In [ ]:
```
# TASK – 4: Optimization Model
## Smart EV Charging Station Location & Capacity Optimization
### Internship Submission – Rishabrizan Ganesh
```

In [ ]:
```
## 1. Business Problem Statement

A city wants to install EV charging stations at selected locations.

Each location:
- Has installation cost
- Has maximum charging capacity#
- Serves a specific demand zone

The city has:
- Limited budget
- Minimum total demand requirement

Objective:
Minimize total installation cost while meeting EV charging demand.
```

In [ ]:
```
## 2. Given Data

Locations Available: 5

| Location | Installation Cost | Max Capacity |
|----------|-------------------|--------------|
| L1 | 50000 | 40 |
| L2 | 70000 | 60 |
| L3 | 40000 | 30 |
| L4 | 60000 | 50 |
| L5 | 55000 | 45 |

Total Budget Available = ₹2,00,000
Minimum Demand Required = 120 Charging Units
```

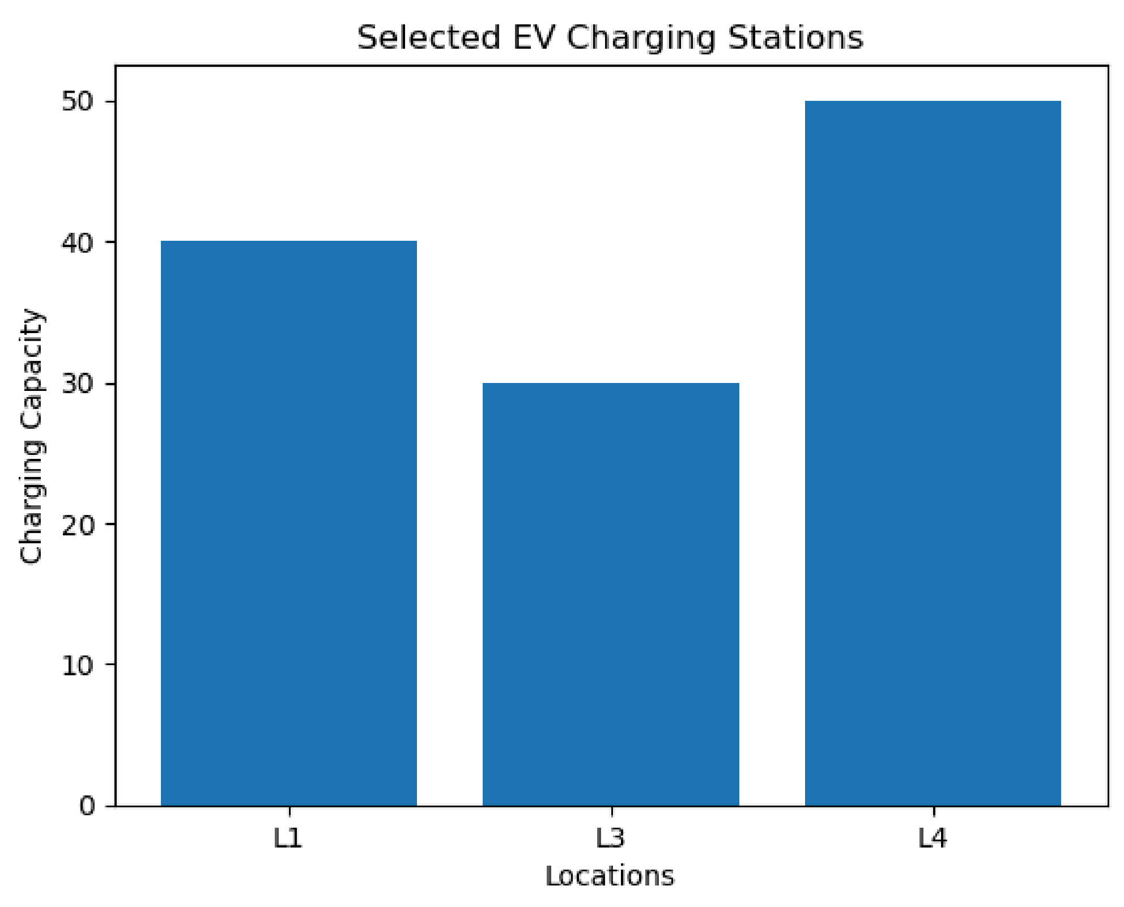
In [1]:
```
!pip install pulp
```
```
Requirement already satisfied: pulp in c:\users\risha\anaconda3\lib\site-packages (3.3.0)
```

In [3]:
```
import pulp
import matplotlib.pyplot as plt
```

In [4]:
```
# Create model (Minimization Problem)
model = pulp.LpProblem("EV_Charging_Optimization", pulp.LpMinimize)

# Installation cost
cost = {
    "L1": 50000,
    "L2": 70000,
    "L3": 40000,
    "L4": 60000,
    "L5": 55000
}

# Capacity
capacity = {
    "L1": 40,
    "L2": 60,
    "L3": 30,
    "L4": 50,
    "L5": 45
}

locations = cost.keys()

# Decision Variables (Binary: Install or Not)
x = pulp.LpVariable.dicts("Install",
                          locations,
                          cat="Binary")

# Objective Function (Minimize total cost)
model += pulp.lpSum(cost[i] * x[i] for i in locations)

# Budget Constraint
model += pulp.lpSum(cost[i] * x[i] for i in locations) <= 200000

# Demand Satisfaction Constraint
model += pulp.lpSum(capacity[i] * x[i] for i in locations) >= 120
```

In [5]:
```
model.solve()

print("Status:", pulp.LpStatus[model.status])
print("\nSelected Locations:")

total_cost = 0
total_capacity = 0

for i in locations:
    if x[i].varValue == 1:
        print(i)
        total_cost += cost[i]
        total_capacity += capacity[i]

print("\nTotal Installation Cost:", total_cost)
print("Total Capacity Achieved:", total_capacity)
```
```
Status: Optimal

Selected Locations:
L1
L3
L4

Total Installation Cost: 150000
Total Capacity Achieved: 120
```

In [6]:
```
selected = [i for i in locations if x[i].varValue == 1]
selected_capacity = [capacity[i] for i in selected]

plt.bar(selected, selected_capacity)
plt.title("Selected EV Charging Stations")
plt.xlabel("Locations")
plt.ylabel("Charging Capacity")
plt.show()
```



In [7]:
```
## 3. Mathematical Formulation

Decision Variable:
Xi = 1 if location is selected, 0 otherwise

Objective:
Minimize Σ (Cost_i × Xi)

Subject to:

1. Budget Constraint:
Σ (Cost_i × Xi) ≤ 200000

2. Demand Constraint:
Σ (Capacity_i × Xi) ≥ 120

3. Binary Constraint:
Xi ∈ {0,1}
```
```
  Cell In[7], line 7
    Minimize Σ (Cost_i × Xi)
                       ^
SyntaxError: invalid character '×' (U+00D7)
```

In [8]:
```
## 4. Key Insights
• The model selects the minimum-cost combination of locations.
• Demand requirement is fully satisfied.
• Budget constraint ensures financial feasibility.
• Binary optimization helps in strategic infrastructure planning.
• This model supports sustainable smart city development.
```
```
  Cell In[8], line 3
    • The model selects the minimum-cost combination of locations.
    ^
SyntaxError: invalid character '•' (U+2022)
```

In [9]:
```
model.constraints.pop("Demand_Satisfaction_Constraint", None)
model += pulp.lpSum(capacity[i] * x[i] for i in locations) >= 150
model.solve()
```
Out[9]:
```
1
```

In [ ]:
```
## 5. Conclusion

This project demonstrates how Integer Linear Programming can solve real-world infrastructure planning problems.
```

```
Using PuLP, we determined the optimal set of EV charging stations that minimize installation cost while satisfying city-wide charging demand within budget constraints.

Optimization enables data-driven, sustainable, and financially efficient decision-making.
```