For your test, we'd like you to put together a dataset search service to demonstrate knowledge in how an end-to-end entity indexing system operates and show how you approach the process of taking requirements and building production-ready code to implement them.

All of our primary records for user-owned entities (datasets, for example) are stored in DynamoDB on AWS. Dynamo publishes an event stream (similar to a Kafka stream, or Kinesis if you're familiar with AWS) containing a JSON representation of each change operation executed against a table. We want to design a search index over the entities defined in one of these tables, kept up-to-date by subscribing to this update stream, and a small web service to execute queries against that index.

As input, we are sending you a JSON file that contains a sample set of update messages such as you might get by subscribing to the AWS DynamoDB update stream. You should:

- Implement an indexer job that will run against this stream. This should be a bit of code in whatever language/framework you prefer (we use Java for these types of jobs) that could be run on AWS Lambda, which would receive the events one at a time from the stream. To test this, you could actually set up a Kinesis or Kafka stream and feed the source events into it - or you could create a simple test "harness" that parses the sample JSON and feeds the individual events to your indexer - the key is that the indexer code would be deployable as a stream listener using a serverless platform like Lambda.
- Design the elastic search index to be suitable for answering a range of queries about datasets - for example, to find matching datasets based on free text searching in the headline, name, abstract, etc. of the dataset.
- Build a small web API to wrap the elastic search index and provide API methods for searching. you can use any framework you like - we use DropWizard for building REST APIs in java, but any framework that you're experienced with is fine.

We'll be looking at your approach - what engineering decisions did you make as you built the project out? Think about this project as though it's going to become production code - how did you test it, how did you deploy the various components? Given the short amount of time you'll spend on this project, what do you prioritize and how do you think about future expansion?