CS 564
Database Management Systems
University of Wisconsin - Madison
Madison, WI 53706

Project Stage #5

Submitted by:
Jin Ruan
Shuang Wu
Qing Li

Submitted to:
Prof. AnHai Doan

Date Due: Apr 24, 2015

- Platforms & Languages:
We used SQLite as our relational database. Struts including jsp, java, xml is used as our web application.

- **Database Tables:**

We have ten tables in total. Here is one sample of creating a table:

```sql
CREATE TABLE COMMMAPSDEV.food_user (
    USER_ID      VARCHAR2(16) NOT NULL,
    USER_PASSWORD        VARCHAR2(16) NOT NULL,
    NICKNAME        VARCHAR2(16) NOT NULL,
    AVERAGE_RATINGS  NUMBER (2,1),
    MEMBER_SINCE   DATE DEFAULT (sysdate),
    PRIMARY KEY (USER_ID));
```

The ten tables are shown in the following screenshot:

- **Here is the interface looks like:**

Welcome to Online Food Paradise!

To get started, you need to enter your details to enroll with us. Or login to access your details, if you are already enrolled.

- **Here is the codes look like (Struts was used as MVC):**
• Login to insert tuples into the database:

Welcome to Online Food Paradise
Signup
It's absolutely quick!

Fill all the information in the form
Userid: jin\n
Password: jin\n
Nickname: jin

Signup

After filling the signup form, the codes will read the information in the frontend and translates it into the SQL insert query, here is the screenshot of the codes and the sql query it translated:
You will insert the information into the database and it will show up in the database:

- Search query will ask queries over database:
Welcome to Food Paradise, jin
Let’s have a tour of the local restaurants, special events, and make some friends.

Then since you selected restaurants, events, and users, it will ask all the matched restaurants, events, and restaurants. The query here involves joint operator. The SQL query the codes made will be:

```
SELECT * FROM COMMMAPSDEV.FOOD_EVENT,COMMMAPSDEV.FOOD_RESTAURANT WHERE EVENT_NAME LIKE '%t%' AND COMMMAPSDEV.FOOD_RESTAURANT.RESTAURANT_ID = COMMMAPSDEV.FOOD_EVENT.RESTAURANT_ID AND ROWNUM < 10
SELECT USER_ID,NICKNAME,AVERAGE_RATINGS,MEMBER_SINCE FROM COMMMAPSDEV.FOOD_USER WHERE NICKNAME LIKE '%t%' AND ROWNUM < 10
```
The returned tuples will show up in the webpage:

### Matched Restaurants

<table>
<thead>
<tr>
<th>Name</th>
<th>Address</th>
<th>Latitude</th>
<th>Longitude</th>
<th>Rating</th>
<th>Rating_Number</th>
<th>Review</th>
<th>Check In</th>
</tr>
</thead>
<tbody>
<tr>
<td>test1</td>
<td>test1</td>
<td>80</td>
<td>80</td>
<td>2.1</td>
<td>12</td>
<td>Review</td>
<td>Check In</td>
</tr>
<tr>
<td>test2</td>
<td>test2</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>Review</td>
<td>Check In</td>
</tr>
</tbody>
</table>

### Matched Events

<table>
<thead>
<tr>
<th>Name</th>
<th>Restaurant</th>
<th>Time</th>
<th>Event Cost</th>
<th>Description</th>
<th>Attend</th>
</tr>
</thead>
<tbody>
<tr>
<td>party</td>
<td>test1</td>
<td>2015-04-23 16:36:59.0</td>
<td></td>
<td></td>
<td>Attend</td>
</tr>
</tbody>
</table>

### Matched Users

<table>
<thead>
<tr>
<th>Nickname</th>
<th>Average Rating</th>
<th>Member_Since</th>
<th>Friend</th>
<th>Unfriend</th>
</tr>
</thead>
<tbody>
<tr>
<td>test</td>
<td>2015-04-23 16:29:18.0</td>
<td></td>
<td>Friend</td>
<td>Unfriend</td>
</tr>
<tr>
<td>test</td>
<td>2015-04-23 15:57:32.0</td>
<td></td>
<td>Friend</td>
<td>Unfriend</td>
</tr>
</tbody>
</table>

As you can see from the above screenshot, you can check in or review a restaurant, attend a event, friend or unfriend other users. For the friend, attend and checkin, after you clicking the button, it will add a record into the specific table in the database.
## Food Checkin Table

<table>
<thead>
<tr>
<th>CHECKIN_ID</th>
<th>USER_ID</th>
<th>RESTAURANT_ID</th>
<th>CHECKIN_TIMESTAMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>101</td>
<td>test</td>
<td>23-Apr-2015 19:53:25</td>
</tr>
<tr>
<td>2</td>
<td>102</td>
<td>test</td>
<td>23-Apr-2015 19:52:55</td>
</tr>
<tr>
<td>3</td>
<td>103</td>
<td>test</td>
<td>23-Apr-2015 19:51:41</td>
</tr>
<tr>
<td>4</td>
<td>104</td>
<td>test</td>
<td>23-Apr-2015 19:50:25</td>
</tr>
<tr>
<td>5</td>
<td>105</td>
<td>test</td>
<td>23-Apr-2015 19:49:59</td>
</tr>
</tbody>
</table>

## Food Event Table

<table>
<thead>
<tr>
<th>EVENT_ID</th>
<th>EVENT_NAME</th>
<th>RESTAURANT_ID</th>
<th>EVENT_TIMESTAMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Party</td>
<td>test</td>
<td>23-Apr-2015 19:51:41</td>
</tr>
</tbody>
</table>
• **Review function will query, insert, and update record in the database**

When you click the review button, it will firstly check if the review was already in the database. If it was, the previous review will return and enable you to edit it.

Welcome to Food Paradise, jin

Make a review.

Restaurant: test1  Rating: 1  Comments: it is horrible

After you change the previous review, it will do the modification query, which will update the original value.

```
UPDATE COMMMAPSDEV.FOOD_REVIEWS SET RATINGS = 2 WHERE RESTAURANT_ID = 'test1' AND USER_ID = 'jinruan'
UPDATE COMMMAPSDEV.FOOD_RESTAURANT SET RESTAURANT_RATING = '2.076923076923077' WHERE RESTAURANT_ID = 'test1'
```

If the user didn’t make a review before, after user submit the review, it will insert a review into the database.

For the unfriend button, it will delete the record that is in the friend table.

For other functions like login, it will do the select query to ask the database to return the records satisfying conditions.