In the project, the Node and Graph classes are closely related, as they are used together to represent the graph structure and work with vertices, edges, and their weights.

Node:

Node Class represents a single vertex of the graph, which is used to find a path (for example, in the A* algorithm).

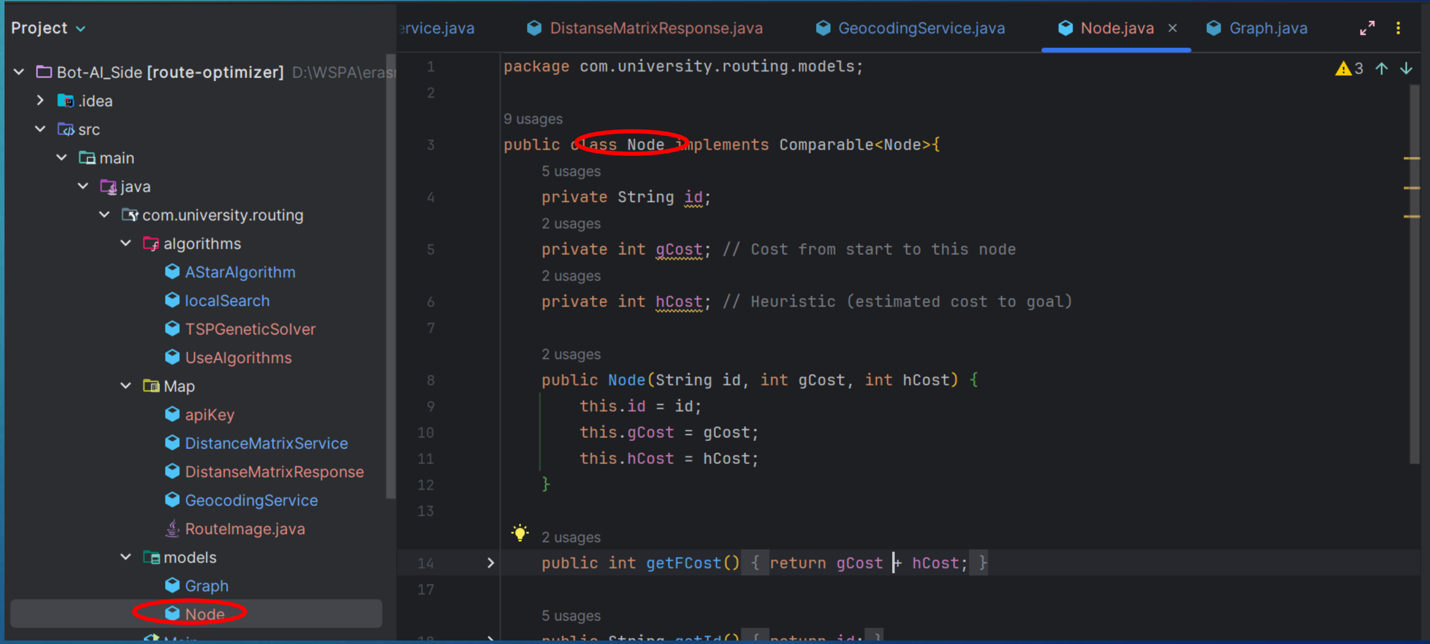
- *Each Node has:*

1. **id**: a unique vertex identifier (coordinates).
2. **gCost:** The cost of the path from the starting point to this vertex.
3. **hCost**: A heuristic estimating the distance from this vertex to the endpoint.
4. **fCost**: The sum of gCost + hCost, used to evaluate priorities in the A* algorithm.

it is calculated dynamically via the getFCost() method. fCost is not stored in the Node object, but is calculated each time as the sum of gCost + hCost.

- *Key methods:*

1. **getfocus()**: returns the overall priority of the vertex.
2. **compareTo()**: Compares two vertices by their fCost for ordering.



```java
package com.university.routing.models;

9 usages
public class Node implements Comparable<Node>{
    5 usages
    private String id;
    2 usages
    private int gCost; // Cost from start to this node
    2 usages
    private int hCost; // Heuristic (estimated cost to goal)

    2 usages
    public Node(String id, int gCost, int hCost) {
        this.id = id;
        this.gCost = gCost;
        this.hCost = hCost;
    }

    2 usages
    public int getFCost() { return gCost + hCost; }

    5 usages
    public String getId() { return id; }
```