Script principal

```python
#-----------------------------------------------------
# Main script
#-----------------------------------------------------

if __name__ == "__main__":  # Runs only if called as a script but not if imported
    print("Hello and welcome to EMSY")
    # Calcul du Point de rosee
    COEFF1 = 17.62              #  BETA
    COEFF2 = 243.12             #  GAMA
    t, rh = read_sensor()       # Appel de la focntion read_sensor()
    rh_value = rh.percent_rh    # Recupere la valeur de l'humidite relative du capteur en pourcentage .
    t_value = str(t)            # Convertit la valeur de temperature en chaine de caracteres.
    t_value =t_value.split()    # Divise la chaine de caracteres qui contient la valeur de temperature
    t_value =float(t_value[0])  # Convertit la premiere partie de la chaine divisee en un nombre flottant.
    # Calcul partie du numerateur
    part1 = COEFF2 * (math.log(rh_value / 100) + COEFF1 * t_value / (COEFF2 + t_value))
    # Calcul partie du denominateur
    part2 = COEFF1 - (math.log(rh_value / 100) + COEFF1 * t_value / (COEFF2 + t_value))
    # Divise les deux parties
    Prosee = part1 / part2
    print("Temperature:", t_value, "°C")              # Affiche la temperature
    print("Humidite relative:", rh_value, "%")        #Affiche la valeur de l'Humidite relative
    print("Point de Rosee:", round(Prosee, 2), "°C")  # Affiche Point de Rosee et l'arrondie 2 decimales


    system_datetime = datetime.datetime.now()               # Obtient l'heure et la date actuelles du systeme
    current_date = system_datetime.strftime("%d.%m.%Y")     # Prend l'heure actuelle au format jour/mois/annee
    current_time = system_datetime.strftime("%H:%M")        # Prend l'heure actuelle au format heure/minute
    print("Heure:", current_time)                           # Affiche l'heure
    print("Date", current_date)                             # Affiche la date
    #Engeristrer les donnees dans un fichier csv qui se trouve dans le chemin /home/debian/TempLog.csv
    save_to_csv('/home/debian/TempLog.csv', current_date, current_time, t_value,round(rh_value, 2), round(Prosee, 2))


    # Controler si la temperature deppase la limite et envoyer le mail
    temperature_limite = 28.0
    if t_value > temperature_limite:                                # Si la temperature depasse 28°C
        print("Temperature limite depasse! ENVOI D'ALARME")
        subject = "Alarme de Temperature"
        message = f"La temperature a depasse la limite de {temperature_limite}°C. La temperature actuelle est de {t_value}°C."
        send_email("luis.pucuji@etml-es.ch", subject, message)      #Envoi du mail avec l'alarme
    else:
        print("Temperature dans la plage de securite :)")           # Si la temperature <= à 28°C afficher le texte...
```

Calcul du *POINT DE ROSEE*

On a utilisé la fonction read_sensor() proposée.

```
def read_sensor():
    try:
        t, rh = sht40.single_shot_measurement()
        # Watch out! t and rh are variable that contain not only the values but also the units.
        # You can print the values with the units (print(t)) or you can also recover only the value
        # by specifying which one: t.degrees_celsius or rh.percent_rh
    except Exception as ex:
        print("Error while recovering sensor values:", ex)
    else:
        return t, rh
    return 0 # In case something went wrong
```

Script :

```
if __name__ == "__main__":  # Runs only if called as a script but not if imported
    print("Hello and welcome to EMSY")
    # Calcul du Point de rosee
    COEFF1 = 17.62               #  BETA
    COEFF2 = 243.12              #  GAMA
    t, rh = read_sensor()        # Appel de la focntion read_sensor()
    rh_value = rh.percent_rh     # Recupere la valeur de l'humidite relative du capteur en pourcentage .
    t_value = str(t)             # Convertit la valeur de temperature en chaine de caracteres.
    t_value =t_value.split()     # Divise la chaine de caracteres qui contient la valeur de temperature
    t_value =float(t_value[0])   # Convertit la premiere partie de la chaine divisee en un nombre flottant.
    # Calcul partie du numerateur
    part1 = COEFF2 * (math.log(rh_value / 100) + COEFF1 * t_value / (COEFF2 + t_value))
    # Calcul partie du denominateur
    part2 = COEFF1 - (math.log(rh_value / 100) + COEFF1 * t_value / (COEFF2 + t_value))
    # Divise les deux parties
    Prosee = part1 / part2
    print("Temperature:", t_value, "°C")              # Affiche la temperature
    print("Humidite relative:", rh_value, "%")        #Affiche la valeur de l'Humidite relative
    print("Point de Rosee:", round(Prosee, 2), "°C")  # Affiche Point de Rosee et l'arrondie 2 decimales
```

- Il y a deux variables : **COEFF1** (β) et **COEFF2**(γ) avec des valeurs constantes que nous utilisons dans la formule.
- On appel la fonction **read_sensor()** pour obtenir les donnés du capteur, c'est-à-dire, la température et l'humidité relative pour assigner ces valeurs aux variables t et rh.
- On récupère la valeur de l'humidité relative du capteur et on le met dans la variable **rh_value.**
- Pour **t_value :** Avec ces 3 lignes de code :

  ```
  t_value = str(t)
  t_value =t_value.split()
  t_value =float(t_value[0])
  ```

  On convertit la valeur de la température en chaine de caractère pour ensuite extraire seulement la partie numérique de la valeur de la température grâce à la méthode ***split,*** et finalement on le convertit en un nombre flottant
- Pour le calcul on a partagé la grosse formule en deux parties pour que ça soit plus compréhensible. La variable ***part1*** représente la partie du numérateur et la ***partie2*** représente la partie du dénominateur. Ensuite dans la variable ***Prosee*** on fait le calcul final, donc la division de **part1/part2.**
- Finalement on imprime les valeurs de la température, humidité relative et point de rosee dans le terminal pour vérifier que tout s'est bien passé.

Enregistrement des données dans le fichier .csv

On a utilisé la fonction save_to_csv() proposée.

```
def save_to_csv(filename, date, time, temperature, humidity, dew_point):
    with open(filename,'a') as file:
        writer = csv.writer(file)
        writer.writerow([date, time, temperature, humidity, dew_point])
```

Dans le script nous avons ajouté quelques lignes de code pour obtenir la date et l'heure pour ensuite les enregistrer dans un fichier .csv

```
system_datetime = datetime.datetime.now()               # Obtient l'heure et la date actuelles du systeme
current_date = system_datetime.strftime("%d.%m.%Y")     # Prend l'heure actuelle au format jour/mois/annee
current_time = system_datetime.strftime("%H:%M")        # Prend l'heure actuelle au format heure/minute
print("Heure:", current_time)                           # Affiche l'heure
print("Date", current_date)                             # Affiche la date
#Engeristrer les donnees dans un fichier csv qui se trouve dans le chemin /home/debian/TempLog.csv
save_to_csv('/home/debian/TempLog.csv', current_date, current_time, t_value,round(rh_value, 2), round(Prosee, 2))
```

- La variable **system_datetime** nous permet d'obtenir l'heure et la date actuelles du système.
  nomVariable = module.classe.méthode()
  Dans le module **datetime** il y a une classe nommé **datetime** aussi, dans la classe **datetime** il y a quelques méthodes pour gérer les dates et les heures, **now()** est une méthode qui renvoie l'heure et la date actuelles du système.
- Ces deux lignes de code :
  ```
  current_date = system_datetime.strftime("%d.%m.%Y")
  current_time = system_datetime.strftime("%H:%M")
  ```
  Nous permettent de prend l'heure et la date pour les mettre au format **jour/mois/année** et **heure/minute**
- Ensuite on imprime les valeurs de **current_date** et **current_time** dans le terminal.
- Cette ligne de code :
  ```
  save_to_csv('/home/debian/TempLog.csv', current_date, current_time, t_value,round(rh_value, 2), round(Prosee, 2))
  ```
  Nous permet d'enregistrer l'heure, la date, l'humidité relative et le point de rosée dans le fichier .csv placé dans le chemin : /home/debian/TempLog.csv, de cette façon lors de chaque exécution du code toutes les informations seront enregistrées dans ce fichier .csv

Envoi de l'email

On a utilisé la fonction send_email() proposée.

```python
def send_email(receiver, subject, message):
    with smtplib.SMTP_SSL('smtp.gmail.com', 465) as server:
        server.login("ETML.ES.EMSY@gmail.com","cely neve caly akjz")
        sender = "ETML.ES.EMSY@gmail.com"

        headers = {
            'Content-Type': 'text/html; charset=utf-8',
            'Content-Disposition': 'inline',
            'Content-Transfer-Encoding': '8bit',
            'From': sender,
            'To':receiver,
            'Date': datetime.datetime.now().strftime('%a, %d %b %Y  %H:%M:%S %Z'),
            'X-Mailer': 'python',
            'Subject': subject
        }
        # create the message
        msg = ''
        for key, value in headers.items():
            msg += "%s: %s\n" % (key, value)

        # add contents
        msg += "\n%s\n" % (message)

        try:
            server.sendmail(headers['From'], headers['To'], msg.encode("utf8"))
            server.quit()
            print("Email sent successfully!")
        except Exception as ex:
            print("Something went wrong.", ex)
```

Et nous avons rajouté quelques lignes de code pour gérer l'envoi de l'email

```python
# Controler si la temperature deppase la limite et envoyer le mail
temperature_limite = 28.0
if t_value > temperature_limite:                                # Si la temperature depasse 28°C
    print("Temperature limite depasse! ENVOI D'ALARME")
    subject = "Alarme de Temperature"
    message = f"La temperature a depasse la limite de {temperature_limite}°C. La temperature actuelle est de {t_value}°C."
    send_email("luis.pucuji@etml-es.ch", subject, message)      #Envoi du mail avec l'alarme
else:
    print("Temperature dans la plage de securite :)")           # Si la temperature <= à 28°C afficher le texte...
```

Si la température est supérieure à 28°C (selon cahier de charge) un email d'alerte doit être envoyé à l'utilisateur. Dans le cas contraire si nous sommes dans la plage de sécurité, c'est-à-dire, inférieur ou égal à 28°C il n'y a pas d'email d'alerte.