**Interview – Technical Assessment (Intermediate Java Developer – Backend)**

**Your Requirement:**

|  |  |
| --- | --- |
|  | If you are unable to complete all the requirements, prioritize the most important core functionality and work on those.  Good Luck! |

1. Create a repository, Commit your code to a GitHub repository (<https://github.com/>). Every subsequent commit should be pushed to this repository. Invite the following participants who will review your code:

* rammohan@tataelxsi.co.in
* sethum@tataelxsi.co.in

1. The REST Service must be built using Spring Boot Framework and use the Spring Initializer to create the Project.
2. The root of the resource should be @RequestMapping(path = **"/customer"**)
3. You are required to build a REST Service must perform Crud operations for the following:

* INSERT a customer record into database

@PutMapping(**"/"**)

* UPDATE a customer Record into database

@PostMapping(path = **"/"**, consumes = **"application/json"**, produces = **"application/json"**)

* DELETE a customer Record from Database

@DeleteMapping(path = **"/{customernumber}"**, produces = **"application/json"**)

* GET a customer record by Customer Number

@GetMapping(path = **"/{customernumber}"**, produces = **"application/json"**)

* GET ALL Customers from database

@GetMapping(path = **"/"**, produces = **"application/json"**)

All data must be persisted to a local instance of MongoDb, no Authentication required:

**spring.data.mongodb.uri**=**mongodb://localhost:27017/test-assessment**

1. All Requests to the REST service must validate the Authorization Header of the HTTP Request, which will contain BASIC Authorization Header information for:

* Username: technical
* Password: Assessment

If any request does not pass the Authorization Header validation, a response of HTTP 401 must be returned.

1. Cater for a “status” field for each Customer Record. If a GET Customer Record By Customer

Number finds a customer that has a status of false, then return a response of HTTP 401.

1. Write Unit tests and mock any calls to your database. Ensure all Rest Operations are covered.
2. Cache you Customer Records to help the HTTP GET REST Calls. Preferred Caching to use is REDIS or Caffeine
3. Provide some notes around the planning and technical decisions made during the development of this service. Also, any ideas or features that could be added to this solution.

Your REST service should follow a Structure like the following (use this a guide, you don’t have to build your solution exactly like this):

