Flask: Database Operations

Recall

In Ch14.Python Database Connection, we discussed how to use package `pymysql` to connect Python applications and MySQL database.

Task 4.1: The first task here is to create the database tables we need in the application "UIC Calendar". We have two tables to be created, `dates_events` and `events`. Here are the tables with example data. You can create these tables with any methods anywhere.

- `dates`

date_id	date
1	2019-03-27
2	2019-03-28

  The primary key of this table is `date_id`, which has the type `INT` and it should be auto increased.
  Attribute `date` has the type `DATE`.
- `dates_events`

dates_events_id	date_id	event_id
1	1	1
2	2	2

  The primary key of this table is `dates_events_id`, which has the type `INT` and it should be auto increased.
  `dates_id` has the type `INT` and it is the foreign key connects dates.
  `event_id` has the type `INT` and it is the foreign key connects events.
- `events`

event_id	event_name
1	Mid-term exam

---

event_id	event_name
2	Staff annual party

`event_id` is the primary key in this table. It has the type `INT` and it should be auto increased. Attribute `event` has the type `TEXT`.

Query Events

The major tasks in this part are the following,

- Replace the fixed events list for index page with a database query.
- Make a new page with its functions to allow users to check events in any given date.

It is easy to discover that these two tasks can be finished with the same piece of code.

Task 4.2: Following the rules in software engineering, you are going to write a database operation class to handle all database connection and queries. Look at the following tips before you move on.

- Class structure

```
# application.py

import pymysql

class DatabaseOperations():
  # Fill in the information of your database server.
  __db_url = ''
  __db_username = ''
  __db_password = ''
  __db_name = ''
  __db = ''

  def __init__(self):
      """Connect to database when the object is created."""
      self.__db = self.db_connect()

  def __del__(self):
      """Disconnect from database when the object is destroyed."""
      self.__db.close()

  def db_connect(self):
      self.__db = pymysql.connect(self.__db_url, self.__db_username,
                                  self.__db_password, self.__db_name)
      return self.__db

  def query_events_by_date(self, date):
      # Finish this function to query events in given date.
```

- Sample SQL statement

> Disclaimer: this is NOT the only solution.

```
SELECT event FROM `events` WHERE `event_id` =
(SELECT `event_id` FROM `dates` INNER JOIN `dates_events`
ON dates.date_id = dates_events.date_id WHERE `date` =
str_to_date("2019-03-27","%Y-%m-%d"))
```

Now, refer to the sample, slides of Ch14 and any other materials to finish this database handling class.

Display Query Result

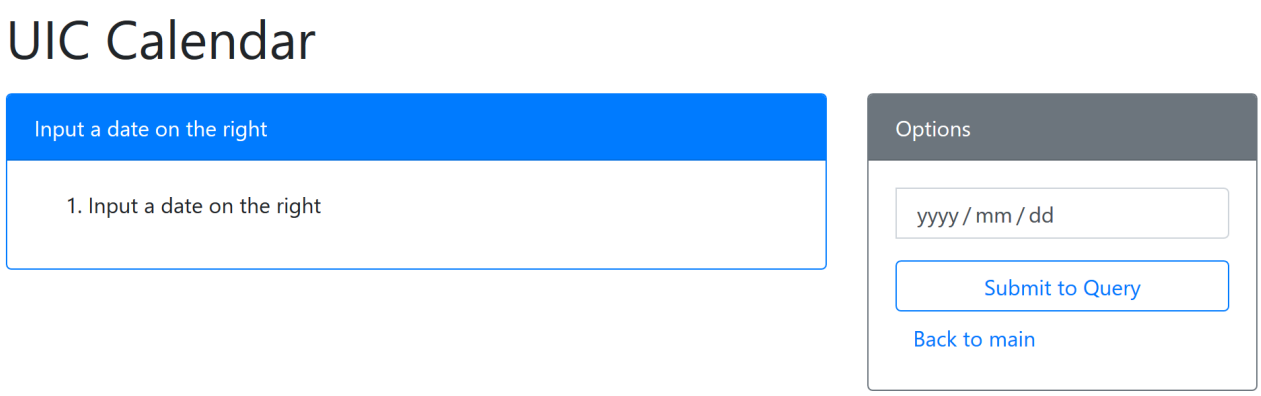
On the code provided to you, you can see a template called `query.html`. Now let's see how the page is constructed.

Write a function call `query()` in `application.py` to allow the rendering engine to render the page.

```
# application.py

@app.route('/query', methods=['GET', 'POST'])
def query():
    return render_template('query.html', date='Input a date on the right',
                           events=['Input a date on the right'])
```

Now, run your Flask application, as usual, input the URL `http://127.0.0.1:5000/query` to your browser, you should see the following page.



Task 4.3: Now change the method above, that allows the user to input a date and query the events of the given date from the database. Refer to your code in `login()` for form handling. After finish, you should see the result below.

Task 4.4: Refer to the code from the last task, change the method `index()`, to enable the system query today's events automatically from the database.

Recall the code in `application.py` for displaying `index.html`, you can see the data structure `events` that passes to rendering engine is a list. You should transfer the result tuple to a list.

References

- http://flask.pocoo.org/docs/1.0/tutorial/database/