

IoT Project Report

keep your distance

Mohammadreza Bakhshizadeh Mohajer 913233 - 10647452 | August 2020

# Introduction

This project was created due to the situation created by Corona virus pandemic. It is aimed at helping people keep their distance in enclosed areas such as museums and galleries. The model is based on TinyOS as operating system, Cooja as the simulation environment, Node-red as connection between the nodes and the internet for sending notifications to a mobile phone.

In the simulation phase, if two nodes come to a certain range from another mote an alarm procedure is triggered as follows.

1. The Node ID of the other node in the proximity is saved in an array inside the node.
2. The red LED on the node starts to blink while showing on its own screen and sending an alarm message containing the Node ID of the violating node to Node-red.
3. In Node-red the message is then converted to a message format suitable for mobile phone notification and sent as one using the IFTTT platform.
4. The notification that contains first the ID of the violating node and then the ID of the other node is sent to be viewed on the handset of the phone.

# Implementation

One code has been used for the nodes which starts by booting the devices then turning on the radio interface. After making sure that the radio is on the mote sets a timer of 500ms. As the timer expires a message containing the Node ID of the mote is broadcasted in its proximity. Soon as two motes become close to each other, they receive the broadcast message so they extract the Node ID of the other node and store it in an array as in Figure 1 while toggling on and off the red LED of the mote, Figure 2.

The alarm message is shown on the screens of the violating motes that breached the distance limit which is demonstrated in Figure 3. Using a unique socket for each mote we connect them to Node-red. In Figure 4 the architecture is depicted for 5-motes scenario. In Node-red first the received messages from each TCP socket must be cleaned from the unwanted characters due to the bogus function of printf in Cooja. After trimming the messages, they are displayed in Node-red and then a switch statement classifies them according to which node is the conflicting mote, then we use a rate limiter to control the flow of notifications to the phone. The function after the limiter creates a template for each notification providing the event name and values which are the Node IDs to trigger the notification using the IFTTT platform. All the outputs of the template creator function are collected in a log.txt file in the directory of the program. In the end using a HTTP POST node we post the request to trigger the notification on the IFTTT website. Figure 5 shows the Node-red program. The messages printed in Node-red are in Figure 6, plus a screenshot of the notifications received on the cellular phone handset can be found in Figure 7.

# Figures

A close up of text on a white background

Description automatically generated

Figure : Node ID of the violating mote is being written in the memory array of each node.

A close up of a logo

Description automatically generated

Figure 2: The red LEDs of the close motes are blinking.

![A close up of a device

Description automatically generated]()

Figure 3: The alarm message shown on the node screens (Here in Cooja simulator).

A screenshot of a computer

Description automatically generated

Figure 4: The 5-mote scenario architecture that each mote is connected to Node-red using a unique socket.

A close up of a map

Description automatically generated

Figure 5: Node-red design of the 5-mote program with a unique socket for each mote.

A screenshot of a cell phone

Description automatically generatedA screenshot of text

Description automatically generated

Figure 6 (Left): Messages printed in Node-red. Figure 7 (Right): Notifications on the phone.