Task 2 – [Web App] Seating Arrangement System

The school would like to implement an online seating arrangement system for class and subjects. In this task, you are required to implement a prototype using normalised database and flask web application to manage these records.

Please save all your files inside the `Task 2` folder.

The following information of each `Student` is stored:
`MatricNo` – unique string in the format of `"HS-YYYY-XXX"` where `YYYY` is the year of entry to school and `XXX` is a 3-digit string ranged from `"001"` to `"999"`.
`Name` – name of student
`Class` – class of student
`IndexNo` – index number of the student in the class
`Gender` – gender of student, to be stored as a single character, using either `"M"` or `"F"`

The following information of each `ClassGroup` is stored:
`ClassGroupID` – unique autoincrement integer to identify the class group
`ClassGroupName` – name of the class group, e.g. `"Computing"`
`Venue` – name of venue for this class group, e.g. `"4A Classroom", "iCode Lab"`

The following information of each `SCR` (Student-ClassGroup-Relationship) is stored:
`MatricNo` – matric number of the student
`ClassGroupID` – id of the class group

The information is to be stored in three tables:
`Student`
`ClassGroup`
`SCR`

Task 2.1

Create an SQL file called `Task2_1.sql` to show the SQL code to create the database `seating.db` with the three tables.

The table `Student` must use `MatricNo` as its primary key, and the table `ClassGroup` must use `ClassGroupID` as its primary key. The table `SCR` should use `MatricNo` and `ClassGroupID` as a composite key, while `MatricNo` and `ClassGroupID` must refer to `MatricNo` in `Student` and `ClassGroupID` in `ClassGroup` as foreign keys.

Save your SQL code as
`Task2_1.sql`

[5]

Task 2.2

The files `students.csv`, `classgroups.csv` and `scr.csv` contains information about the student, class groups and the Student-ClassGroup-Relationships. The first row of each file contains the header of the respective columns. Each row in the files is a comma-separated list of information.

Write a Python program to insert all information from the three files into the database `seating.db`. Run the program.

Save your program code as
`Task2_2.py` [5]

Task 2.3

A teacher would like to generate the class list of classgroup with the **ClassGroupName** **`"Comp_4AB"`**. Query and display a list of data with the following fields as shown in the table, sorted in the **ascending** order according to **Class**, followed by **IndexNo** of the student.

Class	IndexNo	Name	ClassGroupName
…	…	…	…

Write the SQL code required.

Save this code as
`Task2_3.sql` [6]

Task 2.4

The school wants to implement a function where teachers can enter the `ClassGroupName` and the system will generate the seating arrangement based on class and index number of students.

Write a Python program and the necessary files to create a web application that:

- Receive the `ClassGroupName` from a HTML form, then,
- Creates and returns a HTML document that enables the browser to display `(students class(index), name)` in a table format. As shown in this image:
-

Seating Plan

Seating Arrangement for Classgroup Comp_4AB:

('4A(3)', 'Chloe Loy')	('4A(4)', 'Ee Pei Chi Neoma')	('4A(6)', 'Hafizah Wong')	('4A(8)', 'Nichole Wong')	('4A(9)', 'Sandy Ho')
('4A(10)', 'Dave Khoo')	('4A(14)', 'Victor Fong')	('4B(3)', 'Devi Lieu')	('4B(4)', 'Emma Kaur')	('4B(6)', 'Neoma Chen')
('4B(7)', 'Benton Ling')	('4B(9)', 'Han Teck Heng Sterling')	('4B(11)', 'Rick Fu')	('4B(14)', 'Wyatt Danker')	

You may assume:

- Table have 5 columns
- Each row starts from the left hand side
- All inputs are valid

Save your program as
`Task2_4.py`
With additional files or sub-folders as needed in a folder named
`Task2_4`

Run the web application. Enter the following `ClassGroupName`:
`"Comp_4AB"`

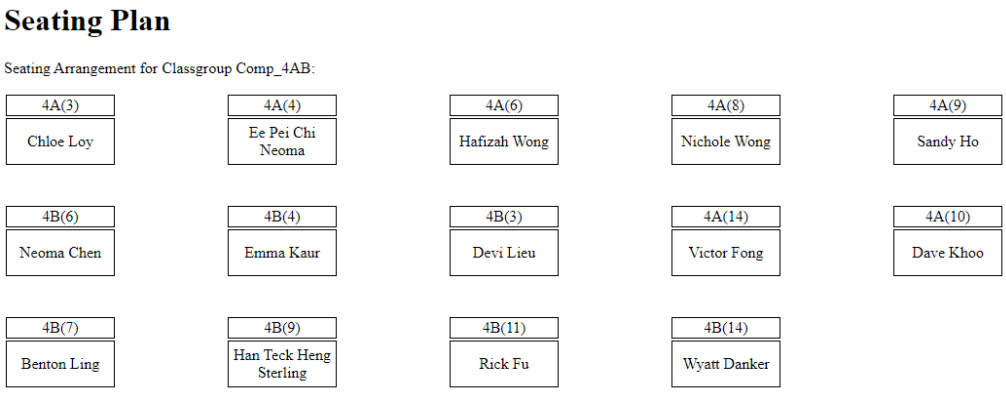
Then save the output of the program as `Task2_4.html`.

[14]

Task 2.5 [Bonus]

Lastly, if you would like to challenge yourself, copy paste your files from folder `Task2_4` to a new folder named `Task2_5`.
Attempt to modify your code to suit the following needs:

1. Search form can allow user to specify number of columns.
2. Snake shape with odd rows starting from left, even rows starting from right.
3. Use css style to display the data in the following format.



[2]

[End of Paper]