Report of CSC3170 project

Web for Abiotic Factor

122090852 Kaiyi Zou

1. Steps to run this project

a. Firstly, you need to install your own mysql on your laptop. Use 'mysql -u -root -p' to open your mysql databases;

```
zoukaiyun@Kennys-Mac ~ % mysql -uroot -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.1.0 MySQL Community Server - GPL

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+--------------------+
| Database           |
```

b. Create the database 'AbioticFactor' and exit.

Using the sql script 'backup_file.sql' given to clone database. Enter the terminal and input 'mysql -u root -p AbioctiFactor < /path/to/backup_file.sql'.

Notice that the path must be your path to the backup_file.sql. Then you can check whether the database is created correctly. After that, input 'use AbioticFactor ;' to enter the database.

There are some tables within the databases:

```
mysql> show tables;
+--------------------------+
| Tables_in_abioticfactor |
+--------------------------+
| Entity                   |
| Location                 |
| staff                    |
| Theory                   |
+--------------------------+
4 rows in set (0.00 sec)
```

c. Use IDE to run the project. I use pycharm to run the project. You can use your prefered IDE as well. You need to install some package within your environment, such as pymysql, flask. Then, you can run the APP.py to start the project!

2. Overall description

Abiotic Factor is a popular video game. I like it very much so I decide to make a project regarding it.

There are several tables within the databases. The main role is staff, which is the key for you to login the web and view different information of different contents, including staff, location, theory, entity. I combine web page and database mysql using pymysql and flask.

This is a background information about this game:

"Home to the world's greatest minds, GATE operates a global network of secretive research laboratories, spanning every field of scientific study - and the realms beyond.

As a GATE employee you extend the bounds of human knowledge, and seek to explain the unexplainable - including anomalies and paranormal entities: from gravity-distorting artifacts to supernatural creatures with an unbridled instinct for violence. Safety, security, and secrecy are of the utmost importance… usually.

After a catastrophic containment breach, your workplace has become a cosmic battle zone: anomalous entities are on the loose, enemies from other dimensions are invading via portals, and an arcane military sect - known as The Order - is targeting personnel and entities indiscriminately, aiming to seize artifacts and put an end to the chaos.

Containment procedures have failed and help isn't on the way. Stranded miles beneath the surface, it's up to you and your fellow scientists to band together, plan your escape, and make this underground complex your new home - for now."

3. Requirement Analysis & database design

The main roles in this project are staffs. There are two kinds of staff, one has authority to add, edit, or delete information within the database, another has no such authority(normal staff can only see the information).

All of them can view the information of different contents. If you have no account, you can click sign in button in the main login page to register a new staff !

The followings are schema of different tables:

Entity table:

```
mysql> desc Entity;
+-------------+-------------+------+-----+---------+----------------+
| Field       | Type        | Null | Key | Default | Extra          |
+-------------+-------------+------+-----+---------+----------------+
| id          | int         | NO   | PRI | NULL    | auto_increment |
| codename    | varchar(32) | YES  |     | NULL    |                |
| name        | varchar(32) | YES  | UNI | NULL    |                |
| description | text        | YES  |     | NULL    |                |
+-------------+-------------+------+-----+---------+----------------+
4 rows in set (0.01 sec)
```

Location table:

```
mysql> desc Location;
+-------------+-------------+------+-----+---------+----------------+
| Field       | Type        | Null | Key | Default | Extra          |
+-------------+-------------+------+-----+---------+----------------+
| id          | int         | NO   | PRI | NULL    | auto_increment |
| name        | varchar(32) | YES  | UNI | NULL    |                |
| description | text        | YES  |     | NULL    |                |
+-------------+-------------+------+-----+---------+----------------+
3 rows in set (0.00 sec)
```

staff table:

```
mysql> desc staff;
+------------+-------------+------+-----+---------+----------------+
| Field      | Type        | Null | Key | Default | Extra          |
+------------+-------------+------+-----+---------+----------------+
| id         | int         | NO   | PRI | NULL    | auto_increment |
| name       | varchar(32) | YES  | UNI | NULL    |                |
| password   | varchar(32) | YES  |     | NULL    |                |
| authorized | int         | YES  |     | 0       |                |
| sector     | varchar(64) | NO   |     | NULL    |                |
+------------+-------------+------+-----+---------+----------------+
5 rows in set (0.01 sec)
```

Theory table:

```
mysql> desc Theory;
+-------------+-------------+------+-----+---------+----------------+
| Field       | Type        | Null | Key | Default | Extra          |
+-------------+-------------+------+-----+---------+----------------+
| id          | int         | NO   | PRI | NULL    | auto_increment |
| name        | varchar(64) | YES  |     | NULL    |                |
| description | text        | YES  |     | NULL    |                |
+-------------+-------------+------+-----+---------+----------------+
3 rows in set (0.00 sec)
```

The data in these tables are mostly data from the game, which is based on the game setting. Some of data comes from testing, I think you can easily distinguish.

4. SQL queries design

I have some comments in APP.py, you can check it.

If you want to add new data into database in terminal directly, you can use the followings SQL quries:

Add Entity:

insert into Entity(codename,name,description) values(“”, “”, “”);

Add Location:

insert into Location(name,description) values(“”, “”);

Add Staff:

insert into Staff(name,password,authorized,sector) values(“”, “”, 0/1, “”);

(a) Showing information use sql like

"select * from staff"

And use some javascript to show data in the form, which enable you to view them in the web page.

(b) Notice that %s are parameters

(c) Achieving login and sign up need to check the database if there are any staff matches.

Login check:

```
"select * from staff where name=%s and password=%s", (username, password)
```

Sign up:

```
insert into staff(name,password,sector) values(%s,%s,%s)
```

(d) Add new item, delete, edit are as followings(use Entity database as an example):

```
"insert into Entity(codename,name,description) values(%s,%s,%s)"
'DELETE FROM Entity WHERE id = %s'
"update Entity set name=%s, description=%s where codename=%s"
```

5. Front and back-end design and implemention

For front end design, I use flask, html, CSS and javascript to implement. Besides, I use some CSS from fontawesome website and bootstrapt website to modify my website to make it more beautiful. There are many different html file in folder 'template', which will show corressponding information in web page. One video from bilibili is quite educational and meaningful and I put it here:

【MySQL 数据库，保姆级 web 前端数据库精讲视频，99%的新手选择】
https://www.bilibili.com/video/BV1Ns4y1w7sg/?share_source=copy_web&vd_source=ba9569846bdf4f9af33dd437098e8516

It teaches how to implement html web page, how to combine web with mysql database.

For back-end design, Please see codes in APP.py, where I implement all functions needed by the project, including adding, deleting, editing new objects(Entity, staff, Location,Theory) and check staff which contains aroud 12 functions.

Some functions and its description are followings (many are similar) :

Requirement	Function	Function introduction
User Registration	`SignUp()` (Backend)	Handles user registration by receiving form input and storing the new user's data in the database.
Display Signup Page	`render_template("SignUpPage.html")` (Frontend)	Renders the signup HTML page where users enter their information to register.
User Login	`Login()` (Backend)	Displays the login page and handles user login requests.
Check User Login	`Check()` (Backend)	Verifies user credentials against the database and manages redirection based on authentication success or failure.
Display Login Page	`render_template("LoginMainPage.html")` (Frontend)	Renders the login page for users to input their credentials.
Entity List	`EntityList()` (Backend)	Fetches the list of entities from the database and renders them in the frontend.
Display Entity List	`render_template("EntityContent.html")` (Frontend)	Renders the list of entities on a webpage.
Staff List	`StaffList()` (Backend)	Retrieves and displays the staff list from the database.
Display Staff List	`render_template("StaffForm.html")` (Frontend)	Renders the list of staff on a webpage.
Location List	`LocationList()` (Backend)	Retrieves and displays the location list from the database.
Display Location List	`render_template("LocationContent.html")` (Frontend)	Renders the list of locations on a webpage.
Theory List	`TheoryList()` (Backend)	Retrieves and displays the list of theories from the database.
Display Theory List	`render_template("TheoryContent.html")` (Frontend)	Renders the list of theories on a webpage.
Delete Entity	`deleteEntity()` (Backend)	Deletes an entity from the database based on the provided ID and redirects to the entity list.
Delete Staff	`deleteStaff()` (Backend)	Deletes a staff member from the database based on the provided ID and redirects to the staff list.
Display Edit Theory Page	`render_template("EditTheoryPage.html")` (Frontend)	Renders the edit theory page for modifying theory details.
Signup Button Click	`button_click_for_signup()` (Backend)	Handles the signup button click event and redirects to the signup page.

Update: I also implement function to pagination and display 10 elements per page. You can check it.