Project Name: Employee Management System

Programming Part:

```
import customtkinter
from tkinter import *
from tkinter import ttk
import tkinter as tk
from tkinter import messagebox
import database

app = customtkinter.CTk()
app.title('Employee Management System')
app.geometry('900x420')
app.config(bg='#161C25')
app.resizable(False,False)

font1 = ('Arial',20,'bold')
font2 = ('Arial',12,'bold')

def add_to_treeview():
    employees= database.fetch_employees()
    tree.delete(*tree.get_children())
    for employee in employees:
        tree.insert('', END, values=employee)


def clear(*clicked):
    if clicked:
        tree.selection_remove(tree.focus())
        tree.focus()
    id_entry.delete(0,END)
    name_entry.delete(0,END)
    role_entry.delete(0,END)
    variable1.set('')
    status_entry.delete(0,END)

def display_data(event):
    selected_item = tree.focus()
    if selected_item:
        row = tree.item(selected_item)['values']
        clear()
        id_entry.insert(0,row[0])
        name_entry.insert(0,row[1])
        role_entry.insert(0,row[2])
        variable1.set(row[3])
        status_entry.insert(0,row[4])
    else:
```

```
        pass

def delete():
    selected_item = tree.focus()
    if not selected_item:
        messagebox.showerror('Error', 'Choose an employee to delete.')
    else:
        id = id_entry.get()
        database.delete_employees(id)
        add_to_treeview()
        clear()
        messagebox.showinfo('Success', 'Data has been deleted.')

def update():
    selected_item = tree.focus()
    if not selected_item:
        messagebox.showerror('Error', 'Choose an employee to update.')
    else:
        id = id_entry.get()
        name = name_entry.get()
        role = role_entry.get()
        gender = variable1.get()
        status = status_entry.get()
        database.update_employee(name, role, gender, status, id)
        add_to_treeview()
        clear()
        messagebox.showinfo('Success', 'Data has been updated.')

def insert():
    id = id_entry.get()
    name = name_entry.get()
    role = role_entry.get()
    gender = variable1.get()
    status = status_entry.get()
    if not (id and name and role and gender and status):
        messagebox.showerror('Error', 'Enter all fields.')
    elif database.id_exists(id):
        messagebox.showerror('Error', 'ID already exists.')
    else:
        database.insert_employee(id,name,role,gender,status)
        add_to_treeview()
        messagebox.showinfo('Success', 'Data has been inserted.')

#------------------------------------------------------------------------
------------------------------------------------------------------------
------------------------------------------------------------------------
---------------------------------------
```

```
id_lable = customtkinter.CTkLabel(app, font=font1, text='ID:',
text_color='#fff', bg_color='#161C25')
id_lable.place(x=20,y=20)

id_entry = customtkinter.CTkEntry(app, font=font1,text_color='#000',
fg_color='#fff', border_color='#0C9295', border_width=2, width=180)
id_entry.place(x=100,y=20)

name_lable = customtkinter.CTkLabel(app, font=font1, text='Name:',
text_color='#fff', bg_color='#161C25')
name_lable.place(x=20,y=80)

name_entry = customtkinter.CTkEntry(app, font=font1,text_color='#000',
fg_color='#fff', border_color='#0C9295', border_width=2, width=180)
name_entry.place(x=100,y=80)

role_lable = customtkinter.CTkLabel(app, font=font1, text='Role:',
text_color='#fff', bg_color='#161C25')
role_lable.place(x=20,y=140)

role_entry = customtkinter.CTkEntry(app, font=font1,text_color='#000',
fg_color='#fff', border_color='#0C9295', border_width=2, width=180)
role_entry.place(x=100,y=140)

gender_lable = customtkinter.CTkLabel(app, font=font1, text='Gender:',
text_color='#fff', bg_color='#161C25')
gender_lable.place(x=20,y=200)

options = ['Male','Female']
variable1 = StringVar()

gender_options = customtkinter.CTkComboBox (app, font=font1,
text_color='#000', fg_color='#fff', dropdown_hover_color='#0C9295',
button_color='#0C9295', border_color='#0C9295', width=180, variable=variable1,
values=options, state='readonly')
#gender_options.set('Male')
gender_options.place(x=100,y=200)

status_lable = customtkinter.CTkLabel(app, font=font1, text='Status:',
text_color='#fff', bg_color='#161C25')
status_lable.place(x=20,y=260)

status_entry = customtkinter.CTkEntry(app, font=font1,text_color='#000',
fg_color='#fff', border_color='#0C9295', border_width=2, width=180)
status_entry.place(x=100,y=260)

#-------------------------------------------------------------------------
-------------------------------------------------------------------------
```

```
-------------------------------------------------------------------------------
----------------------------------------

add_button = customtkinter.CTkButton(app, command=insert, font=font1,
text_color='#fff', text= 'Add Employee', fg_color='#05A312',
hover_color='#00850B', bg_color='#161C25', cursor= 'hand2', corner_radius=15,
width=260)
add_button.place(x=20, y=310)

clear_button = customtkinter.CTkButton(app, command=lambda:clear(True),
font=font1, text_color='#fff', text= 'New Employee', fg_color='#161C25',
hover_color='#FF5002', bg_color='#161C25', border_color='#F15704',
border_width=2, cursor= 'hand2', corner_radius=15, width=260)
clear_button.place(x=20, y=360)

update_button = customtkinter.CTkButton(app, command=update, font=font1,
text_color='#fff', text= 'Update Employee', fg_color='#161C25',
hover_color='#FF5002', bg_color='#161C25', border_color='#F15704',
border_width=2, cursor= 'hand2', corner_radius=15, width=260)
update_button.place(x=300, y=360)

delete_button = customtkinter.CTkButton(app, command=delete, font=font1,
text_color='#fff', text= 'Delete Employee', fg_color='#161C25',
hover_color='#FF5002', bg_color='#161C25', border_color='#F15704',
border_width=2, cursor= 'hand2', corner_radius=15, width=260)
delete_button.place(x=580, y=360)


style = ttk.Style(app)

style.theme_use('clam')
style.configure('Treeview', font=font2, foreground='#fff', background='#000',
fieldbackground='#313837')
style.map('Treeview',background=[('selected', '#1A8F2D')])

tree = ttk.Treeview(app,height=15)

tree['columns'] = ('ID', 'Name', 'Role', 'Gender', 'Status')

tree.column('#0', width=0, stretch=tk.NO)
tree.column('ID', anchor=tk.CENTER, width=120)
tree.column('Name', anchor=tk.CENTER, width=120)
tree.column('Role', anchor=tk.CENTER, width=120)
tree.column('Gender', anchor=tk.CENTER, width=100)
tree.column('Status', anchor=tk.CENTER, width=120)

tree.heading('ID', text='ID')
tree.heading('Name', text='Name')
tree.heading('Role', text='Role')
```

```
tree.heading('Gender', text='Gender')
tree.heading('Status', text='Status')

tree.place(x=300,y=20)

tree.bind('<ButtonRelease>', display_data)

add_to_treeview()

app.mainloop()
```

Database Part:

```
import sqlite3

def create_table():
    conn = sqlite3.connect('Employees.db')
    cursor = conn.cursor()

    cursor.execute('''
        CREATE TABLE IF NOT EXISTS Employees(
            id TEXT PRIMARY KEY,
            name TEXT,
            role TEXT,
            gender TEXT,
            status TEXT)''')
    conn.commit()
    conn.close()


def fetch_employees():
    conn = sqlite3.connect('Employees.db')
    cursor = conn.cursor()
    cursor.execute('SELECT * FROM Employees')
    employees = cursor.fetchall()
    conn.close()
    return employees

def insert_employee(id, name, role, gender, status):
    conn = sqlite3.connect('Employees.db')
    cursor = conn.cursor()
    cursor.execute('INSERT INTO Employees (id, name, role, gender, status)
VALUES (?, ?, ?, ?, ?)',
                   (id, name, role, gender, status))
    conn.commit()
```

```
    conn.close()

def delete_employees(id):
    conn = sqlite3.connect('Employees.db')
    cursor = conn.cursor()
    cursor.execute('DELETE FROM Employees WHERE id = ?', (id,))
    conn.commit()
    conn.close()

def update_employee(new_name, new_role, new_gender, new_status, id):
    conn = sqlite3.connect('Employees.db')
    cursor = conn.cursor()
    cursor.execute("UPDATE Employees SET name = ?, role = ?, gender = ?,
status = ? WHERE id = ?",
                   (new_name, new_role, new_gender, new_status, id))
    conn.commit()
    conn.close()

def id_exists(id):
    conn = sqlite3.connect('Employees.db')
    cursor = conn.cursor()
    cursor.execute('SELECT COUNT(*) FROM Employees WHERE id = ?', (id,))
    result = cursor.fetchone()
    conn.close()
    return result[0] > 0

create_table()
```