Alex Rappa
10/22/2016
CS 496

Assignment 3: Part 2

Due Monday by 1:59am **Points** 10 **Submitting** a file upload
File Types zip **Available** after Oct 18, 2015 at 2am

Backend API URL: http://cs-496-webapp.appspot.com

1. URL structure used to access resources

Resource	POST	GET	PUT	DELETE
/user	Create new user	List all users	Update all users	Delete all users
/user/<id>	N/A	Show user	Update user	Delete user
/board	Create new board	List all boards	Update all boards	Delete all boards
/board/<id>	N/A	Show board	Update board	Delete board
/post	N/A	List all posts	Update all posts	Delete all posts
/post/<id>	N/A	Show post	Update post	Delete post
/board/<id>/user	N/A	List all subscribed users	Bulk subscribed user update	Delete all subscribed users
/board/<id>/user/<id>	Subscribe user to board	Show user	Update user	Remove subscribed user
/board/<id>/user/<id>/post	Create new post in board by user	Show post	Update post	Delete post

2. RESTful constraints you met and which you did not

Client-server: Met, the database is entirely managed by this backend API and has no frontend UI.

Stateless: Met, there is no state at the backend that is maintained by the API

Cacheable: Not Met, I have not implemented any caching functionality.

Layed System: Met, as layed out in the URL structure, different HTTP requests are sent to the same base handler but backend API actions are dependent on the type of request. The user never sees this and only needs to concern themselves with URL structure listed above.

Uniform Interface: Met, the URL structure for Users, Boards, and Posts are all similar which allow clients to observe consistency & have accurate expectations on how to interact with the backend API.

3. A discussion of any changes you needed to make to the schema from last week

I removed some Model's extraneous properties from last week's assignment. Additionally, I left the Comment ndb model out, since I already have 3 other models to work with. Also the

Comment class would function the same way as the Post class except Post is on a Board and Comment is on a Post. So it's implementation would be redundant for this assignment.

4. Tests

Part 1: Users A – G

1.A. Creating Users

```
> curl --data "username=testUser_1" --data "email=testUser_1@email.com" -H "Accept: application/json"
http://localhost:9080/user
{"key": 5681726336532480, "email": "testUser_1@email.com", "username": "testUser_1"}

> curl --data "username=testUser_2" --data "email=testUser_2@email.com" -H "Accept: application/json"
http://localhost:9080/user
{"username": "testUser_2", "key": 5118776383111168, "email": "testUser_2@email.com"}

> curl --data "username=testUser_3" --data "email=testUser_3@email.com" -H "Accept: application/json"
http://localhost:9080/user
{"username": "testUser_3", "key": 6244676289953792, "email": "testUser_3@email.com"}
```

1.B. List Users

```
> curl -H "Accept: application/json" http://localhost:9080/user
{"keys": [5118776383111168, 5681726336532480, 6244676289953792]}
```

1.C. Get User by ID

```
> curl -H "Accept: application/json" http://localhost:9080/user/5681726336532480
{"username": "testUser_1", "key": 5681726336532480, "email": "testUser_1@email.com"}

> curl -H "Accept: application/json" http://localhost:9080/user/5118776383111168
{"username": "testUser_2", "key": 5118776383111168, "email": "testUser_2@email.com"}

> curl -H "Accept: application/json" http://localhost:9080/user/6244676289953792
{"username": "testUser_3", "key": 6244676289953792, "email": "testUser_3@email.com"}
```

1.D. Delete User by ID

```
> curl -X DELETE -H "Accept: application/json" http://localhost:9080/user/5681726336532480
User deleted
> curl -H "Accept: application/json" http://localhost:9080/user
{"keys": [5118776383111168, 6244676289953792]}

> curl -X DELETE -H "Accept: application/json" http://localhost:9080/user/5118776383111168
User deleted
> curl -H "Accept: application/json" http://localhost:9080/user
{"keys": [6244676289953792]}

> curl -X DELETE -H "Accept: application/json" http://localhost:9080/user/6244676289953792
User deleted
> curl -H "Accept: application/json" http://localhost:9080/user
{"keys": []}
```

1.E. Delete All Users

```
> curl -X DELETE -H "Accept: application/json" http://localhost:9080/user
All Users Deleted
```

1.F. Get User by ID that do not exit

```
> curl -H "Accept: application/json" http://localhost:9080/user/123
"404 User Not Found"
```

1.G. Delete Users that does not exist

```
> curl -X DELETE -H "Accept: application/json" http://localhost:9080/user/123
> "404 User Not Found"

> curl -X DELETE -H "Accept: application/json" http://localhost:9080/user/123
No Users to Delete
```

Part 2: Message Boards

2.A. Create Boards

```
> curl --data-urlencode "name=Test Board 1" --data-urlencode "posts[]=Post_1" --data-urlencode
"posts[]=Post_2" --data-urlencode "posts[]=Post_3" -d "users[]=5100084685438976" -d
"users[]=5663034638860288"  -H "Accept: application/json" http://localhost:9080/board
{"updates": [], "name": "Test Board 1", "users": [5100084685438976, 5663034638860288], "posts": ["Post_1",
"Post_2", "Post_3"], "key": 5259513871466496}

> curl --data-urlencode "name=Test Board 2" --data-urlencode "posts[]=Post_1" -H "Accept: application/json"
http://localhost:9080/board
{"key": 6385413778309120, "name": "Test Board 2", "users": [], "posts": ["Post_1"], "updates": []}
```

Test Board 1 is created with 2 users subscribed, and 3 posts.

Test Board 2 is created with 0 users subscribed and 1 post.

2.B. List All Boards

```
> curl -H "Accept: application/json" http://localhost:9080/board
{"keys": [5259513871466496, 6385413778309120]}
```

2.C. Get Board by ID

```
>curl -H "Accept: application/json" http://localhost:9080/board/5259513871466496
{"key": 5259513871466496, "name": "Test Board 1", "users": [5100084685438976, 5663034638860288],
"posts": ["Post_1", "Post_2", "Post_3"], "updates": []}

>curl -H "Accept: application/json" http://localhost:9080/board/6385413778309120
{"key": 6385413778309120, "name": "Test Board 2", "users": [], "posts": ["Post_1"], "updates": []}
```

2.D. Add User to Board

```
>curl -X PUT -H "Accept: application/json" http://localhost:9080/board/6385413778309120/user/6526151266664448
{"updates": [], "users": [6526151266664448], "key": 6385413778309120, "posts": ["Post_1"], "name": "Test Board 2"}
```

2.E. Remove User from Board

```
>curl -X DELETE -H "Accept: application/json" http://localhost:9080/board/6385413778309120/user/6526151266664448
> {"users": [], "updates": [], "name": "Test Board 2", "key": 6385413778309120, "posts": ["Post_1"]}
```

2.F. Delete Board by ID

```
>curl -X DELETE -H "Accept: application/json" http://localhost:9080/board/5259513871466496
Board deleted
```

2.G. Delete All Boards

```
>curl -X DELETE -H "Accept: application/json" http://localhost:9080/board
All Boards Deleted

>curl -H "Accept: application/json" http://localhost:9080/board
{"keys": []}
```

Part 3: Posts

3.A. Create Post

```
>curl --data-urlencode "title=Post Title" --data-urlencode "content=this is the content of the post" -H "Accept:
application/json" http://localhost:9080/board/5222955109842944/user/5785905063264256/post
{"title": "Post Title", "author": 5785905063264256, "board": 5222955109842944, "key": 6630329993396224,
"content": "this is the content of the post"}
```

This creates a Post in Board (ID: 5222955109842944) authored by User (ID: 5785905063264256). User must be subscribed to Board in order to post there.

3.B. List All Posts

```
>curl -H "Accept: application/json" http://localhost:9080/post
{"keys": [4941480133132288, 5504430086553600, 6067380039974912, 6348855016685568,
6630329993396224]}
```

3.C. Get Post by ID

```
>curl -H "Accept: application/json" http://localhost:9080/post/4941480133132288
{"content": "this is the content of the post", "board": 5222955109842944, "author": 5785905063264256,
"title": "Post Title", "key": 4941480133132288}
```

3.D. Delete Post by ID

```
>curl -X DELETE -H "Accept: application/json" http://localhost:9080/post/4941480133132288
"Post (4941480133132288) deleted"
>curl -X DELETE -H "Accept: application/json" http://localhost:9080/post/4941480133132288
"404 Post (4941480133132288) Not Found"
```

3.E. Delete All Posts

```
>curl -X DELETE -H "Accept: application/json" http://localhost:9080/post
"All Posts Deleted"
>curl -X DELETE -H "Accept: application/json" http://localhost:9080/post
"No Posts to Delete"
```

5. Having finished the API, what you would have done differently

A. Towards the end of this assignment I began to realize I had been typing a lot of logic over and over again. The Post model is the last entity I added and decided to make some helper functions for the BaseHandler class. The main three functions I used were *insufficientParams(), notFound(),* and *isValidRequest()*. These functions really cleaned up the code because I could just pass a parameter to them and get the response I desired. In my source code I left the User and Board classes without these functions, and only implemented them in the Post class. As a result, the code in the Post class looks much cleaner, is shorter, and a lot easier to following and understand.

B. I failed to write out he URL structure for the resources beforehand. This resulted in a lot of reword on my part that could have been easily avoided if I had taken the time to plan it out from the beginning to ensure consistency.