import sqlite3

class Database:

def \_\_init\_\_(self):

self.con = sqlite3.connect('todo.db')

self.cursor = self.con.cursor()

self.create\_task\_table()

'''CREATE the Tasks TABLE'''

def create\_task\_table(self):

self.cursor.execute(

"CREATE TABLE IF NOT EXISTS tasks(id integer PRIMARY KEY AUTOINCREMENT, task varcahr(50) NOT NULL, due\_date varchar(50), completed BOOLEAN NOT NULL CHECK (completed IN (0, 1)))")

'''CREATE A Task'''

def create\_task(self, task, due\_date=None):

self.cursor.execute("INSERT INTO tasks(task, due\_date, completed) VALUES(?, ?, ?)", (task, due\_date, 0))

self.con.commit()

# Getting the last entered item to add in the list

created\_task = self.cursor.execute("SELECT id, task, due\_date FROM tasks WHERE task = ? and completed = 0",

(task,)).fetchall()

return created\_task[-1]

'''READ / GET the tasks'''

def get\_tasks(self):

# Getting all complete and incomplete tasks

complete\_tasks = self.cursor.execute("SELECT id, task, due\_date FROM tasks WHERE completed = 1").fetchall()

incomplete\_tasks = self.cursor.execute("SELECT id, task, due\_date FROM tasks WHERE completed = 0").fetchall()

return incomplete\_tasks, complete\_tasks

'''UPDATING the tasks status'''

def mark\_task\_as\_complete(self, taskid):

self.cursor.execute("UPDATE tasks SET completed=1 WHERE id=?", (taskid,))

self.con.commit()

def mark\_task\_as\_incomplete(self, taskid):

self.cursor.execute("UPDATE tasks SET completed=0 WHERE id=?", (taskid,))

self.con.commit()

# returning the task text

task\_text = self.cursor.execute("SELECT task FROM tasks WHERE id=?", (taskid,)).fetchall()

return task\_text[0][0]

'''Deleting the task'''

def delete\_task(self, taskid):

self.cursor.execute("DELETE FROM tasks WHERE id=?", (taskid,))

self.con.commit()

'''Closing the connection '''

def close\_db\_connection(self):

self.con.close()