**Design Decisions:**

**SYSC 3110 Project – Network Routing Simulator**

**Git Repo:** <https://github.com/SYSC3110/project-team-smalltalkers>

Group Members:

Mohammed Omar Rasheed Khan - 100983417

Omar Ibrahim - 100943448

Adnan Hajar - 100983224

Ali Faizan - 100935765

**Design:**

**Our design consists of six classes:**

1. Attribute.java
2. Model.java
3. NetworkUI.java
4. Node.java
5. RandomRoutingAlgorithm.java
6. Strategy.java

We decided to use the MVC pattern and the strategy pattern for our design. The MVC pattern is divided in classes Model.java which is the model, the Node.java class which is the controller and lastly the NetworkUI.java which is the view. The strategy pattern comes into play when we want to implement the algorithms in this milestone it is the RandomRoutingAlgorithm.java, the strategy pattern is used because the algorithm is varied independently from the clients that use it. This also makes it easier for upcoming milestone where would only need to implement 3 remaining algorithms and change the functionality of the GUI.

**Design decisions (Individual classes):**

1. Attribute.java

-This class holds information about the message, the source and destination of each node.

1. Model.java

-This class holds information about the logic of the entire simulator system.

1. NetworkUI.java

-This is the class responsible for running the project. It has the main method. It deals with user input and processing the input for error checking. All events are printed to the console such as adding a node and status of each event with a menu.

1. Node.java

-This class has information about the Router node. It contains an array list of all the neighbours in the router for each router node and is primarily responsible for adding and removing them.

1. RandomRoutingAlgorithm.java

-This class is primarily responsible for the algorithm that the router currently executes. It randomly uses a number of hops and packets to reach from a source to the destination.

1. Strategy.java

* This class is an abstract class that is implemented by all the algorithm subclasses