

newfeatures Collect Data

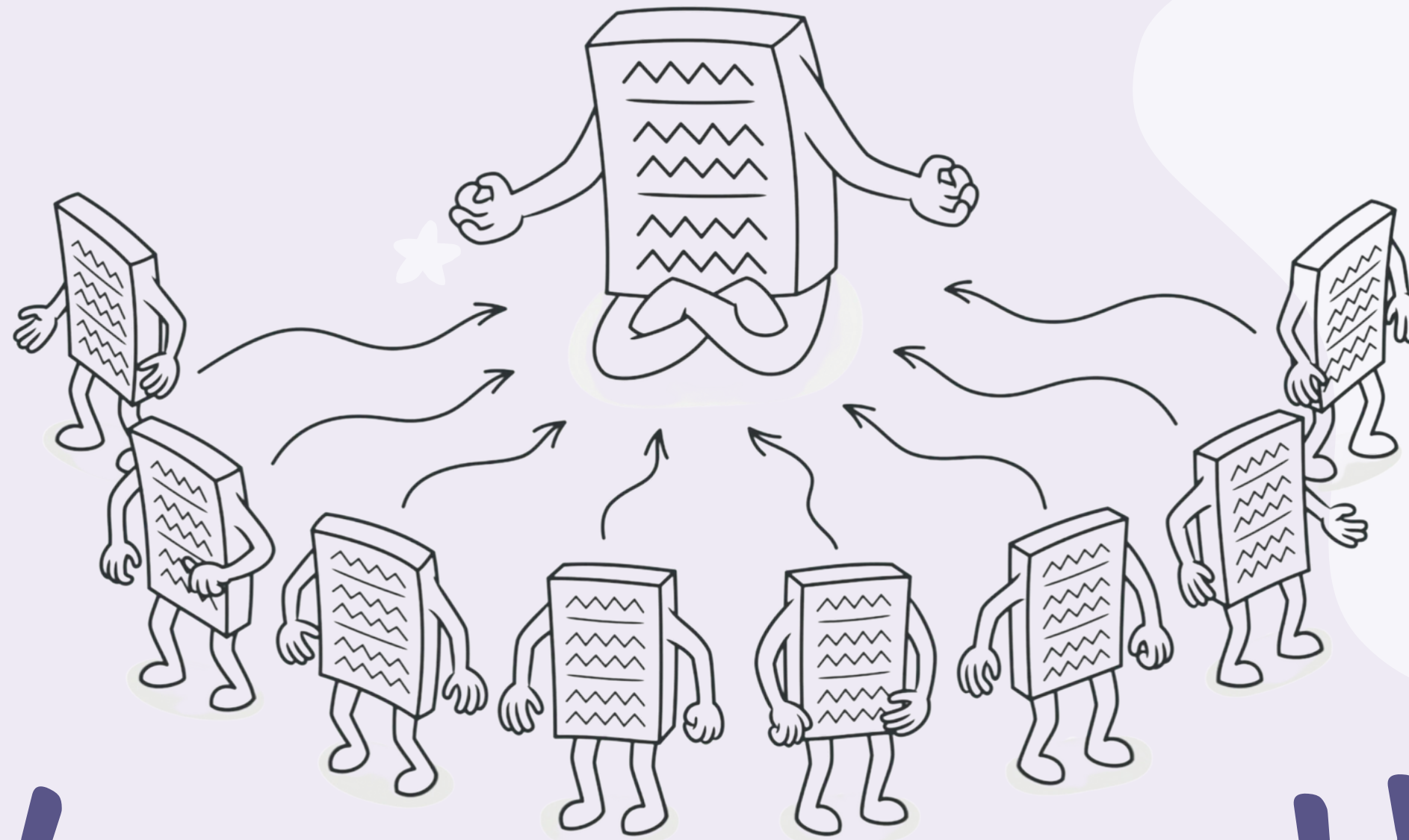
Presentation by Minh Uyen Nguyen



Collect data

1	Entity Name	Delivery Volume	Entity Speed	Trip Distance	Entity Strategy	Battery Status	Trip Duration
2	Drone	3	30	2926.19	dijkstra	332.605	169.869
3	Robot 1	1	30	453.583	astar	0	17.8008
4	Robot 2	1	30	360.945	dfs	0	18.6003
5	Robot 3	1	30	390.054	dijkstra	0	19.7543





Singleton design pattern

The Singleton class guarantees that only one instance of itself is produced and gives a global access point to that instance.



Back-end Implementation



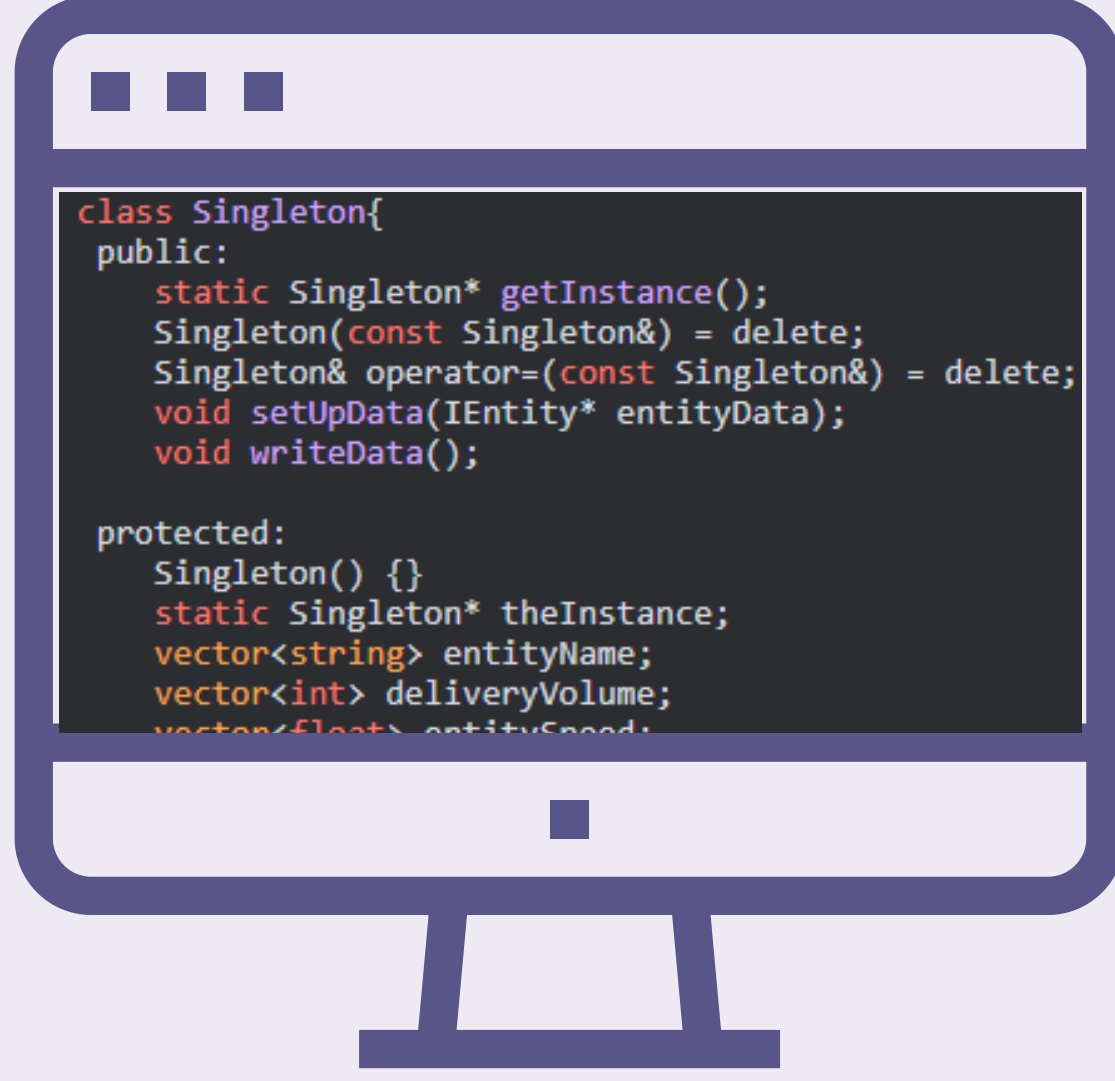
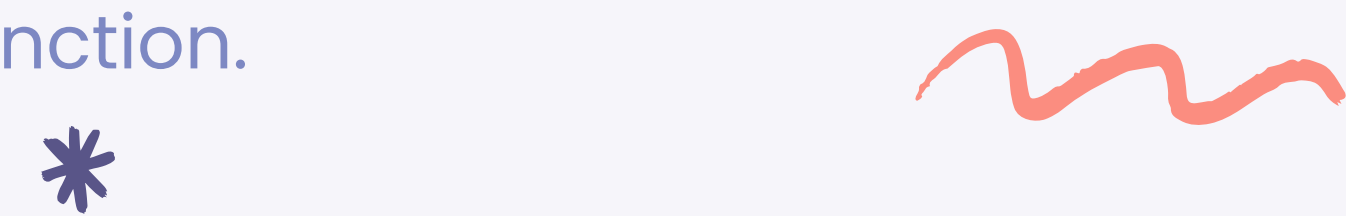
Key methods:

theInstance: A static reference to the Singleton class's unique instance.

getInstance(): A static method that returns the Singleton class's unique instance.

setUpData(IEntity* entity): The function accepts an IEntity pointer as an argument and set up the method of collecting data

writeData(): The modified data is written to a CSV file using this function.



```
class Singleton{
public:
    static Singleton* getInstance();
    Singleton(const Singleton&) = delete;
    Singleton& operator=(const Singleton&) = delete;
    void setUpData(IEntity* entityData);
    void writeData();

protected:
    Singleton() {}
    static Singleton* theInstance;
    vector<string> entityName;
    vector<int> deliveryVolume;
    vector<float> entitySpeed;
```

Front-end implementation *

<button>Get the CSV</button> A client development interactive element.
toggleCSV(): JavaScript function that sends a "writeCSV" command.
cmd = "writeCSV": The receive command in transit_service.cc function



START