Swift

Backend Interview

Task Description

You will design a simple JAVA web server/application, having a set of 6 REST endpoints, and using MongoDB to store the details. The server should be written in *Light-4j*, a lightweight Java web framework.

Dummy data APIs for your use are available at JSON PLaceholder. You will only use the `/users` endpoint for your code.

Quickstarts and References linked below:

1. Light-4j
2. MongoDB
3. REST API

Rest Endpoints

- Create a Rest service using `light-rest-4j` (either use light-codegen with a simple openapi3 spec **OR** create from scratch)
- No need to use any authentication for `MongoDB`

HTTP VERB	URL	Description
GET	/load	Loads 10 users into DB, should be called automatically on application startup as well, user data should be fetched from [JSON PLaceholder], should always return empty response - with code *200* on success, else an appropriate error code.
DELETE	/users	Deletes all users in the DB
DELETE	/users/:userId	Delete user with *userId* from DB
GET	/users/:userId	Get the specific user with *userId* from DB and return its data in *json* format
PUT	/users	Puts a new user into the DB, the data is sent to the server as the request body of this api, should return appropriate error code if *userId* already exists; See the user data format below.
POST	/users/:userId	Updates an existing user in the DB, the data is sent to the server as the request body of this api, should return appropriate error code if *userId* doesn't exist; See the user data format below.

Certain details (like the response for some endpoints) are deliberately left empty for you to design according to best REST practices. Few things for you to think about:

- Error code and response if a *userId* doesn't exist when using GET/DELETE

- Error code if POST body has different *userId* field that the one it tries to update. Should it even have a *userId* in the POST body ?
- `link` header in response of POST/PUT

User Data Format

```
{
  "id": 1,
  "name": "Leanne Graham",
  "username": "Bret",
  "email": "Sincere@april.biz",
  "address": {
    "street": "Kulas Light",
    "suite": "Apt. 556",
    "city": "Gwenborough",
    "zipcode": "92998-3874",
    "geo": {
      "lat": "-37.3159",
      "lng": "81.1496"
    }
  },
  "phone": "1-770-736-8031 x56442",
  "website": "hildegard.org",
  "company": {
    "name": "Romaguera-Crona",
    "catchPhrase": "Multi-layered client-server neural-net",
    "bs": "harness real-time e-markets"
  }
}
```

Hints

Some resources which may help duirng development:

- [Light-4J Examples Repository](#)
- [Light-4J Cross Cutting Concerns](#)
- [Light-4J Body Parser](#)

Submission Guidelines

- *Please do not upload to any public source code repository like github*
- Please zip up your code and upload it to your google_drive and share its link with us
- The zip should include the Light4J services created in the one project.