

WEB PROGRAMMING LANGUAGE

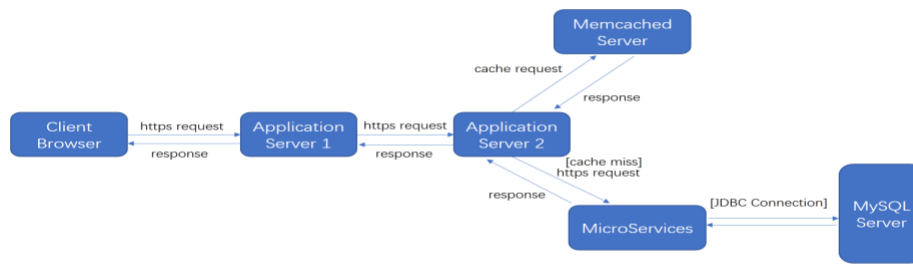
2017 Fall

Blue Leaf

BY
Radhika Kalaiselvan,
Akash Kumar Agile Mohan,
Gaojie Wang

2017/12/2

Project Architecture



Description of technologies used

Application Server 1:

- JSP, Java Servlet, jQuery, Bootstrap.
Choose Java over PHP, because Java language have high performance, and it is a good chance to learn more about Java EE.
Choose jQuery over JavaScript, because jQuery is more easy to write, and code looks cleaner. Bootstrap is free and open-source, it is very popular, and have great documents to work with.

Application Server 2:

- Java, RESTful, SSL/TLS.
Java is a very popular choice to develop business web applications, and it is a good change to practice what we learned in this class. RESTful is less rely on tools, easier to understand and develop. SSL/TLS give us a secured way to transport data which is crucial for a real business application.

Micro Services:

- Java, RESTful, SSL/TLS.
RESTful is less rely on tools, easier to understand and develop

Memcached Server:

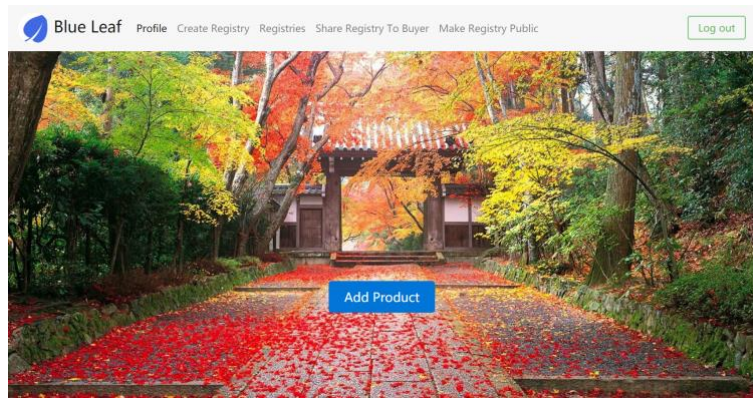
- Java, Caching, Memcached.
Caching can improve the performance of web service. Memcached is a distributed memory object caching system. It can greatly speed up web application.

MySQL Server:

- Java, RESTful, MySQL.
MySQL is open-source and free, it is one of the most powerful RDBMS and extremely widely used.

Description of functionalities

1. New regular user registration
User can sign up in our register page with his email, password is required.
2. Regular user log in, log out.
User can log in and out with his email and password after signing up.

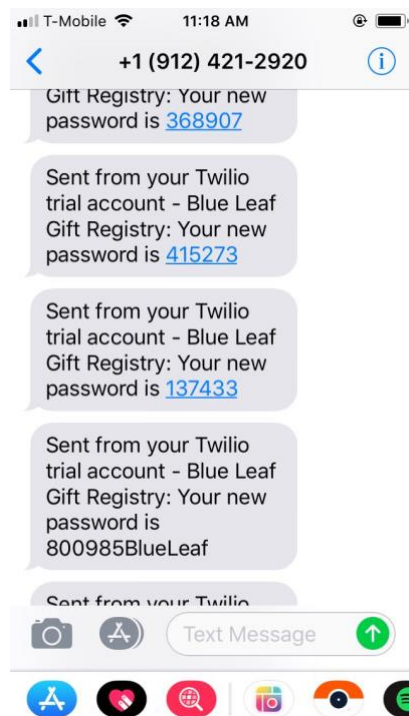


3. User profile information display and modification

User can edit his profile. Email cannot be edit.

4. Forgot password functionality

Twilio <https://www.twilio.com> provides a wide variety of APIs to send SMS messages. We used these APIs to send OTP to the registered mobile number which will be used a temporary password. The user can later login and reset the password in profile information page.



5. Ability to create a registry

User can create a registry, give the registry a name and make it public or private.

6. Display products in table, filter products on four properties

When user try to add product to their registry, we will display products in table, and user has four optional properties to filter the products.

List of Products

HOME

Filters

Category

☐ Accessories
☐ Electronics
☐ Instruments

Rating





☐ 1
☐ 2
☐ 3
☐ 4
☐ 5

Brand

☐ Chanel
☐ Apple
☐ Nike

Certification

☐ 1 YES
☐ 0 NO

RegisterID	ProductImage	ProductName	BrandName	ProductID	Price	Certification	Rating	Category		
*		<input type="text" value="Enter RegID"/>	PEN	Chanel	900	40.0	0	4	Accessories	<input type="button" value="AddToRegistry"/>
*		<input type="text" value="Enter RegID"/>	IPHONE	Apple	902	1400.0	1	5	Electronics	<input type="button" value="AddToRegistry"/>
*		<input type="text" value="Enter RegID"/>	MAC	Apple	905	1400.0	1	5	Electronics	<input type="button" value="AddToRegistry"/>
*		<input type="text" value="Enter RegID"/>	BIKE	Nike	906	300.0	1	4	Instruments	<input type="button" value="AddToRegistry"/>

7. Ability to add/remove items from a registry

User can add products to his registry, and in the display table of his registry, user has option to delete products from registry.

8. Ability to share registry to particular user or make it public

User can share his registry to other user, which means the other user will get the information about this registry and be able to make a purchase with it. User can make his private registry public which means visible to the public.

9. Self-assign an item on another user's registry

If user be shared a registry, he/she will be able to self-assign an item to the user who shared the registry.

10. 404 page

When we try to access some resource which does not exist, a 404 page will show up.

11. Admin log in, log out

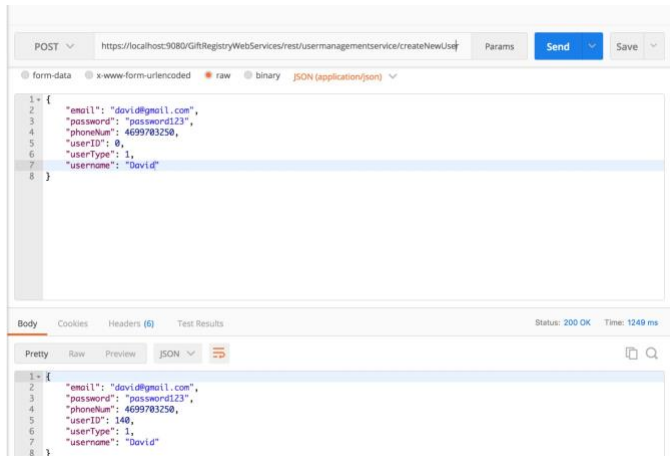
Admin user can log in and out of his account.

12. Admin can Add/remove items into/from inventory

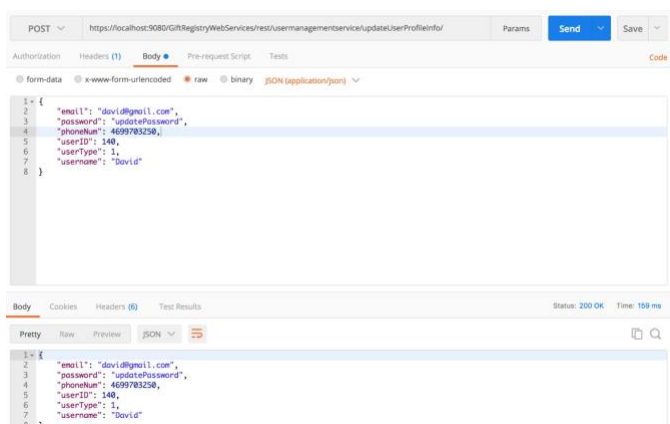
Admin can add a new product to the inventory, or remove already exist products.

Functionalities via a web service API

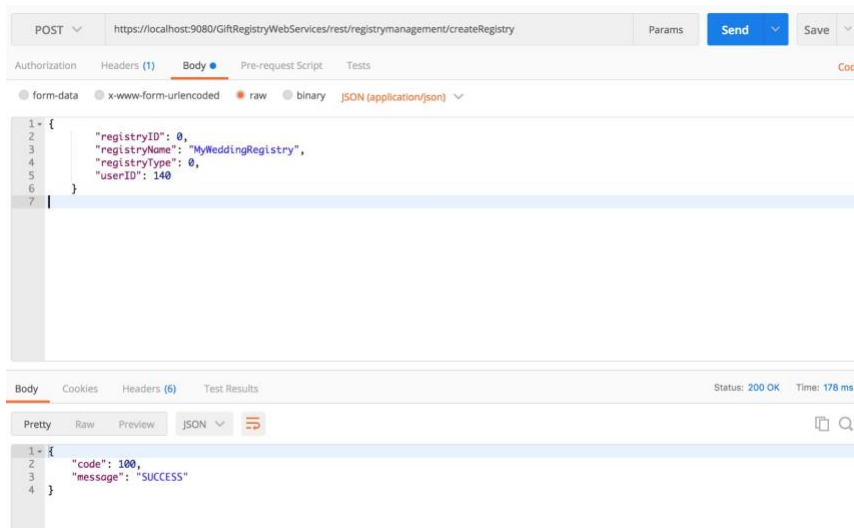
1. User registration



2. User profile information access and modification



3. Create a registry



4. Search for items in the inventory

```
GET https://localhost:9080/GiftRegistryWebServices/rest/productregistrymanagement/getProductsFromRegistry/921... Params Send Save Status: 200 OK Time: 223 ms

Pretty Raw Preview JSON

9  {
10  "imageUrl": "",
11  "price": 1400,
12  "productID": 905,
13  "productName": "MAC",
14  "rating": 5
15  },
16  {
17    "assignedTo": "-",
18    "brandID": 1,
19    "brandName": "LAPTOP",
20    "category": "Electronics",
21    "certification": 0,
22    "imageUrl": "",
23    "price": 1900,
24    "productID": 906,
25    "productName": "LAPTOP",
26    "rating": 5
27  },
28  {
29    "assignedTo": "John",
30    "brandID": 1,
31    "brandName": "TABLET",
32    "category": "Electronics",
33    "certification": 0,
34    "imageUrl": "",
35    "price": 1900,
36    "productID": 908,
37    "productName": "TABLET",
38    "rating": 5
39  }
40  ]
41 }
```

5. Add/Remove items from a registry

```
GET https://localhost:9080/GiftRegistryWebServices/rest/productregistrymanagement/addProductToRegistry/903/905... Params Send Save

Authorization Headers (1) Body Pre-request Script Tests

Type No Auth

Body Cookies Headers (6) Test Results Status: 200 OK Time: 16

Pretty Raw Preview JSON

1 {
2   "code": 100,
3   "message": "SUCCESS"
4 }
```

```
GET http://localhost:9080/GiftRegistryWebServices/rest/productregistrymanagement/deleteProductFromRegistry/903/905... Params Send

Authorization Headers (1) Body Pre-request Script Tests

Type No Auth

Body Cookies Headers (6) Test Results Status: 200

Pretty Raw Preview JSON

1 {
2   "code": 100,
3   "message": "SUCCESS"
4 }
```

6. Share registry to particular user or make it public

```
GET http://localhost:9080/GiftRegistryWebServices/rest/registrysharing/shareprivateRegistry/901/124/david@gmail.com Params Send Save

Authorization Headers (6) Body Pre-request Script Tests

Type No Auth

Body Cookies Headers (6) Test Results Status: 200 OK Time

Pretty Raw Preview JSON

1 {
2   "code": 100,
3   "message": "SUCCESS"
4 }
```

```
GET http://localhost:9080/GiftRegistryWebServices/rest/registrymanagement/makepublic/901 Params Send

Authorization Headers (6) Body Pre-request Script Tests

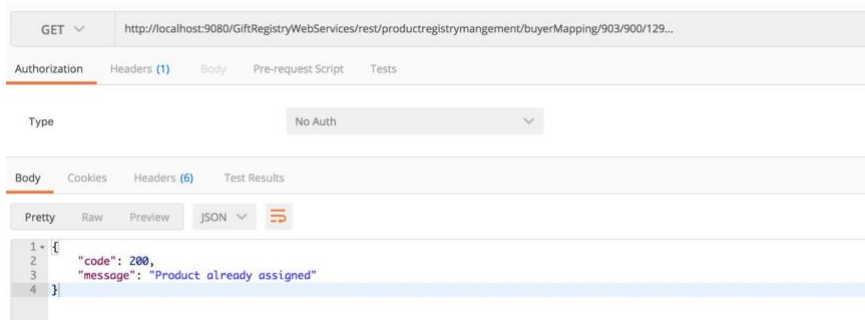
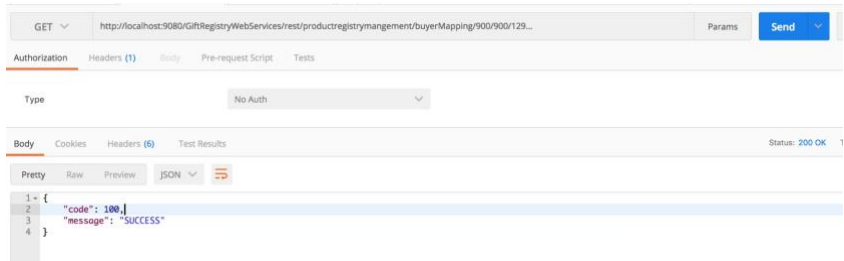
Type No Auth

Body Cookies Headers (6) Test Results Status: 200 OK

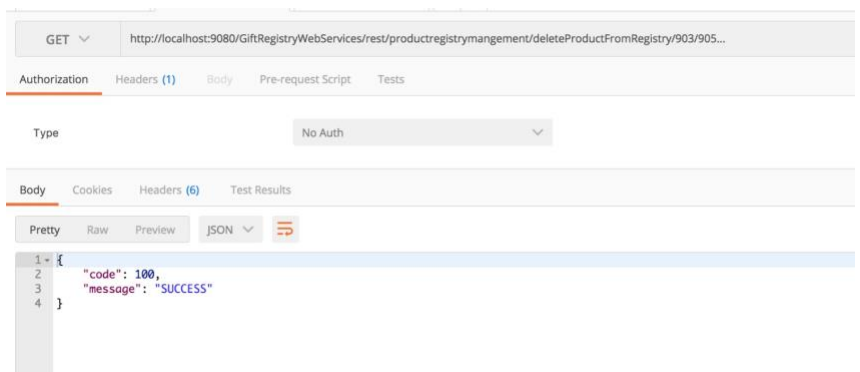
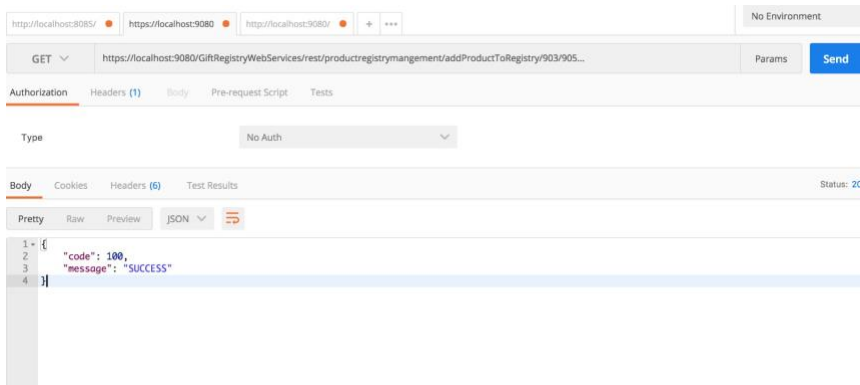
Pretty Raw Preview JSON

1 {
2   "code": 100,
3   "message": "SUCCESS"
4 }
```

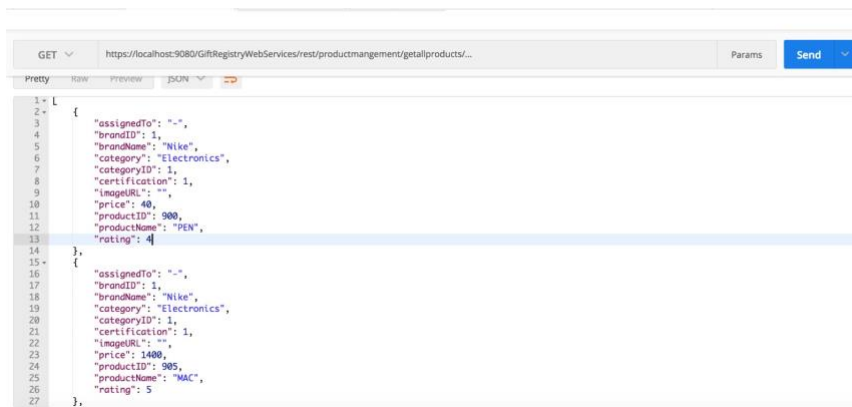
7. Self-assign an item on another user's registry



8. Add/remove items into/from inventory



9. Display the items in the inventory



```
GET https://localhost:5080/GiftRegistryWebServices/rest/productmanagement/getallproducts/... Params Send

1 - L
2 - {
3   "assignedTo": "-",
4   "brandID": 1,
5   "brandName": "Nike",
6   "category": "Electronics",
7   "categoryID": 1,
8   "certification": 1,
9   "imageURL": "-",
10  "price": 40,
11  "productID": 900,
12  "productName": "PEN",
13  "rating": 4
14 },
15 - {
16   "assignedTo": "-",
17   "brandID": 1,
18   "brandName": "Nike",
19   "category": "Electronics",
20   "categoryID": 1,
21   "certification": 1,
22   "imageURL": "-",
23   "price": 1400,
24   "productID": 905,
25   "productName": "MAC",
26   "rating": 5
27 }
```

Description of Web Services and Microservices

Web service 1: DatabaseWebService.

Exchange data with database, manage user info, product info, registry info, sign up, self-assign product, etc.

Microservices:

BuyerProductMappingService

- To assign a product to self from shared registry.

CustomerDetailsService

- To get user Info from user ID.
- To get user Info for the profile Info page.
- To update profile information or password.

ListUsersServiceService

- To get all user Info.

ProductManagementService

- To get all the list of Products
- To add a product.
- To delete a product.
- To get brand.
- To get categories.

RegistryManagementService

- To create a new registry.
- To delete a registry. Make a registry public.
- To get all registries.

RegistryProductMappingService

- To get all registries of the given userID.
- To add product to registry.
- To delete a product from registry.
- To get all products from registry which was added to a registry.

SelfAssignedProductsService

- To get the value of list of products already assigned to self in a particular registry.

UserRegistrationService

- To add new user.
- To check old user.

Web service 2: AppWebService.

Exchange data with DBWebServer, manage user info, product info, registry share, make public, etc.

Microservice:

ProductManagementService

- To get all the list of Products for admin page.

- To add a product. Used in admin page.
- To delete a product, called from admin page.

ProductRegistryMappingService

- To add product to registry.
- To delete a product from registry.
- To get all products from registry which was added to a registry.

RegistryMangementService

- To create a new registry.
- To delete a registry. Make a registry public.
- To get all shared registries.

RegistrySharingService

- To share a registry with user.

UserManagementService

- To authenticate a user.
- To add a new user to DB.
- To get user Info for the profile Info page.
- To update profile information or password.

Web service 3: FrontEndService

Exchange data with AppWebServer, manage user info, product info, registry share, make public, etc.

Microservice:

- Add product service
- Change password service
- Create registry service
- Delete product service
- Delete registry service
- Edit profile service
- Log in service
- List my registries service
- New sign up service
- List products service

Summary of problems

- Faced minor issues only, fixed it with the help on resources from eLearning.
 Problem1: Implement session.
 Solved by:
 Study with the resource provide on e-learning.
 Problem2: Implement HTTPS.
 Solved by:
 Study with the resource provide on e-learning and google for related documents.
 Problem3: Implement filters.
 Solved by:
 Study with Stack Overflow and refer to jQuery document.
 Problem4: Implement TLS/SSL.
 Solved by:
 Study with YouTube tutorial, read related documents and group discussion.