Student ID:

tuni.fi:150164676

E-mail:

zannatul.2.ferdous@tuni.fi

Event driven application:

Event-driven programming is a programming paradigm in which the flow of program execution is determined by events .
Event programming is not a type of technology or programming language, but an approach that is implemented during the product development stage
. For software teams, developing an event-driven application inevitably adds complexity to system design and delivery.
In the request-driven case, the customer drives the interactions, dictating which service will process the request. The customer must then wait for a response to the request.

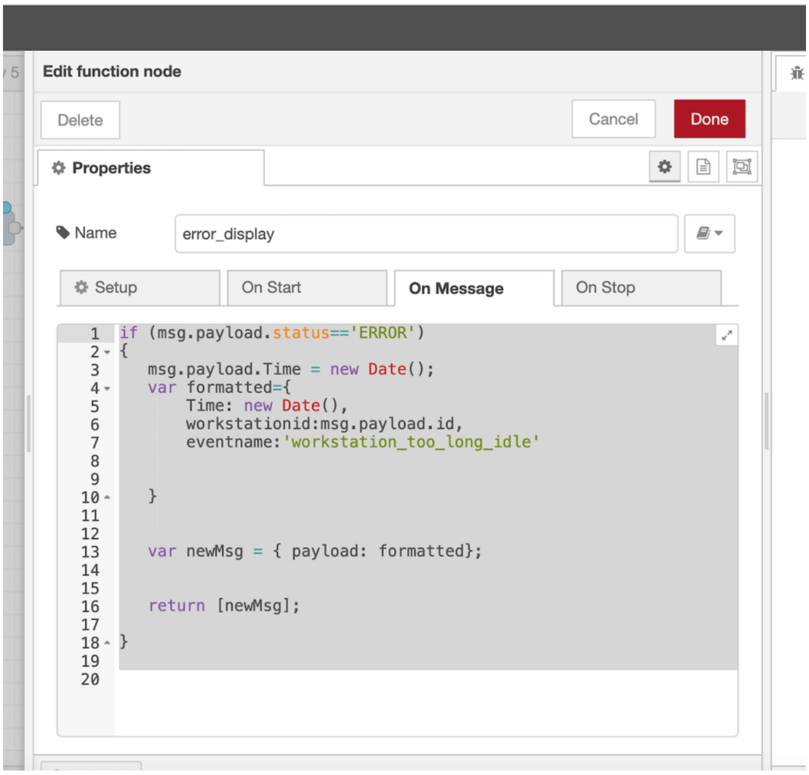
Some examples of Event-driven application are:

- An HTML message has been received (web server)
- A key has been pressed (text editor)
- A loan application has been accepted/rejected (commercial business)
- A new rostering schedule is ready for distribution to all personnel (management system)
- An illegal trade pattern has been detected (fraud detection)
- A car in a computer game hits another car (racing game)
- A robot has reached its destination (real-time warehouse management)

Node Red Snippets:

#Code inside the Function:

When status is ’ERROR’

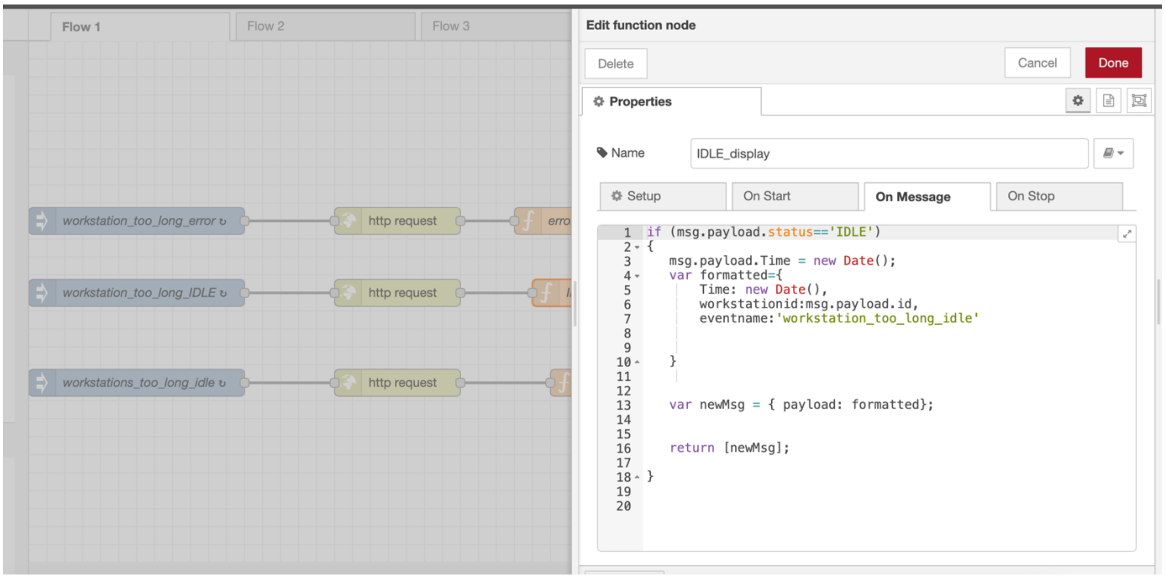


Setup

Name: error_display

```
if (msg.payload.status=='ERROR')
{
    msg.payload.Time = new Date();
    var formatted={
         Time: new Date(),
         workstationid:msg.payload.id,
         eventname:'workstation_too_long_idle'


    }


    var newMsg = { payload: formatted};


    return [newMsg];

}


```

#When status is ’IDLE:

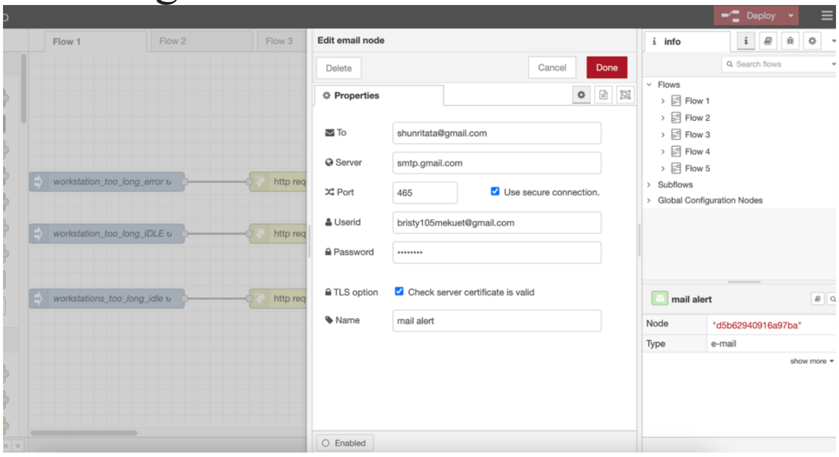


Name: IDLE_display

Setup | On Start | On Message | On Stop

```
if (msg.payload.status=='IDLE')
{
    msg.payload.Time = new Date();
    var formatted={
        Time: new Date(),
        workstationid:msg.payload.id,
        eventname:'workstation_too_long_idle'


    }

    var newMsg = { payload: formatted};

    return [newMsg];
}
```

#sending emails:



To: shunritata@gmail.com

Server: smtp.gmail.com

Port: 465 ☑ Use secure connection.

Userid: bristy105mekuet@gmail.com

Password: ••••••••

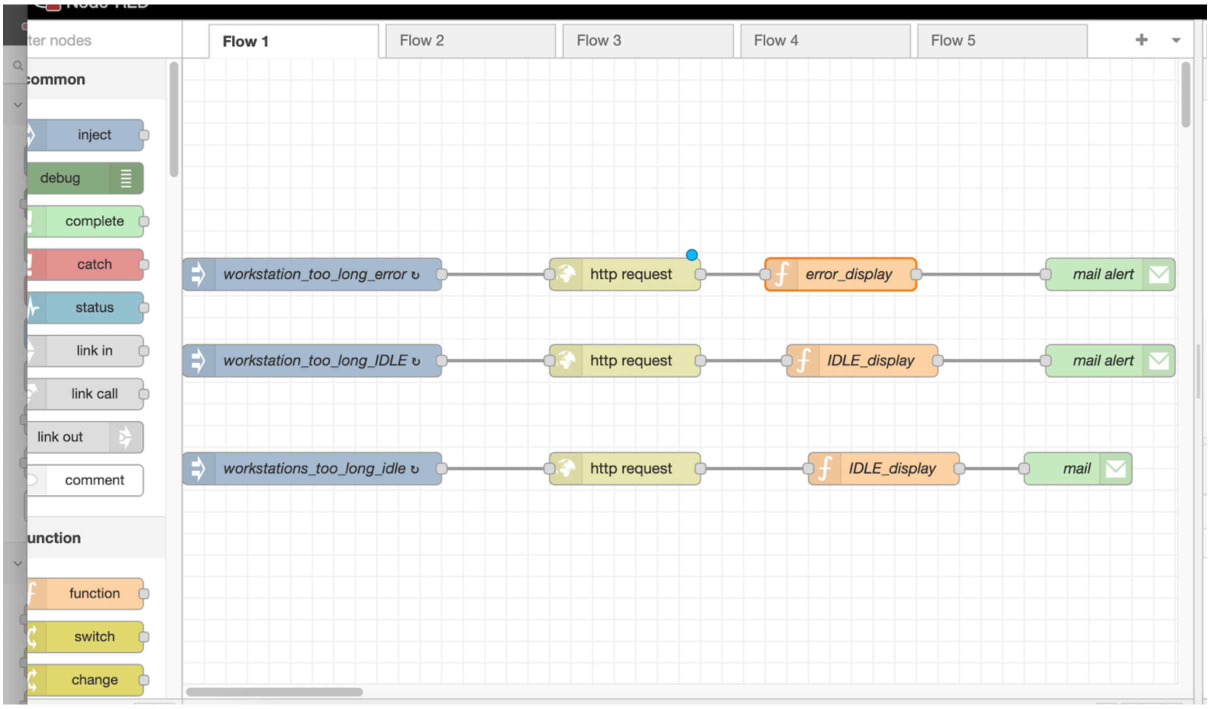
TLS option ☑ Check server certificate is valid

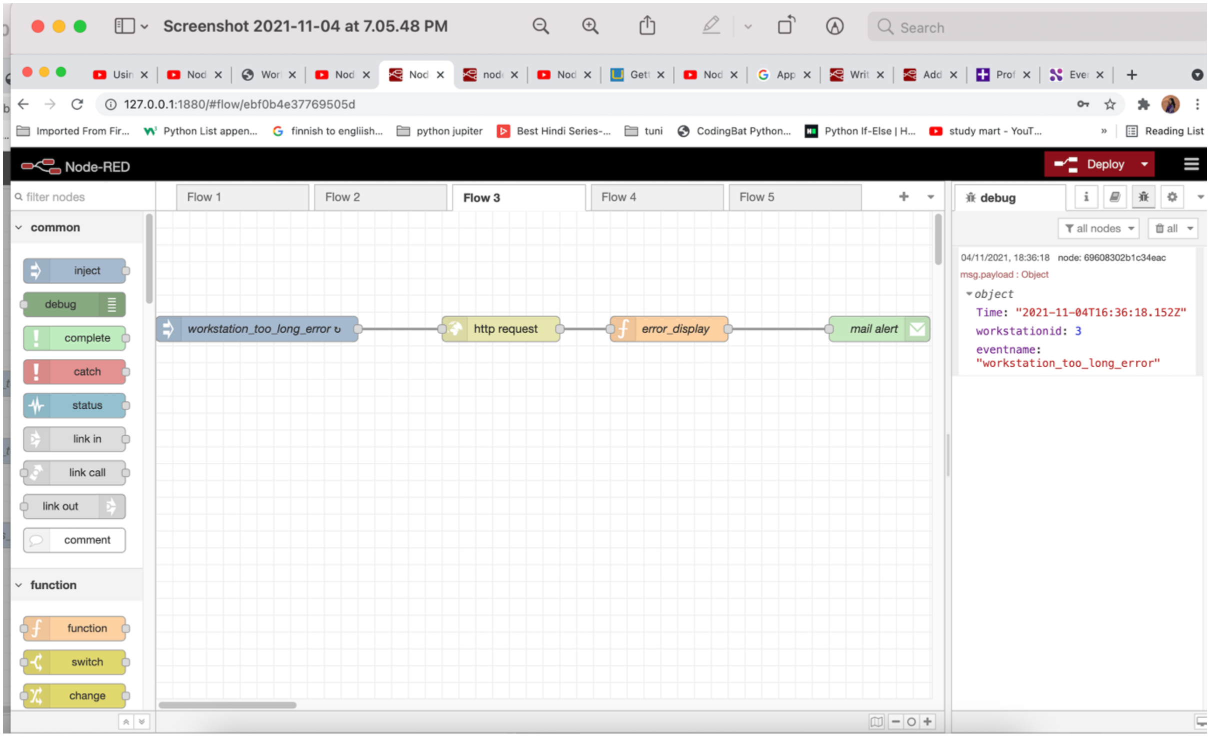
Name: mail alert

mail alert

Node	"d5b62940916a97ba"
Type	e-mail

#2 Workstation, having 3 events:





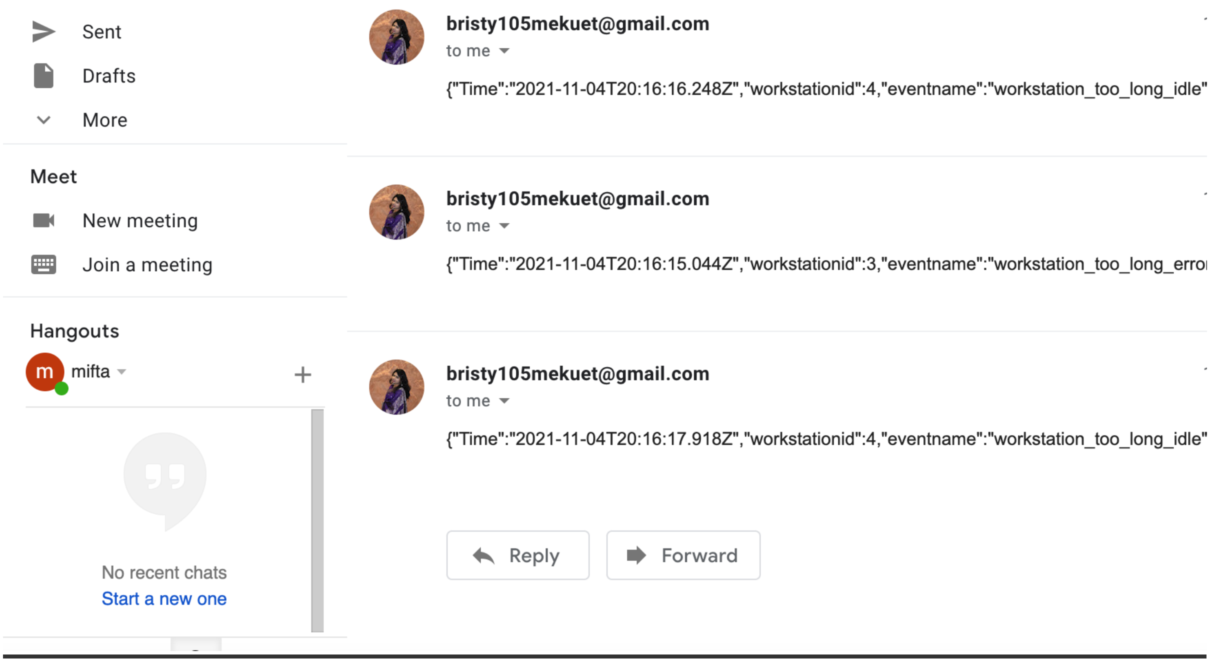
```
04/11/2021, 18:36:18   node: 69608302b1c34eac
msg.payload : Object
▾object
  Time: "2021-11-04T16:36:18.152Z"
  workstationid: 3
  eventname:
  "workstation_too_long_error"
```

RESULTS:

#When workstation_too_long_error:

Condition: When a workstation has been more than 1 minutes in ERROR state

So, here I guess time interval is 2 minutes.



bristy105mekuet@gmail.com
to me

{"Time":"2021-11-04T20:16:16.248Z","workstationid":4,"eventname":"workstation_too_long_idle"

bristy105mekuet@gmail.com
to me

{"Time":"2021-11-04T20:16:15.044Z","workstationid":3,"eventname":"workstation_too_long_erro

bristy105mekuet@gmail.com
to me

{"Time":"2021-11-04T20:16:17.918Z","workstationid":4,"eventname":"workstation_too_long_idle"

#When workstations_too_long_idle:

Condition: When two workstations have been in IDLE state for more than 2 minutes

So, time interval is taken 3minutes here.

#Analyze the Results:

For the event 1: every 4 minutes interval (as the condition is, greater than 3 minutes) in timestrap, after the http request, if function finds out the payload is in idle state, only then it can send the email notification.

For the event 2: every 2 minutes interval (as the condition is, greater than 1 minutes) in timestrap, after the http request, if function finds out the payload is in 'error' state, only then it can send the email notification.

For the event 3: every 3 minutes interval (as the condition is, greater than 2 minutes) in timestrap, after the http request, if function finds out the payload is in idle state, only then it can send the email notification.

#Improve the solution:

For improving this solution, 1 function node and 1 email node can be called instead of 3 nodes for function and email using here. The code behind every event can be written in a single function and using 'if' condition, it can check which status it should proceed.

#Conclusion:

It has been successfully tested. All the states are manipulated using API successfully.