USTH
VIETNAM FRANCE UNIVERSITY

TRƯỜNG ĐẠI HỌC
KHOA HỌC VÀ CÔNG NGHỆ
HÀ NỘI

Advanced Programming with Python Report

Contents

1. Introduction

a. Member:

- Group 2:

- + Nguyễn Huy Hùng BI10-071
- + Trịnh Quốc Hiếu BI10-060
- + Nguyễn Hoàng Sơn BI10-155
- + Phạm Đức Thắng BI10-159
- + Đỗ Thành Đạt BI10-026

b. Project content

- Topic 9: Public Governmental Service Information Management System (e.g. birth/death declaration, marriage. . .)

c. Purpose

- Due to covid-19 situation, social distance and digital transformation in the Industrial Revolution 4.0, we need a project to manage Public Governmental Service Information.

- The normal way:

+ We have to go to the administrative Committee to fill our information by writing on paper.

+ It takes us time to wait our turn, maybe money. It needs many people to manage each work.

+ In bad situation, we can lost our paper of information, some bad people can fake our information, wrong information.…

=> That's why we need a software to do the work by the smartest way.

- The new way:

+ We can save time and money for the future

+ The new software does not need many people to do the work.

+ We can do it at home.

+ It is easy to find, manage our own information.

- Idea: The project is based on law of Vietnam Website, dichvucong.gov.vn…

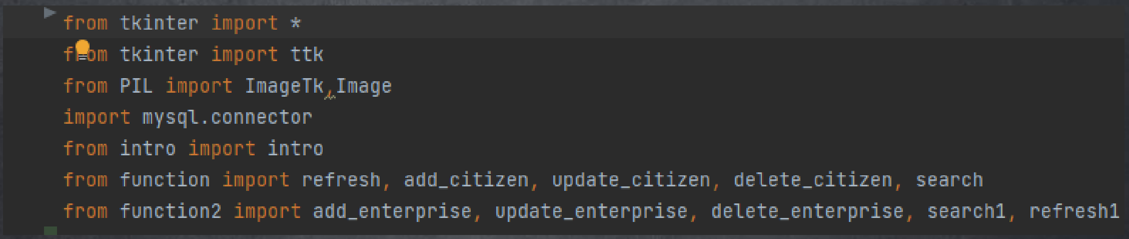
2. Structure

- In our project, we use tkinter to make the GUI, mysql workbench and xampp to store data in database, mysql.connector to connect project’s data witch database

a. Modules and Packages

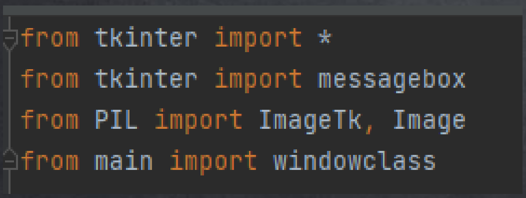
- We use package tkinter with five modules are five functions and mysql.connector

- In main.py



```
from tkinter import *
from tkinter import ttk
from PIL import ImageTk,Image
import mysql.connector
from intro import intro
from function import refresh, add_citizen, update_citizen, delete_citizen, search
from function2 import add_enterprise, update_enterprise, delete_enterprise, search1, refresh1
```

- In login.py



```
from tkinter import *
from tkinter import messagebox
from PIL import ImageTk, Image
from main import windowclass
```

b. Class and Method

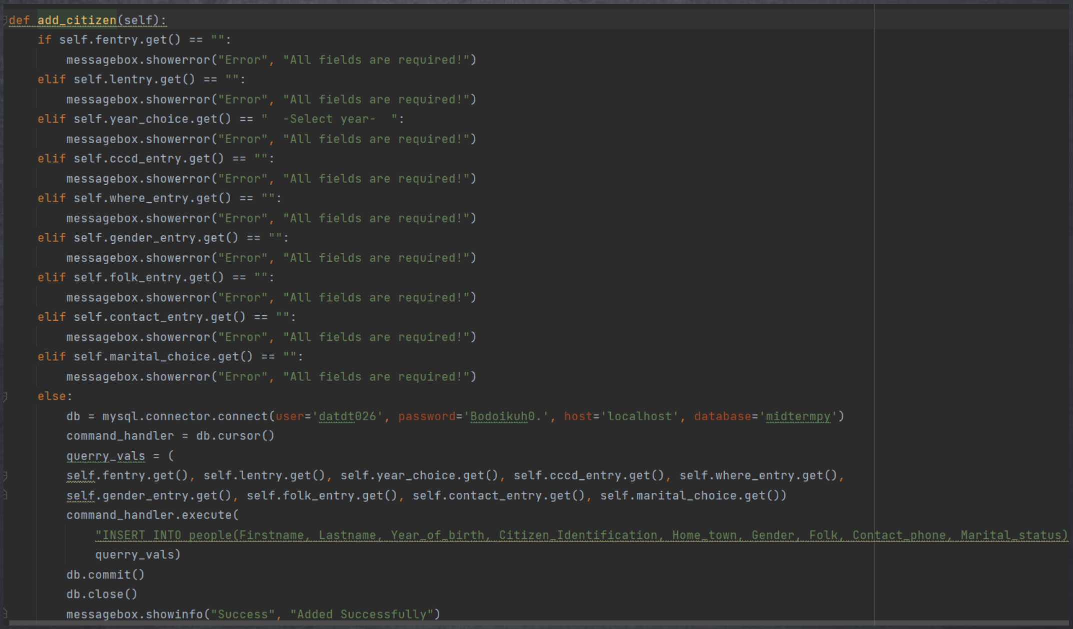
- We have 2 classes: windowclass() and login().

- Class windowclass() has GUI with methods: “add, update, delete, refresh, search, show_list”. Class login() has GUI of login window and login method.

- We have 2 data types are people’s basic data and people’s employment.

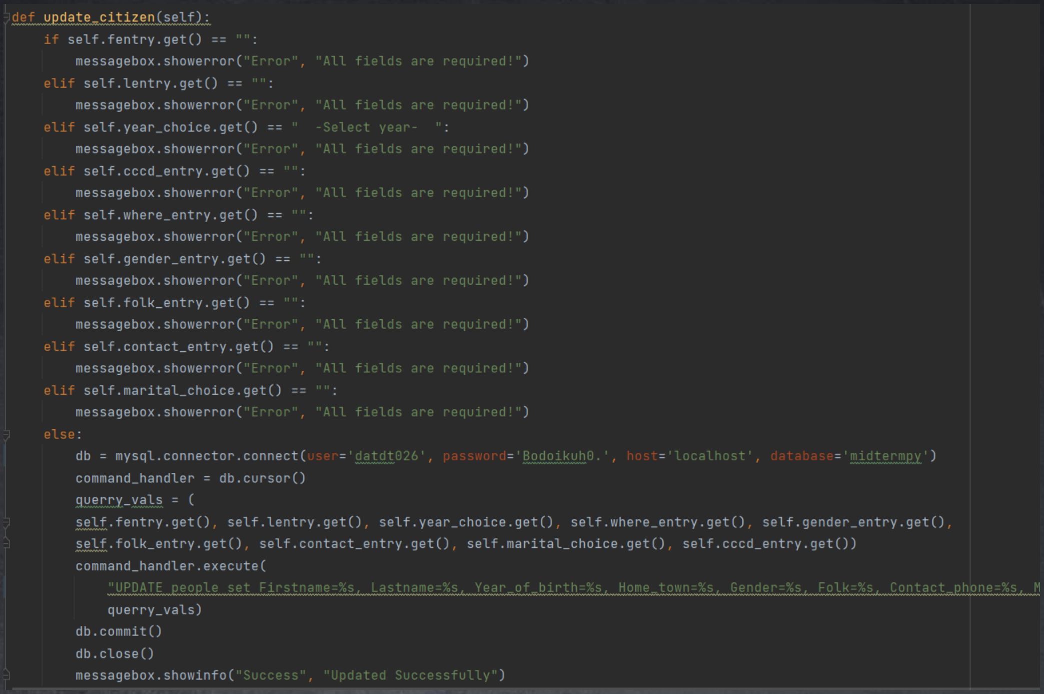
- We have these functions:

+ Add: we fill all the fields Name, Year of birth, Citizen Identification, Company’s Name, Business Code, Address…, create new data of one person



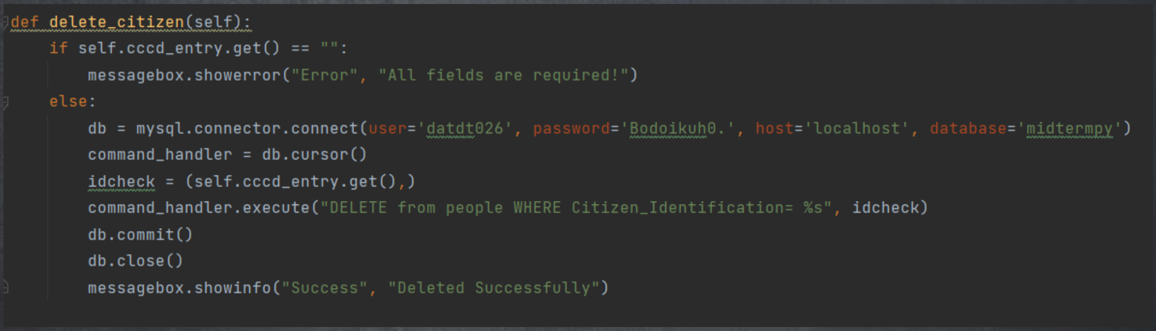
```python
def add_citizen(self):
    if self.fentry.get() == "":
        messagebox.showerror("Error", "All fields are required!")
    elif self.lentry.get() == "":
        messagebox.showerror("Error", "All fields are required!")
    elif self.year_choice.get() == "  -Select year-  ":
        messagebox.showerror("Error", "All fields are required!")
    elif self.cccd_entry.get() == "":
        messagebox.showerror("Error", "All fields are required!")
    elif self.where_entry.get() == "":
        messagebox.showerror("Error", "All fields are required!")
    elif self.gender_entry.get() == "":
        messagebox.showerror("Error", "All fields are required!")
    elif self.folk_entry.get() == "":
        messagebox.showerror("Error", "All fields are required!")
    elif self.contact_entry.get() == "":
        messagebox.showerror("Error", "All fields are required!")
    elif self.marital_choice.get() == "":
        messagebox.showerror("Error", "All fields are required!")
    else:
        db = mysql.connector.connect(user='datdt026', password='Bodoikuh0.', host='localhost', database='midtermpy')
        command_handler = db.cursor()
        querry_vals = (
        self.fentry.get(), self.lentry.get(), self.year_choice.get(), self.cccd_entry.get(), self.where_entry.get(),
        self.gender_entry.get(), self.folk_entry.get(), self.contact_entry.get(), self.marital_choice.get())
        command_handler.execute(
            "INSERT INTO people(Firstname, Lastname, Year_of_birth, Citizen_Identification, Home_town, Gender, Folk, Contact_phone, Marital_status)
            querry_vals)
        db.commit()
        db.close()
        messagebox.showinfo("Success", "Added Successfully")
```

+ Update: we can change Name, Year of birth, Citizen Identification, Company’s Name, Business Code, Address…, CI and Business Code can’t be changed



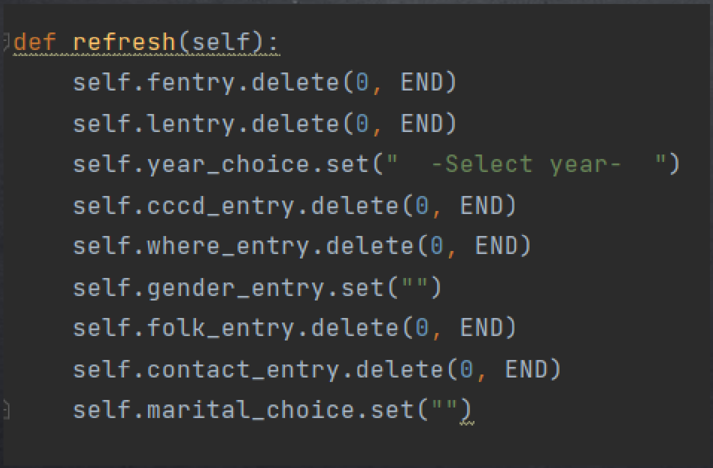
```python
def update_citizen(self):
    if self.fentry.get() == "":
        messagebox.showerror("Error", "All fields are required!")
    elif self.lentry.get() == "":
        messagebox.showerror("Error", "All fields are required!")
    elif self.year_choice.get() == "  -Select year-  ":
        messagebox.showerror("Error", "All fields are required!")
    elif self.cccd_entry.get() == "":
        messagebox.showerror("Error", "All fields are required!")
    elif self.where_entry.get() == "":
        messagebox.showerror("Error", "All fields are required!")
    elif self.gender_entry.get() == "":
        messagebox.showerror("Error", "All fields are required!")
    elif self.folk_entry.get() == "":
        messagebox.showerror("Error", "All fields are required!")
    elif self.contact_entry.get() == "":
        messagebox.showerror("Error", "All fields are required!")
    elif self.marital_choice.get() == "":
        messagebox.showerror("Error", "All fields are required!")
    else:
        db = mysql.connector.connect(user='datdt026', password='Bodoikuh0.', host='localhost', database='midtermpy')
        command_handler = db.cursor()
        querry_vals = (
        self.fentry.get(), self.lentry.get(), self.year_choice.get(), self.where_entry.get(), self.gender_entry.get(),
        self.folk_entry.get(), self.contact_entry.get(), self.marital_choice.get(), self.cccd_entry.get())
        command_handler.execute(
            "UPDATE people set Firstname=%s, Lastname=%s, Year_of_birth=%s, Home_town=%s, Gender=%s, Folk=%s, Contact_phone=%s, M
            querry_vals)
        db.commit()
        db.close()
        messagebox.showinfo("Success", "Updated Successfully")
```

+ Delete: delete one user’s data



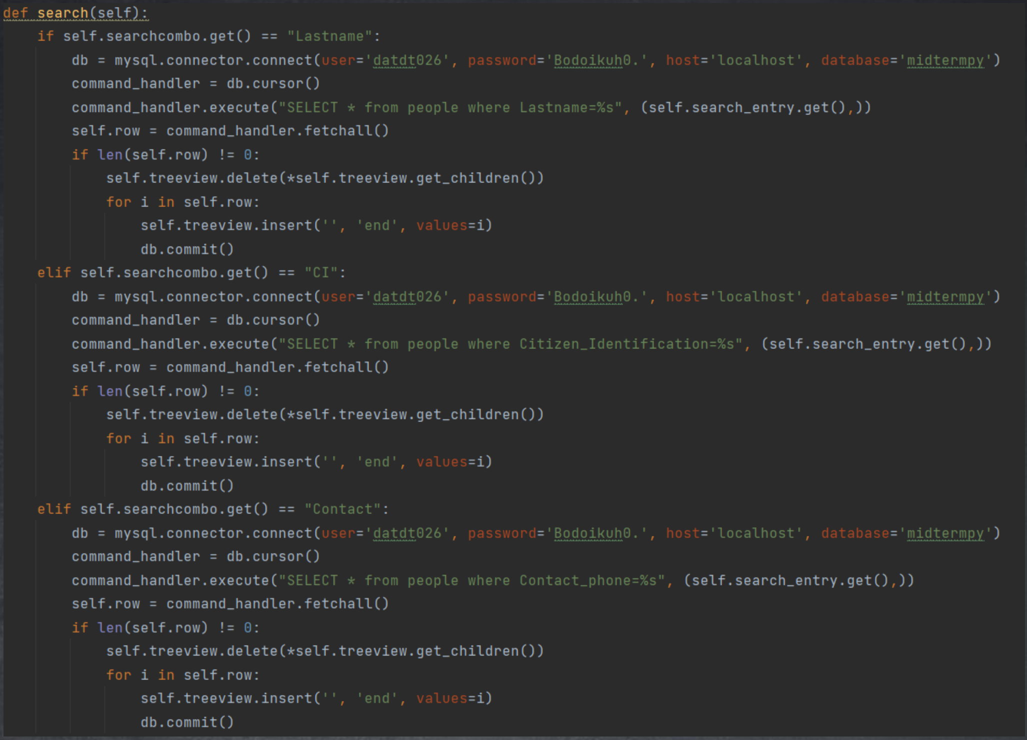
```python
def delete_citizen(self):
    if self.cccd_entry.get() == "":
        messagebox.showerror("Error", "All fields are required!")
    else:
        db = mysql.connector.connect(user='datdt026', password='Bodoikuh0.', host='localhost', database='midtermpy')
        command_handler = db.cursor()
        idcheck = (self.cccd_entry.get(),)
        command_handler.execute("DELETE from people WHERE Citizen_Identification= %s", idcheck)
        db.commit()
        db.close()
        messagebox.showinfo("Success", "Deleted Successfully")
```

+ Refresh: refresh all the fields to blank



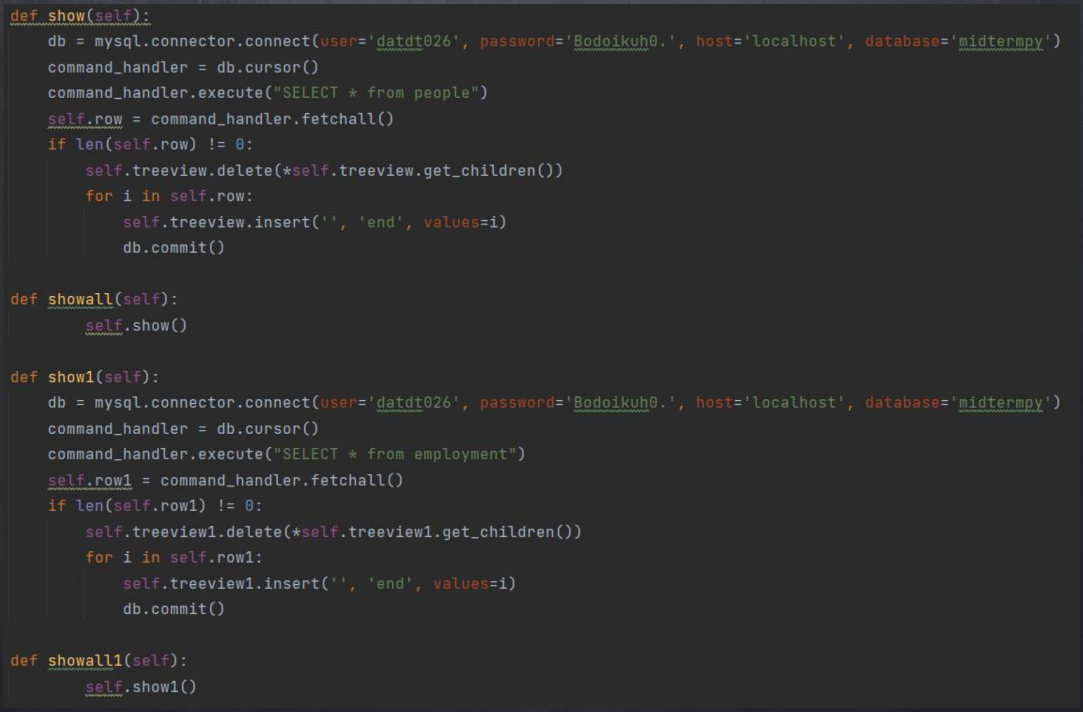
```python
def refresh(self):
    self.fentry.delete(0, END)
    self.lentry.delete(0, END)
    self.year_choice.set("  -Select year-  ")
    self.cccd_entry.delete(0, END)
    self.where_entry.delete(0, END)
    self.gender_entry.set("")
    self.folk_entry.delete(0, END)
    self.contact_entry.delete(0, END)
    self.marital_choice.set("")
```

+ In searching data, we can search by “Last name, Citizen ID, Phone, Name, Code, Year, Type, and Contact”:



```
def search(self):
    if self.searchcombo.get() == "Lastname":
        db = mysql.connector.connect(user='datdt026', password='Bodoikuh0.', host='localhost', database='midtermpy')
        command_handler = db.cursor()
        command_handler.execute("SELECT * from people where Lastname=%s", (self.search_entry.get(),))
        self.row = command_handler.fetchall()
        if len(self.row) != 0:
            self.treeview.delete(*self.treeview.get_children())
            for i in self.row:
                self.treeview.insert('', 'end', values=i)
                db.commit()
    elif self.searchcombo.get() == "CI":
        db = mysql.connector.connect(user='datdt026', password='Bodoikuh0.', host='localhost', database='midtermpy')
        command_handler = db.cursor()
        command_handler.execute("SELECT * from people where Citizen_Identification=%s", (self.search_entry.get(),))
        self.row = command_handler.fetchall()
        if len(self.row) != 0:
            self.treeview.delete(*self.treeview.get_children())
            for i in self.row:
                self.treeview.insert('', 'end', values=i)
                db.commit()
    elif self.searchcombo.get() == "Contact":
        db = mysql.connector.connect(user='datdt026', password='Bodoikuh0.', host='localhost', database='midtermpy')
        command_handler = db.cursor()
        command_handler.execute("SELECT * from people where Contact_phone=%s", (self.search_entry.get(),))
        self.row = command_handler.fetchall()
        if len(self.row) != 0:
            self.treeview.delete(*self.treeview.get_children())
            for i in self.row:
                self.treeview.insert('', 'end', values=i)
                db.commit()
```

+ List: show all the data in the table



```
def show(self):
    db = mysql.connector.connect(user='datdt026', password='Bodoikuh0.', host='localhost', database='midtermpy')
    command_handler = db.cursor()
    command_handler.execute("SELECT * from people")
    self.row = command_handler.fetchall()
    if len(self.row) != 0:
        self.treeview.delete(*self.treeview.get_children())
        for i in self.row:
            self.treeview.insert('', 'end', values=i)
            db.commit()

def showall(self):
        self.show()

def show1(self):
    db = mysql.connector.connect(user='datdt026', password='Bodoikuh0.', host='localhost', database='midtermpy')
    command_handler = db.cursor()
    command_handler.execute("SELECT * from employment")
    self.row1 = command_handler.fetchall()
    if len(self.row1) != 0:
        self.treeview1.delete(*self.treeview1.get_children())
        for i in self.row1:
            self.treeview1.insert('', 'end', values=i)
            db.commit()

def showall1(self):
        self.show1()
```

- If we forget to fill one or some fields, the system will show a window to announce you have to fill all the fields.

- If you fill all the fields, press one of four buttons, the system will announce you do it successfully.

- In doing this program, we have some difficulties:

+ Each member has own database.

+ Database is not big enough.

+ The GUI is simple.

+ We only have 1 account “admin” to manage the project, we have to log in to use the project.

- In the future:

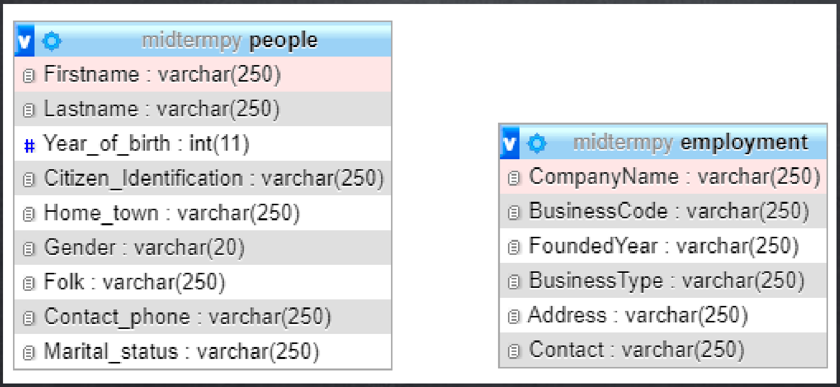
+ We hope that people have their own account to use the project, they can manage their own basic information.

+ We will have box to report about complains, requests, errors…..

+ Admin account will have different field to distinguish with normal account.

c. Database Diagram

- Diagram:



midtermpy people

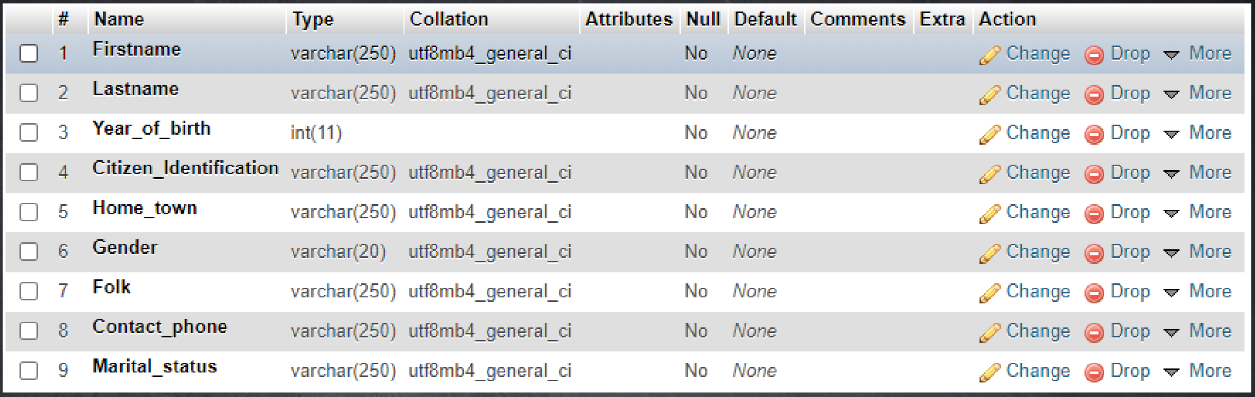
Firstname : varchar(250)
Lastname : varchar(250)
Year_of_birth : int(11)
Citizen_Identification : varchar(250)
Home_town : varchar(250)
Gender : varchar(20)
Folk : varchar(250)
Contact_phone : varchar(250)
Marital_status : varchar(250)

midtermpy employment

CompanyName : varchar(250)
BusinessCode : varchar(250)
FoundedYear : varchar(250)
BusinessType : varchar(250)
Address : varchar(250)
Contact : varchar(250)

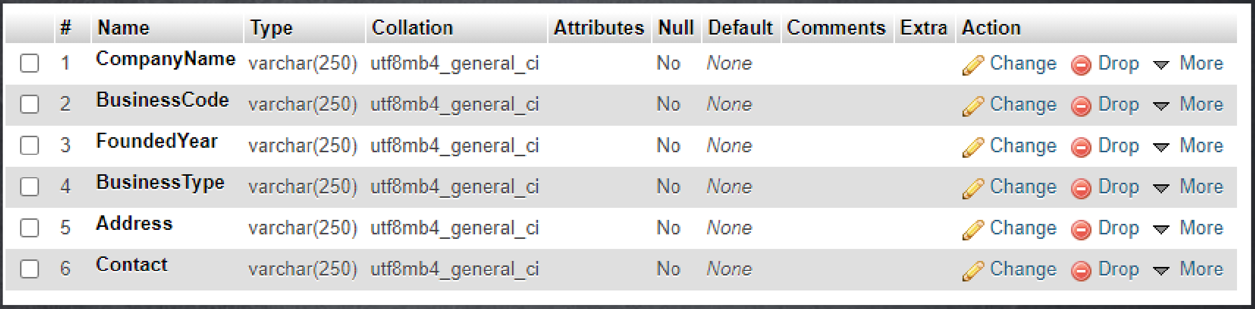
- Tables of fields in database:

+ People:



	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
□	1	Firstname	varchar(250)	utf8mb4_general_ci		No	*None*			✎ Change ⊖ Drop ▾ More
□	2	Lastname	varchar(250)	utf8mb4_general_ci		No	*None*			✎ Change ⊖ Drop ▾ More
□	3	Year_of_birth	int(11)			No	*None*			✎ Change ⊖ Drop ▾ More
□	4	Citizen_Identification	varchar(250)	utf8mb4_general_ci		No	*None*			✎ Change ⊖ Drop ▾ More
□	5	Home_town	varchar(250)	utf8mb4_general_ci		No	*None*			✎ Change ⊖ Drop ▾ More
□	6	Gender	varchar(20)	utf8mb4_general_ci		No	*None*			✎ Change ⊖ Drop ▾ More
□	7	Folk	varchar(250)	utf8mb4_general_ci		No	*None*			✎ Change ⊖ Drop ▾ More
□	8	Contact_phone	varchar(250)	utf8mb4_general_ci		No	*None*			✎ Change ⊖ Drop ▾ More
□	9	Marital_status	varchar(250)	utf8mb4_general_ci		No	*None*			✎ Change ⊖ Drop ▾ More

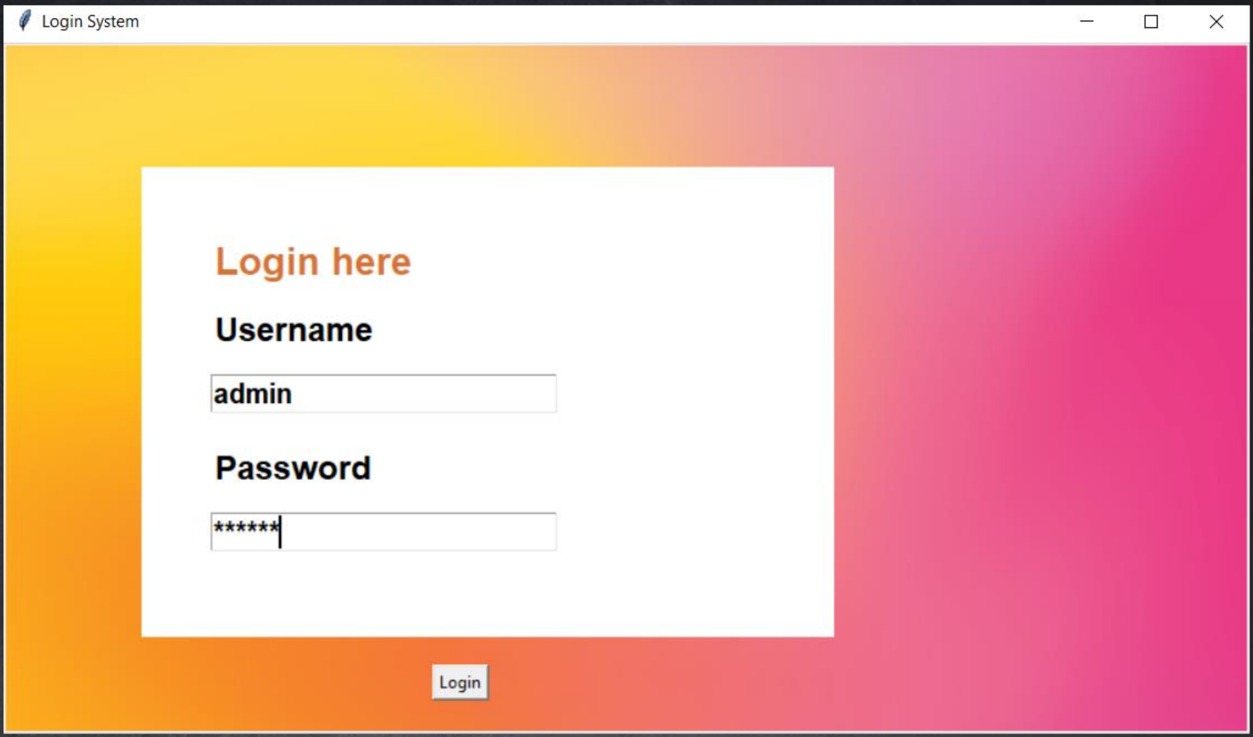
+ Employment:



	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
□	1	CompanyName	varchar(250)	utf8mb4_general_ci		No	*None*			✎ Change ⊖ Drop ▾ More
□	2	BusinessCode	varchar(250)	utf8mb4_general_ci		No	*None*			✎ Change ⊖ Drop ▾ More
□	3	FoundedYear	varchar(250)	utf8mb4_general_ci		No	*None*			✎ Change ⊖ Drop ▾ More
□	4	BusinessType	varchar(250)	utf8mb4_general_ci		No	*None*			✎ Change ⊖ Drop ▾ More
□	5	Address	varchar(250)	utf8mb4_general_ci		No	*None*			✎ Change ⊖ Drop ▾ More
□	6	Contact	varchar(250)	utf8mb4_general_ci		No	*None*			✎ Change ⊖ Drop ▾ More

3. Demo

- Login:



Login System

Login here

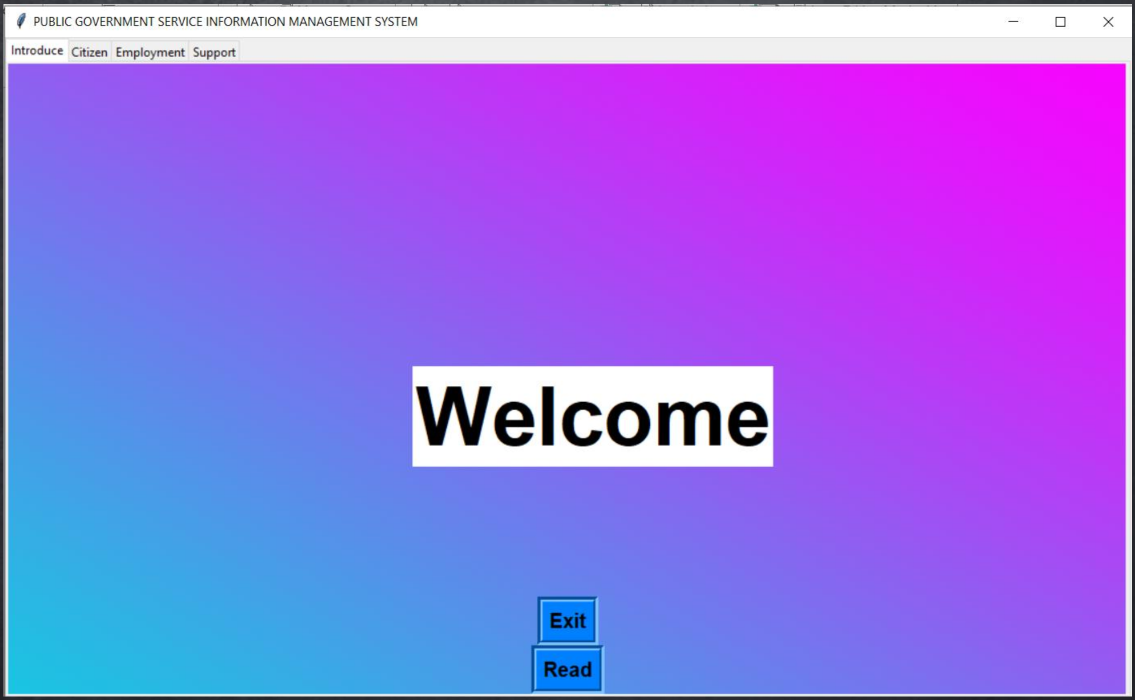
Username

admin

Password

******

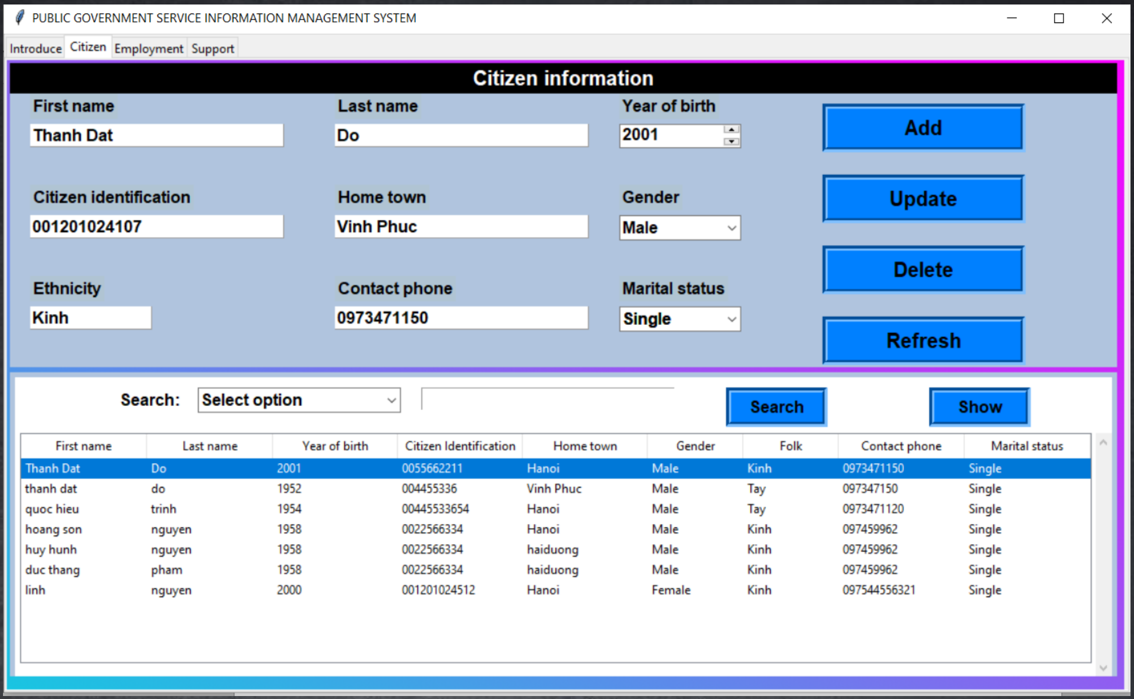
- Introduce



PUBLIC GOVERNMENT SERVICE INFORMATION MANAGEMENT SYSTEM

Welcome

- Citizen



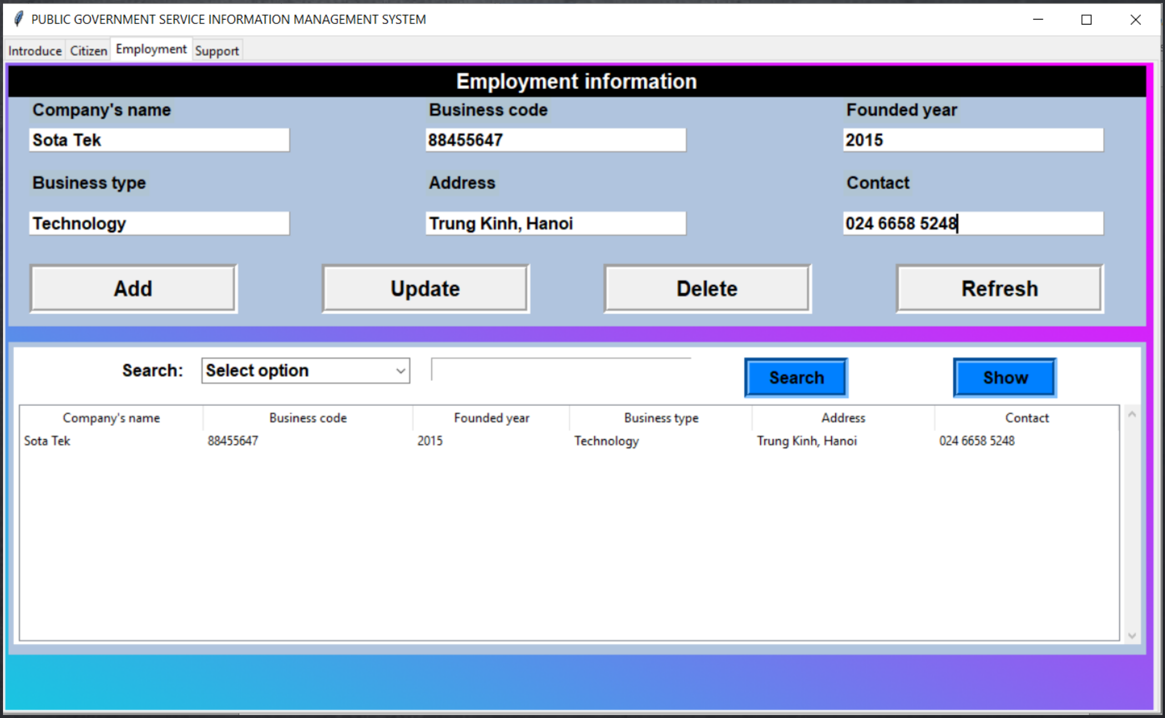
Citizen information

First name: Thanh Dat
Last name: Do
Year of birth: 2001
Citizen identification: 001201024107
Home town: Vinh Phuc
Gender: Male
Ethnicity: Kinh
Contact phone: 0973471150
Marital status: Single

Search: Select option

First name	Last name	Year of birth	Citizen Identification	Home town	Gender	Folk	Contact phone	Marital status
Thanh Dat	Do	2001	0055662211	Hanoi	Male	Kinh	0973471150	Single
thanh dat	do	1952	004455336	Vinh Phuc	Male	Tay	097347150	Single
quoc hieu	trinh	1954	00445533654	Hanoi	Male	Tay	0973471120	Single
hoang son	nguyen	1958	0022566334	Hanoi	Male	Kinh	097459962	Single
huy hunh	nguyen	1958	0022566334	haiduong	Male	Kinh	097459962	Single
duc thang	pham	1958	0022566334	haiduong	Male	Kinh	097459962	Single
linh	nguyen	2000	001201024512	Hanoi	Female	Kinh	097544556321	Single

- Employment



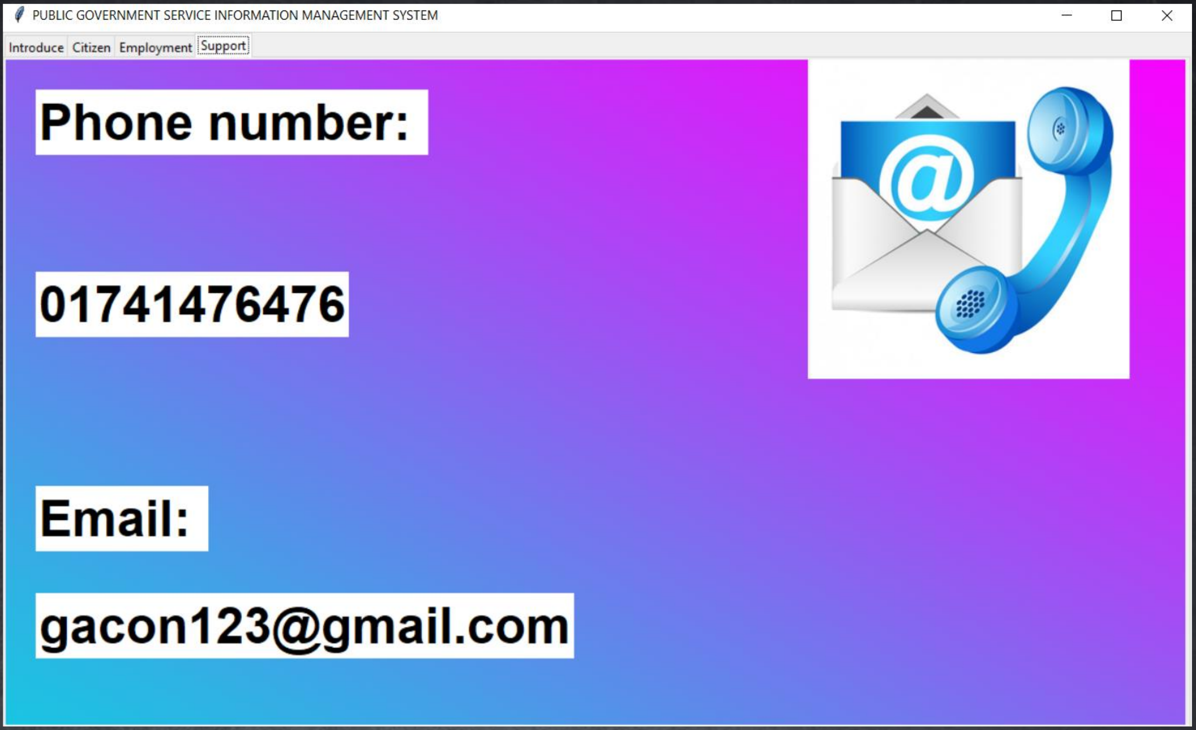
Employment information

Company's name: Sota Tek
Business code: 88455647
Founded year: 2015
Business type: Technology
Address: Trung Kinh, Hanoi
Contact: 024 6658 5248

Search: Select option

Company's name	Business code	Founded year	Business type	Address	Contact
Sota Tek	88455647	2015	Technology	Trung Kinh, Hanoi	024 6658 5248

- Support



Phone number:

01741476476

Email:

gacon123@gmail.com