Problem 1: Reverse Geocoding Assignment

Design and implement a RESTful web service (HTTP) which is capable of looking up a physical street address given a set of geographic coordinates (longitude and latitude values). For example, given the latitude '33.969601' and longitude '-84.100033', the service should return the address of the NCR office in Duluth, GA (2651 Satellite Blvd, Duluth, GA 30096). The implementation should delegate to an online geocoding API (i.e., Google Maps or similar) to perform the lookup; the implementation will serve as a basic abstraction to simplify usage of one or more external services that handle the geo-location aspects.
Non-Functional Requirements:

- The service should cache (locally) the last 10 lookups and provide an additional RESTful API for retrieving this stored data. The data returned from this API should be a collection of the lookups performed, including the longitude and latitude values, the address found, and the date/time of the lookup.
- The service should handle any error conditions (such as invalid input or internal errors) with suitable HTTP error responses.
- The developer is responsible for designing the API signatures, including the input/output data structures, and any exceptions deemed necessary.
- Although Java is preferred, the choice of language and frameworks is at the discretion of the developer. Ideally, the application will run as a simple process/executable, and not require an external container or web server to run

The source code for the solution should be provided directly as a zip file, or as a link to an online source repository such as Github. In addition to the code, a set of instructions for building and running the application should be provided, along with any notes the developer would like to include which explain the design decisions made.

Solution :

Attached is the solution for the problem listed above. Points for the design decisions are as below :

(1) Here I have used Strategy and Factory pattern along with Singleton Pattern to resolve the different services based on request.

(2) It also uses the Spring Boot generate executable Spring Application using inbuilt tomcat.

(3) It uses the Google and Nominatim APIs to look-up the for the Reverse Geo coding address based on the coordinates provided.

(4) It also implements Custom Exception handling to return HTTP error codes at runtime based on various failures or exceptions.

(5) It contains the a sample Mock test case and method to show how a test can be implemented for such services. However, as requirement was not mentioned for generating test cases, I have not create extensive test cases.

(6) It uses Maven to build and maintain dependencies.

(7) It also contains Java Doc for documenting all the classes and methods.

(8) Attached Compressed file contains the HTML document for classes and methods (Java Doc – index.html)

How to Execute :

Attached Compressed file is the solution for the problem listed above. Points for the executing program.

(1) Attached compressed file contains “Target” folder.

(2) inside that folder kindly execute the jar file “ReverseGeoLocation-0.1.0.jar” using command : Java -jar ReverseGeoLocation-0.1.0.jar

(3) kindly open the web browser and execute the requested REST services as shown in below document in Sample Request.

REST Method Details :

Get Reverse Geo Location

This method accepts as Latitude, Longitude and Service type as String. Here, Service can be passed as either “google” or Nominatim.

- **URL**

  `/getLocation`

- **Method:**

  Default Method.

- **URL Params**

  Required:

  lat : as Latitude, it's a Mandatory field.

  log : as Longitude, it's a Mandatory field.

  Optional:

  service : The name of the service either google or nominatim, it's an optional parameter if nothing specified google Geo services will be used.

- **Success Response:**

  - **Code:** 200
    Content: {"address":" ","serviceused":" ","longitude":" ","latitude":" ","lookupdatetime":" ","city":" ","country":" "}

- **Error Response:**

  The Error Response based on the HTTP Errors.

  Code: 503
  Content: {"errorCode":503,"message":"Not Able to Connect Host or Service Unavailable"}

  Code:412
  Content: {"errorCode":412,"message":"Invalid Service Name"}

Code:422
Content: {"errorCode":422,"message":"Invalid Coordinates"}

- **Sample Call:**

http://localhost:8080/getLocation?lat=33.969601&log=-84.100033&service=google

- **Sample Response:**

```
{"address":"2651 Satellite Blvd, Duluth, GA 30096,
USA","serviceused":"google","longitude":"-
84.100033","latitude":"33.969601","lookupdatetime":"2016-12-04
20:49:35","city":"Duluth","country":"United States"}
```

Get All cached Locations.

This is the method which is used to retrieve all the last 10 requested and cached locations.

- **URL**

  `/getCachedLocations`

- **Method:**

  Default Method.

- **URL Params**

  No Parameters for this method.

  Required:

  N/A

  Optional:

  N/A

- **Success Response:**

  This will return the last 10 cached Repones.

  - **Code:** 200
    Content: [{"address":" ","serviceused":" ","longitude":" ","latitude":" ","lookupdatetime":" ","city":" ","country":" "},{"address":" ","serviceused":" ","longitude":" ","latitude":" ","lookupdatetime":" ","city":" ","country":" "}]

- **Error Response:**

  This will return HTTP code and messages.

    Code: 503
    Content: {"errorCode":503,"message":"Not Able to Connect Host or Service Unavailable"}

•**Sample Call:**

http://localhost:8080/getCachedLocations

•**Sample Output:**

```
[{"address":"2413-2497 Tolbert Town Rd, Rockmart, GA 30153,
USA","serviceused":"google","longitude":"-
85.100033","latitude":"33.969601","lookupdatetime":"2016-12-04
20:48:11","city":"Rockmart","country":"United States"},{"address":"2651
Satellite Blvd, Duluth, GA 30096, USA","serviceused":"google","longitude":"-
84.100033","latitude":"33.969601","lookupdatetime":"2016-12-04
20:49:35","city":"Duluth","country":"United States"},{"address":"2413-2497
Tolbert Town Rd, Rockmart, GA 30153,
USA","serviceused":"google","longitude":"-
85.100033","latitude":"33.969601","lookupdatetime":"2016-12-04
20:52:51","city":"Rockmart","country":"United States"},{"address":"Co Rd
71, Bremen, AL 35033, USA","serviceused":"google","longitude":"-
87.100033","latitude":"33.969601","lookupdatetime":"2016-12-04
20:52:56","city":"Bremen","country":"United States"},{"address":"11700-
12244 Little Coffman Rd, Lester, AL 35647,
USA","serviceused":"google","longitude":"-
87.100033","latitude":"34.969601","lookupdatetime":"2016-12-04
20:53:03","city":"Lester","country":"United States"},{"address":"Little
Coffman Road, Gourdsville, Limestone County, Alabama, 35647, United
States of America, us","serviceused":"Nominatim","longitude":"-
87.100033","latitude":"34.969601","lookupdatetime":"2016-12-04
20:53:09","city":"Limestone County","country":"United States of America"},
{"address":"Lauderdale County 45, Lauderdale County, Alabama, United
States of America, us","serviceused":"Nominatim","longitude":"-
88.100033","latitude":"34.969601","lookupdatetime":"2016-12-04
20:53:16","city":"Lauderdale County","country":"United States of America"},
{"address":"Ontario, Canada, ca","serviceused":"Nominatim","longitude":"-
88.100033","latitude":"54.969601","lookupdatetime":"2016-12-04
20:53:22","city":null,"country":"Canada"},{"address":"Ontario P0V,
Canada","serviceused":"google","longitude":"-
88.100033","latitude":"54.969601","lookupdatetime":"2016-12-04
20:53:33","city":null,"country":"Canada"},{"address":"Ontario P0V,
Canada","serviceused":"google","longitude":"-
87.100033","latitude":"54.969601","lookupdatetime":"2016-12-04
20:53:39","city":null,"country":"Canada"}]
```