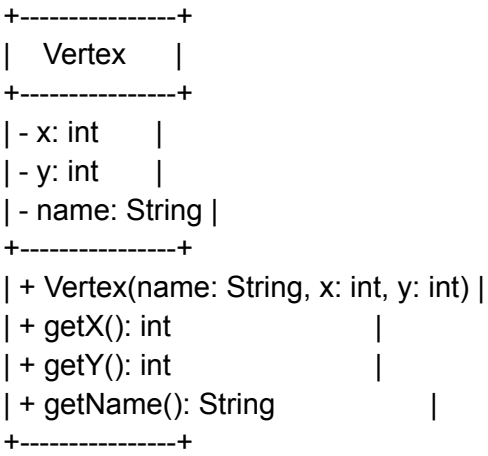
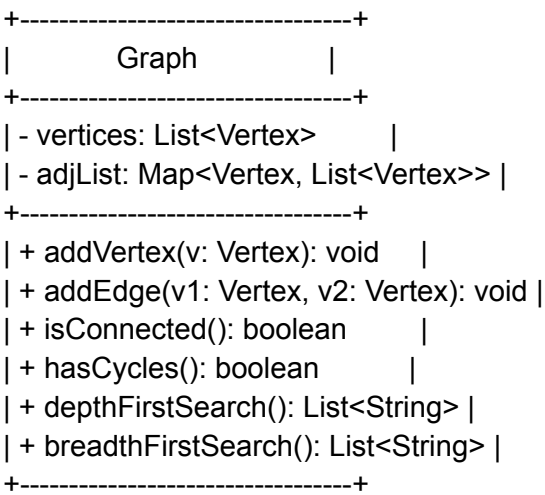
Project 4 Documentation

UML Class Diagram:

Vertex (Immutable Class)



Graph (Logic Class)



GraphPane (JavaFX Display)

```
+---------------------------------------+
|            GraphPane extends Pane     |
+---------------------------------------+
| - graph: Graph                        |
| - vertexMap: Map<String, Vertex>      |
+---------------------------------------+
| + handleMouseClick(e: MouseEvent): void |
| + drawEdge(v1: Vertex, v2: Vertex): void |
+---------------------------------------+
```

MainApp (GUI and Event Handlers)

```
+---------------------+
|      MainApp        |
+---------------------+
| - graphPane: GraphPane |
| - textField1, textField2, messageLabel: TextField/Label |
+---------------------+
| + main(args: String[]): void |
| + createUI(): void          |
+---------------------+
```

Test Plan:

Test Case	Input / Action	Expected Result	Aspect Tested
1. Add vertex A	Click once on canvas	Vertex "A" appears	Vertex creation
2. Add vertex B	Click again in a new location	Vertex "B" appears	Vertex labeling
3. Add edge A-B	Enter "A" and "B" in	Line appears	Edge creation

	text fields → Click "Add Edge"	between A and B	
4. Invalid edge	Enter "A" and "Z" → Click "Add Edge"	Error: one or both vertices not found	Input validation
5. Duplicate edge	Add edge "A"-"B" again	Error: edge already exists	Duplicate edge handling
6. Check isConnected (2 vertices, 1 edge)	Click "Is Connected"	Message: "Graph is connected."	Connectivity check
7. Check hasCycles (2 vertices, 1 edge)	Click "Has Cycles"	Message: "Graph has no cycles."	Cycle detection
8. Add cycle (A-B, B-C, C-A)	Add 3 vertices and 3 edges	"Has Cycles" shows "Graph has cycles."	Cycle detection
9. DFS traversal	Click "Depth First Search"	DFS: A → B → C (example order)	Depth-first search logic
10. BFS traversal	Click "Breadth First Search"	BFS: A → B → C (example order)	Breadth-first search logic
11. Graph resets visually	Restart app and re-add nodes	Vertex labels reset to A	GUI state management

Lessons Learned:

This project helped reinforce my understanding of graph data structures and traversal algorithms such as DFS and BFS. It also challenged me to integrate logical data models with GUI components using JavaFX. I learned how critical it is to ensure that equals() and hashCode() are correctly implemented for custom objects, such as Vertex, especially when using collections like HashMap and HashSet. I also gained experience with managing JavaFX runtime configurations in VS Code. I improved my ability to debug both logic errors and build path issues in a real-world development environment. Overall, the project enhanced both my algorithmic thinking and GUI programming skills.