CHAPTER 4
ANALYSIS AND DESIGN

4.1 Analysis

Minimum spanning tree can be implemented on an undirected weighted graph, so weighted graph is required in this program. A weighted graph consist of nodes, edges, and its weight. The nodes need to be determined first. After the nodes is determined, it need to be linked with lines so it can form edges.

In order to analyze comparison between Kruskal and Prim algorithm, several graphs are required. In this project, there are four graphs that needed to be tested. Each graph has different form. By doing test on different graph, it will be easier to analyze and draw a conclusion about the comparison.

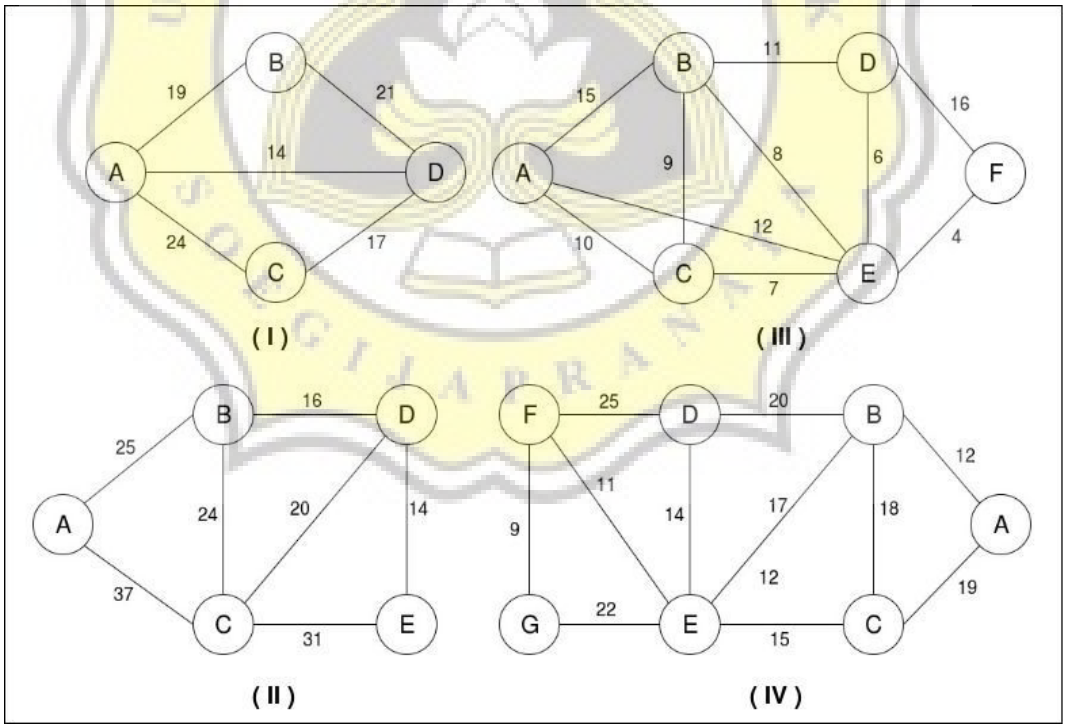

Illustration 4.1: Weighted Graph Samples

Based on the graph sample above, each node is named with character. The numbers on each line are weight of edge. For example on sample I, the distance between node A and node B is 19, and distance between node C and node D is 17. Each graph has different number of nodes and edges. The table below contains detail of each graph that will be used.

Table 4.1: Graph Sample Detail

Sample	Number of Nodes	Number of Edges
I	4	5
II	5	7
III	6	10
IV	7	11

The program will generate a weighted graph, by using node and edge data from graph sample above. Data then will be stored in a text file. This file contains information of nodes relation that is required to generate a graph. The information consist of start node, end node, and weight of the edges. Method that can load data from text file will be needed later.

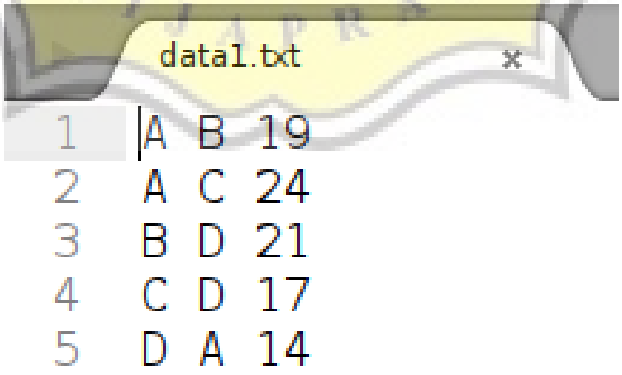

Illustration 4.2: Graph Information on Text File

4.2 Design

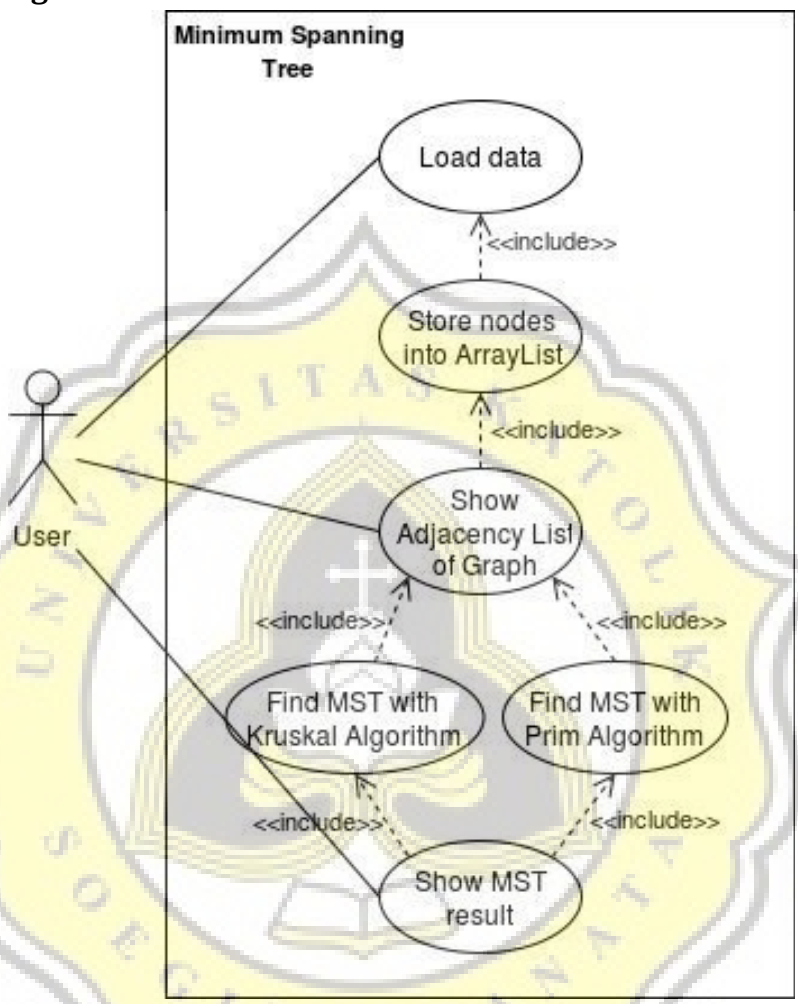

Illustration 4.3: Use Case Diagram

Based on the use case diagram above, user is able to determine graph data. Data that will be processed should be written in a text file. After start node, end node, and cost are inputed, program will process those data and generates a weighted graph. The weighted graph will be represented in adjacency list form. Then program will generate minimum spanning tree using Kruskal and Prim algorithm. After generating process, system will display minimum spanning tree result by using Kruskal and Prim algorithm.

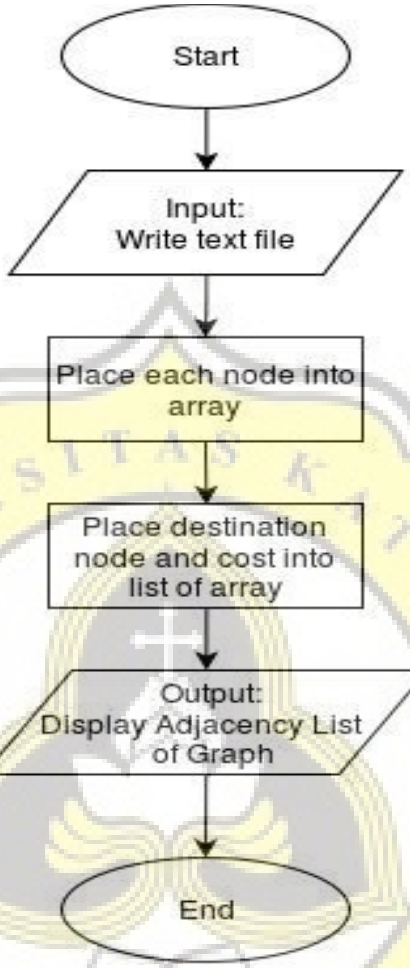

Illustration 4.4: Generating Weighed Graph Flow Chart

A weighted graph consists of nodes, edges, and weight. Each node contains a character as node's name, while each edge contains information of start node, end node, and the cost. After node and edge is determined, a weighted graph can be generated. After weighted graph is determined, then the graph should be displayed as adjacency list of graph.

Kruskal and Prim algorithm is used to find minimum spanning tree. Each algorithm has different method, so there will be two minimum spanning tree result. Each algorithm method will be explained in flow chart below.

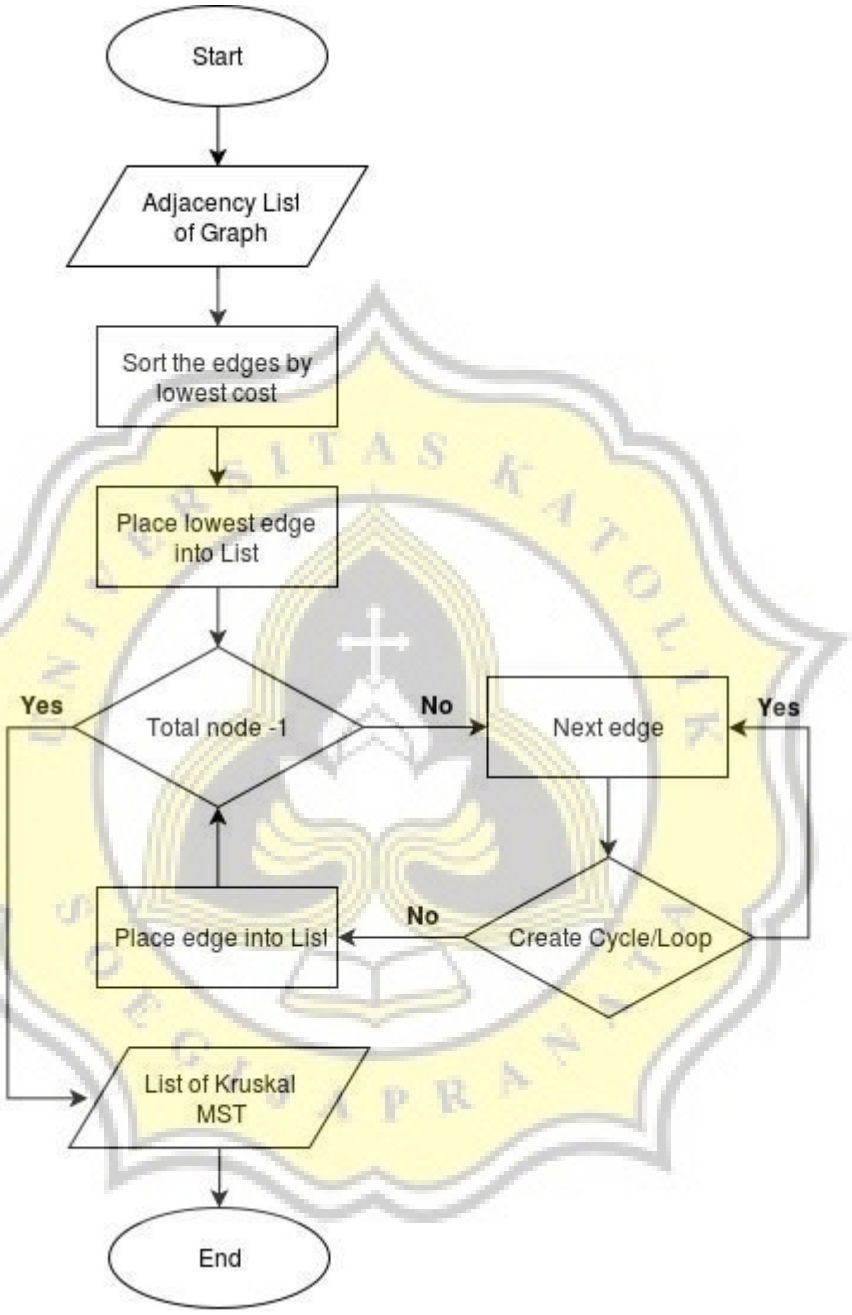

Illustration 4.5: MST Using Kruskal Algorithm Flow Chart

To find minimum spanning tree using Kruskal algorithm, all of edges should be sorted first. Edge with lowest cost will be placed into list. Program should be able to detect if there any cycle in a graph or not. It is also requires a method to sort the edges from the lowest cost.

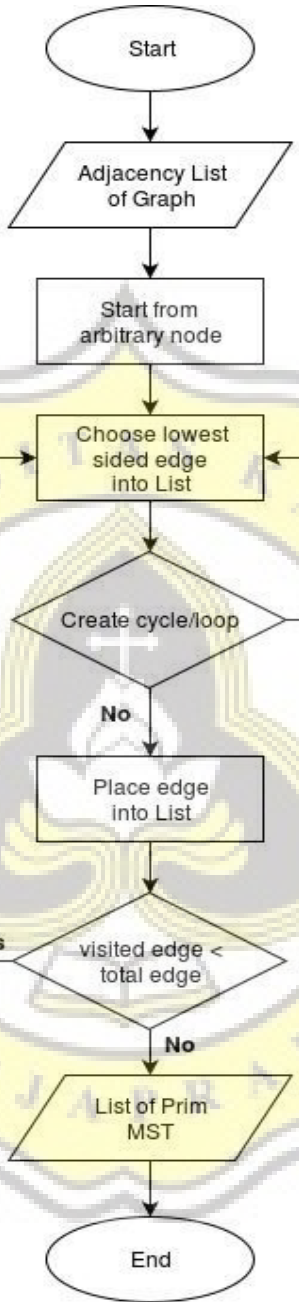

Illustration 4.6: MST Using Prim Algorithm Flow Chart

Prim algorithm has different process compared with Kruskal. The process begin with determining arbitrary node or a node to start. A method to note visited node will be required.

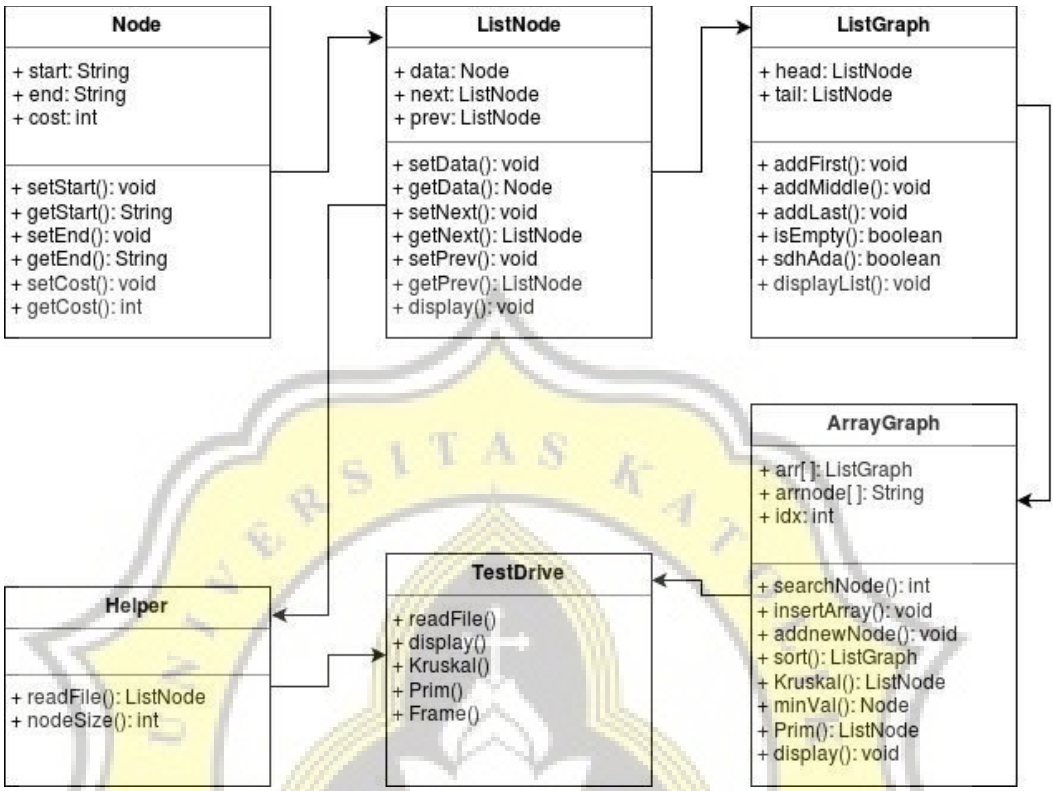

Illustration 4.7: Class Diagram

It is required 6 class to built this project. Class Node is used to determine name of nodes and cost. Class ListNode and ListGraph contain methods to create and add new linked list where edge data will be stored in this list.

Class ArrayGraph contains methods that will be used to generate weighted graph. There is an array that is created to store list of edge. This class also contains method addnewNode() to determine new node, Kruskal and Prim method to find minimum spanning tree. Method sort() is used to sort all of edges from the lowest cost. This method later will be used in finding minimum spanning tree with Kruskal Algorithm. Method minValue() will be used in finding minimum spanning tree with Prim Algorithm. The result will be displayed using display().

TestDrive is the main class. Data will be load in this class. Frame created in this class to show the GUI which will displayed minimum spanning tree result using Kruskal and Prim algorithm.