Setup the following classes

Person:

-attributes (full_name, money, sleepmood, healthRate)

-methods (sleep, eat,buy)

Employee(is a Person class):

-attributes (id, email, workmood, salary,is_manager)
-methods (work,sendEmail)

Office:

-attributes (name,employees)
-methods (get_all_employees, get_employee, fire,hire)

Implement Employee methods

sleep(hours**):** Method in Person class(7→happy, <7 →tired, >7 →lazy)

eat(meals**):** Method in Person class(3 meals →100 health rate, 2meals→75 health rate , 1 meal→50 healthrate)

buy(items**):** Method in Person class(1 Item→decreesMoney 10LE)

sendEmail(to, suject,bodyreceiver_name): on call it createsa file represent theemail.

Implement Employee methods cont…

work(hours): Method in Employee class(8→happy, >8→tired, > 8 →lazy)

- Salary: Property must be 1000 or more
- Health rate: Property must be between 0 and 100
- Email: Property must be verified with regex expression

Implement Office methods

get_all_employees(): Method in Office class get all current employees.

get_employee(empId): Method in Office class get employee data of given employee id,and if he is a manager display all info except salary.

hire(Employee): Method in Office class hires the given employee.

fire(empId): Method in Office class fires the given employeeid.

DB table Employee should maintain employee objects and office retrievals

Let the program be a user command interface.
Print a menu with the functionalities allowed.
For example:
For adding new employee enter “add”
If manager press “mngr”
if normal employee press “nrml”
Enter your data:
>> Name:
>> age:
The final menu option is “q” to quit the application.