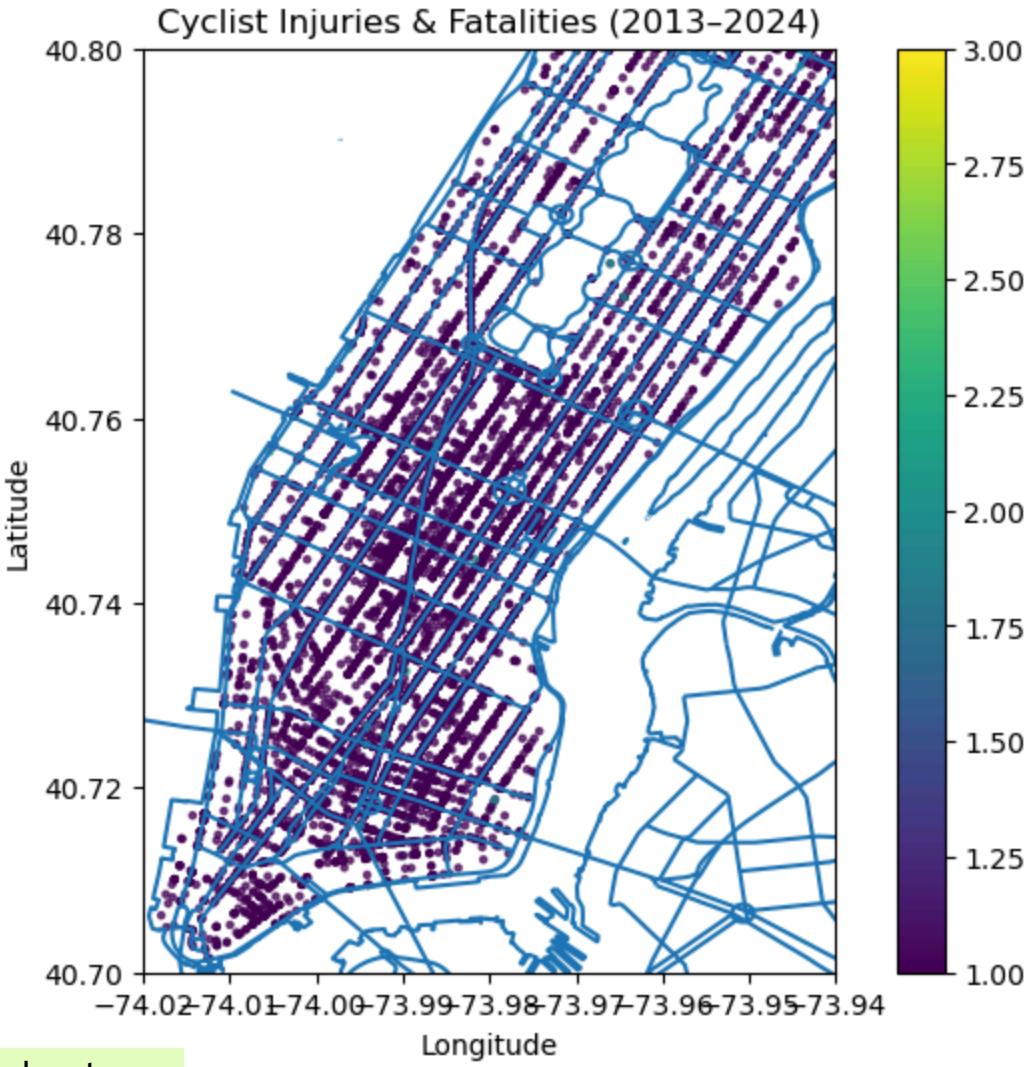
Q2: Part 4: Spatial maps done by me not a requirement

In [31]:
```
import pandas as pd
import geopandas as gpd
from shapely import wkt
import matplotlib.pyplot as plt

# 1. Load & filter Manhattan collisions
acc = pd.read_csv(
    'Motor-Vehicle-Collisions.csv',
```

```
    parse_dates=['CRASH DATE'],
    low_memory=False
)
acc = acc[
    (acc['BOROUGH']=='MANHATTAN') &
    acc['LATITUDE'].notna() & acc['LONGITUDE'].notna()
]

# 2. Convert to GeoDataFrame
gdf_acc = gpd.GeoDataFrame(
    acc,
    geometry=gpd.points_from_xy(acc.LONGITUDE, acc.LATITUDE),
    crs='EPSG:4326'
)

# 3. Load street & shoreline layers
streets   = pd.read_csv('dcm-nyc-major-streets.csv')
shoreline = pd.read_csv('nyc-shoreline.csv')
streets['geometry']   = streets['the_geom'].apply(wkt.loads)
shoreline['geometry'] = shoreline['the_geom'].apply(wkt.loads)
gdf_str   = gpd.GeoDataFrame(streets,   geometry='geometry', crs='EPSG:4326')
gdf_shore = gpd.GeoDataFrame(shoreline, geometry='geometry', crs='EPSG:4326')

# 4A. Pedestrian heatmap
ped = gdf_acc[
    (gdf_acc['NUMBER OF PEDESTRIANS INJURED'] > 0) |
    (gdf_acc['NUMBER OF PEDESTRIANS KILLED'] > 0)
].copy()
ped['total_events'] = ped['NUMBER OF PEDESTRIANS INJURED'] + ped['NUMBER OF PEDESTR

fig, ax = plt.subplots(figsize=(8,6))
gdf_shore.plot(ax=ax)
gdf_str.plot(ax=ax)
ped.plot(
    ax=ax,
    column='total_events',
    markersize=5,
    legend=True,
    alpha=0.7
)
ax.set_xlim(-74.02, -73.94)
ax.set_ylim(40.70, 40.80)
ax.set_title('Pedestrian Injuries & Fatalities (2013–2024)')
ax.set_xlabel('Longitude'); ax.set_ylabel('Latitude')
plt.show()

# 4B. Cyclist heatmap
cyc = gdf_acc[
    (gdf_acc['NUMBER OF CYCLIST INJURED'] > 0) |
    (gdf_acc['NUMBER OF CYCLIST KILLED'] > 0)
].copy()
cyc['total_events'] = cyc['NUMBER OF CYCLIST INJURED'] + cyc['NUMBER OF CYCLIST KIL

fig, ax = plt.subplots(figsize=(8,6))
gdf_shore.plot(ax=ax)
gdf_str.plot(ax=ax)
```

```
cyc.plot(
    ax=ax,
    column='total_events',
    markersize=5,
    legend=True,
    alpha=0.7
)
ax.set_xlim(-74.02, -73.94)
ax.set_ylim(40.70, 40.80)
ax.set_title('Cyclist Injuries & Fatalities (2013–2024)')
ax.set_xlabel('Longitude'); ax.set_ylabel('Latitude')
plt.show()
```

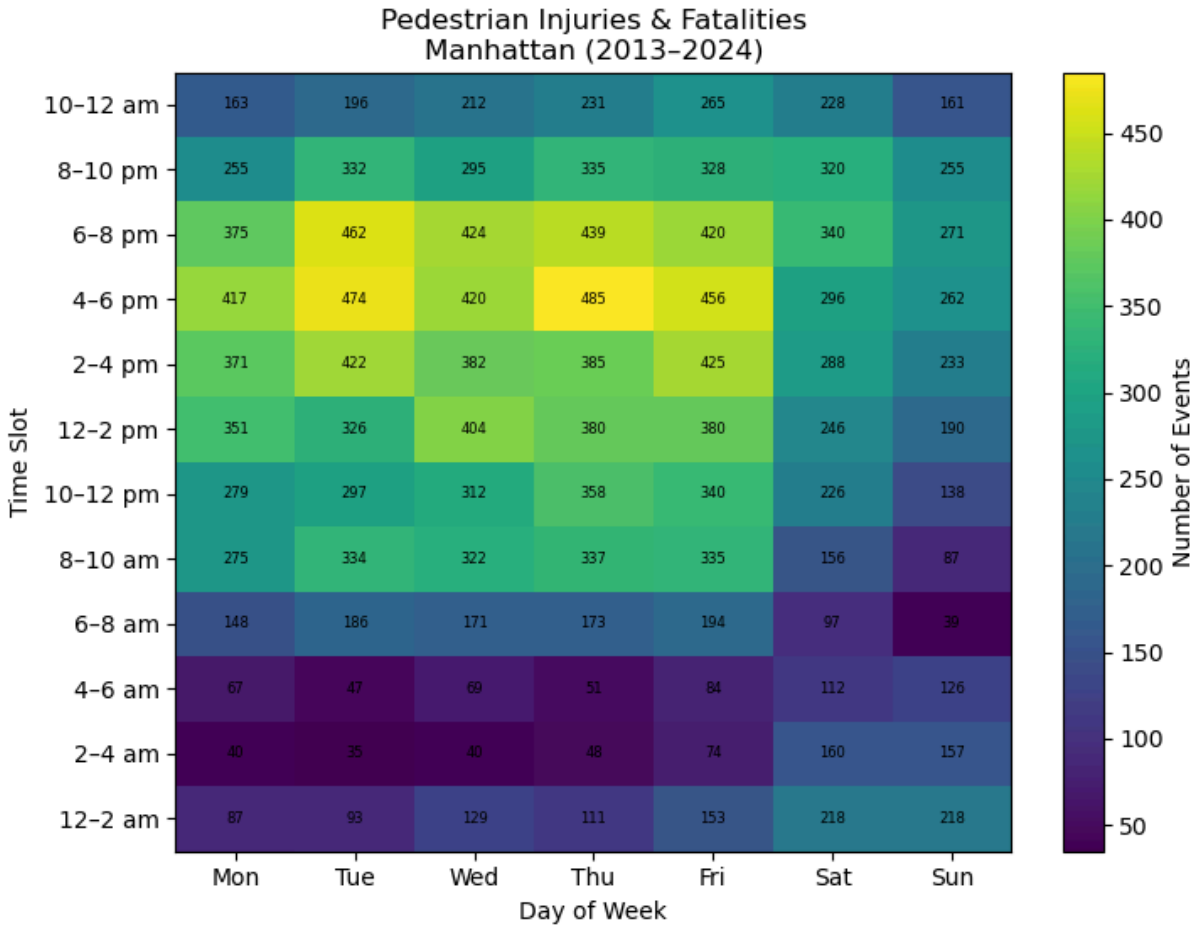


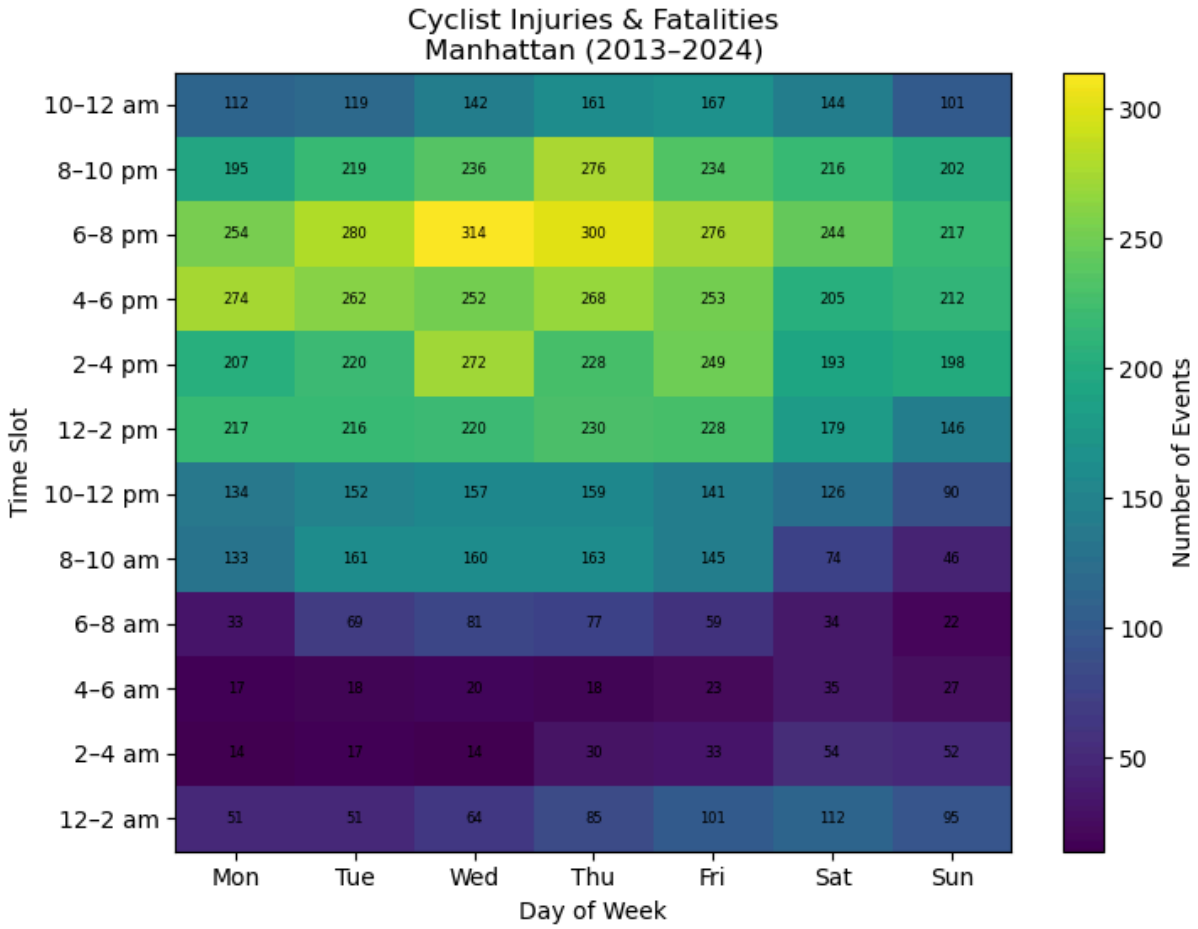


Q2: Part 4 heatmaps

In [45]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

# Prepare empty 12×7 arrays
ped_heat = np.zeros((12, 7), dtype=int)
cyc_heat = np.zeros((12, 7), dtype=int)

cols = [
    'CRASH DATE','CRASH TIME','BOROUGH','LATITUDE','LONGITUDE',
    'NUMBER OF PEDESTRIANS INJURED','NUMBER OF PEDESTRIANS KILLED',
    'NUMBER OF CYCLIST INJURED','NUMBER OF CYCLIST KILLED'
]

for chunk in pd.read_csv(
    'Motor-Vehicle-Collisions.csv',
    usecols=cols,
    parse_dates=['CRASH DATE'],
    chunksize=200_000,
    low_memory=False
):
    # Filter and then make an explicit copy
    m = chunk[
```

```
        (chunk['BOROUGH']=='MANHATTAN') &
        chunk['LATITUDE'].notna() &
        chunk['LONGITUDE'].notna()
    ].copy()         # <<--- copy() here

    if m.empty:
        continue

    # Now these .loc assignments won't warn
    hrs = m['CRASH TIME'].str.extract(r'^(\d{1,2})')[0]
    m.loc[:, 'Hour'] = pd.to_numeric(hrs, errors='coerce').fillna(-1).astype(int)
    m = m[m['Hour'] >= 0]  # drop bad times

    m.loc[:, 'Slot'] = (m['Hour'] // 2).clip(upper=11)
    m.loc[:, 'Day']  = m['CRASH DATE'].dt.weekday

    m.loc[:, 'Ped_events'] = (
        m['NUMBER OF PEDESTRIANS INJURED'] +
        m['NUMBER OF PEDESTRIANS KILLED']
    )
    m.loc[:, 'Cyc_events'] = (
        m['NUMBER OF CYCLIST INJURED'] +
        m['NUMBER OF CYCLIST KILLED']
    )

    # Aggregate
    agg = m.groupby(['Slot','Day'])[['Ped_events','Cyc_events']].sum()
    for (slot, day), row in agg.iterrows():
        ped_heat[int(slot), int(day)] += int(row['Ped_events'])
        cyc_heat[int(slot), int(day)] += int(row['Cyc_events'])

# Labels
days  = ['Mon','Tue','Wed','Thu','Fri','Sat','Sun']
slots = [
    "12–2 am","2–4 am","4–6 am","6–8 am","8–10 am","10–12 pm",
    "12–2 pm","2–4 pm","4–6 pm","6–8 pm","8–10 pm","10–12 am"
]

def plot_heat(data, title):
    fig, ax = plt.subplots(figsize=(8,6))
    im = ax.imshow(data, origin='lower', aspect='auto')
    ax.set_xticks(range(7)); ax.set_xticklabels(days)
    ax.set_yticks(range(12)); ax.set_yticklabels(slots)
    ax.set_xlabel('Day of Week'); ax.set_ylabel('Time Slot')
    ax.set_title(title)
    for i in range(12):
        for j in range(7):
            ax.text(j, i, data[i,j], ha='center', va='center', fontsize=6)
    fig.colorbar(im, ax=ax, label='Number of Events')
    plt.show()

plot_heat(ped_heat, 'Pedestrian Injuries & Fatalities\nManhattan (2013–2024)')
plot_heat(cyc_heat, 'Cyclist Injuries & Fatalities\nManhattan (2013–2024)')
```


Pedestrian Injuries & Fatalities
Manhattan (2013–2024)
Time Slot
10–12 am
8–10 pm
6–8 pm
4–6 pm
2–4 pm
12–2 pm
10–12 pm
8–10 am
6–8 am
4–6 am
2–4 am
12–2 am
163 196 212 231 265 228 161
255 332 295 335 328 320 255
375 462 424 439 420 340 271
417 474 420 485 456 296 262
371 422 382 385 425 288 233
351 326 404 380 380 246 190
279 297 312 358 340 226 138
275 334 322 337 335 156 87
148 186 171 173 194 97 39
67 47 69 51 84 112 126
40 35 40 48 74 160 157
87 93 129 111 153 218 218
Mon
Tue
Wed
Thu
Fri
Sat
Sun
Day of Week
Number of Events
450
400
350
300
250
200
150
100
50


Cyclist Injuries & Fatalities
Manhattan (2013–2024)
Time Slot
10–12 am
8–10 pm
6–8 pm
4–6 pm
2–4 pm
12–2 pm
10–12 pm
8–10 am
6–8 am
4–6 am
2–4 am
12–2 am
112 119 142 161 167 144 101
195 219 236 276 234 216 202
254 280 314 300 276 244 217
274 262 252 268 253 205 212
207 220 272 228 249 193 198
217 216 220 230 228 179 146
134 152 157 159 141 126 90
133 161 160 163 145 74 46
33 69 81 77 59 34 22
17 18 20 18 23 35 27
14 17 14 30 33 54 52
51 51 64 85 101 112 95
Mon Tue Wed Thu Fri Sat Sun
Day of Week
Number of Events
300 250 200 150 100 50

In [ ]:

Question 2.

Part 5)

Over a ten-year stretch, we looked at about 318,000 crashes in Manhattan. Around 17% of those led to at least one person getting hurt, and just under 0.1% were fatal. Altogether, over 64,710 people were injured and about 313 died. Out of those, pedestrians made up around 20,599 injuries and 224 deaths, and cyclists had close to 12,252 injuries and about 33 deaths. Most of the crashes involved passenger cars, taxis, sedans, and SUVs—basically what you'd expect in city traffic. When you map it out, you see that crashes, especially the ones that hurt or kill pedestrians and cyclists, are packed along the big avenues in Midtown and Lower Manhattan. Time-wise, the worst spikes happen during weekday rush hours (6–8 a.m. and 4–6 p.m.) and early mornings on weekends. That’s when people on foot or bike are most at risk. All of this points to one thing: if the city focused on lighting, added protected bike lanes in those hot zones, and cracked down more during peak hours, it could seriously help cut injuries and save lives.