Web Services Project

Télécom Saint-Étienne

Joseph KAMEL _ Yazan MUALLA

1. Background

REpresentational State Transfer (REST) was originally introduced as an architectural style for building large-scale distributed hypermedia systems. It is used to explain the excellent scalability of the HTTP 1.0 protocol. The REST architectural style is based on four principles:

- Resource identification through URI.
- Uniform interface.
- Self-descriptive messages.
- Stateful interactions through hyperlinks.

2. Technologies

We have used the following technologies in our project:

- Eclipse (java);
- Spring framework;
- MySQL as a database and JDBC as a connector with business logic code.

3. Main Functionalities

3.1. DataBase Handling

- **Class handling connection**

webServicesProject\src\main\java\emse\fr\database\DBClass.java

- **Connection method**

```
Class.forName("com.mysql.jdbc.Driver");

Connection con =

DriverManager.getConnection ("jdbc:mysql://localhost:3306/WS_PROJECT", "root", "?");
```

- **Classes handling tables**

webServicesProject\src\main\java\emse\fr\model\{Group.java, Main.java, User.java}

3.2. Database Description

- **USERS table**

It includes all users with ‘email’ as primary key.

- **GROUPS table**

It includes all groups with ‘name’ as primary key

- **GROUP_USER table**

Each user may be in more than one group, and each group may contain more than one user. We used this table to handle the Many-to-Many relation between the table (GROUPS) and the table (USERS).

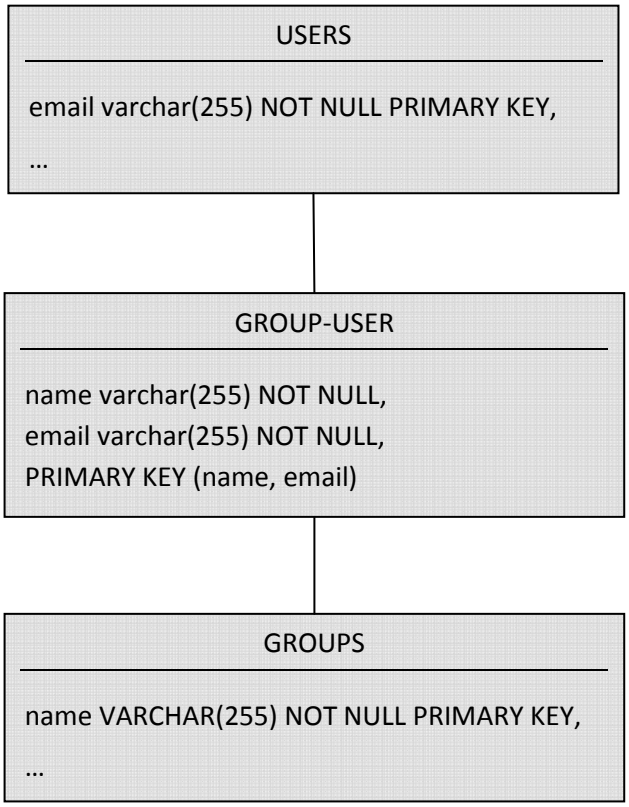

Figure 1: Handling Many-to-Many relation between USERS and GROUPS tables

- **COMMENTS table**

It includes all users comments.

3.3. RESTfull Handling

Entities are stored as data base tables but they are addressable, interconnected resources, that can be accessed as uniform HTTP request and the result will be formulated as suitable representations (HTML and JSON). Only representations are exposed to the client, and user can not access data base directly, as everything is access with HTTP requests. Basically we are accessing URL that corresponds to the resource page with the appropriate HTTP verb. The server does not store the state of the user. All REST communications to achieve the needed functionalities are listed below:

- **Classes handling REST communication**

webServicesProject\src\main\java\emse\fr\controller\

{ RESTController.java, HTMLController.java}

- **Classes handling User functionalities**

Table(1) shows all details about RESTfull business logic and communication between server and client related to the user functionalities. It is mainly handled by Spring framework.

Functionality	URL	Business logic functions	Parameters	Affected Resources
User sign up	... /newuser	newUser	-@RequestParam("email") -@RequestParam("first_name") -@RequestParam("last_name") -@RequestParam("brief_biography")	USERS
User changes his/her full name	... /changename	changeName	-@RequestParam("email") -@RequestParam("first_name") -@RequestParam("last_name")	USERS
User changes his/her biography	... / changebiography	changeBiography	-@RequestParam("email") -@RequestParam("biography")	USERS
User deletes his/her account	... / deteleaccount	deteleAccount	@RequestParam("email")	-USERS -GROUP_USER
User joins group	... /joingroup	joinGroup	-@RequestParam("email") -@RequestParam("name")	GROUP_USER
User leaves the group	... /leavegoup	leaveGroup	-@RequestParam("email") -@RequestParam("name")	GROUP_USER
User checks the profile of other users	... /checkprofile	checkProfile	@RequestParam("email")	USERS
User comments on the dashboard of the group	... /comment	newComment	-@RequestParam("email") -@RequestParam("name") -@RequestParam("comment")	COMMENTS

Table 1: User functionalities

- **Classes handling Group functionalities**

Table(2) shows all details about RESTfull business logic and communication between server and client related to the group functionalities. It is mainly handled by Spring framework.

Functionality	URL	Business logic functions	Parameters	Affected Resources
User creates new group	... /newgroup	newGroup	-@RequestParam("name") -@RequestParam("description") -@RequestParam("admin")	GROUPS
User changes the description of the group he/she owns	... / changedescription	newGroup (override)	-@RequestParam("name") -@RequestParam("description")	USERS
User deletes groups he/she owns	... /detelegroup	deteleGroup	@RequestParam("name")	-GROUPS -GROUP_USER

User views a list of groups with descriptions and membership count	… /listgroups	listGroups	-	GROUPS
User list groups a user is member of (Extra)	… / listjoinedgroups	listJoinedGroups	@RequestParam("email")	-GROUPS -GROUP_USER
User views members of the group he/she owns or has joined	… /viewmembers	viewMembers	@RequestParam("name")	GROUP_USER
User check the profile of a group (Extra)	… /checkgroup	checkProfile	@RequestParam("name")	GROUPS

Table 2: Group functionalities

- **REST Principles**

Following are the REST principles and indicating what our project adhere:
1. Resource identification (YES): Modeled resources are Groups, Users and Comments.
2. Addressability (NO):
Clients should be able to refer to the resources, a service is addressable if it exposes its data set as resources, each one having a URI. But we did not have time to handle the naming consistency of resources in the URLs.
3. Statelessness (YES): The server does not track or keep any information about the user or client requests. Database is purely about entities and their relations. Every time the server relies on the parameters passed in URL.
4. Resource representations (YES)
Html and JSON
5. Links & Connectedness (NO)
6. Uniform Interface (YES)
All requests from client to server are done using the URL. No parameters are being passed with the header of the POST verb. In contrast, all parameters are passed as path parameters in the URL.
As REST is application-centric, it is a guide for web services not for web sites, that is why our goal was not to create nice web pages interfaces. Instead, we used HTML and JSON to communicate the information.

- **Notes**

- We were working on the authentication of users and admins in separate code, but we did not have the time to combine them with the main functionalities of the project.

- The web interface we have developed was just to show data and interact with user, while all the functionalities were achieved by passing parameters in the URL using HTTP verbs.

4. Conclusion

We have used several technologies in our project. We have used MySQL as database and connected it to eclipse (Java) using JDBC. We depended on Spring framework as the main handler of communications between client and server. Database include four tables (USERS, GROUPS, GROUP_USER, Comments). We had to break the many-to-many relation between USERS and GROUPS using the new table GROUP_USER. All user and group functionalities were handled using HTTP methods with parameters passed in the URL. We have listed all the needed functionalities, used links, business logic functions in the server side, passed parameters and the affected resources in the database.

The main point behind using RSET is that it uses HTTP as an application protocol, so application is part of the web. At the end, we have discussed to what extinct our project was using REST.