**Summary**

Build a REST-API for requesting metadata of a music track from an external service storing it in a database.

The metadata should be fetched via the API of a public streaming services – for this example try the Spotify API.

Additional to the track metadata, use the API to retrieve and store the related cover image.

The REST-API built by you should get secured with HTTP basic authentication (username & password).

**Framework Requirements:**

Spring Boot: This is the only hard requirement on the technology used. Everything else is your decision. You will find some suggestions here which might serve as hints for you about what could be used.

In case you’re not sure what to choose, feel free to reach out for some directions.

**Code Challenge API Details:**

* Write access: There should be a single endpoint to trigger the creation of a track, which takes a single value “ISRC”
  + Sample: POST http://localhost:8080/codechallenge/createTrack?isrc=USMC18620549
  + Input is just the ISRC
    - An ISRC is more or less an identifier of a music track, see “ Spotify API” section below
  + As soon as the call comes in, use the Spotify API to fetch the following metadata:
    - name
    - artistName
    - albumName
    - albumId
    - isExplicit
    - playbackSeconds
  + In case the Spotify API returns multiple tracks, it is enough to handle/store the first one found.
  + Store the ISRC and the additional metadata into the DB
    - No need to care about updating an already existing ISRC, skipping or giving back an error is enough.
  + Use the track’s ‘albumId’ to perform another Spotify API call to download and store the cover image
* Read access: There should be 2 endpoints to retrieve previously stored data identified by the “ISRC”:
  + single result containing the previously stored metadata
    - Sample metadata: GET [http://localhost:8080/codechallenge/getTrackMetadata?isrc=USMC18620549](http://localhost:8080/codechallenge/getTrackMetadata?isrc=USMC18620549%20)
  + single result, downloading the previously stored cover image
    - Sample cover: GET [http://localhost:8080/codechallenge/getCover?isrc=USMC18620549](http://localhost:8080/codechallenge/getCover?isrc=USMC18620549%20)

Spotify API Details:

* You should use this API to retrieve the metadata of a music track which is identified by an ISRC.
* The metadata of a music track contains an album field which includes an id that identifies which album the track is on. Use the id to perform an additional call to store the cover image.
* For calling the Spotify API, you need an access token. You can obtain an access token by following the authorization guide provided at [Spotify Developer](https://developer.spotify.com/documentation/general/guides/authorization/). For this coding challenge, you may use the Client Credentials Flow to get an access token.
* Spotify API Endpoints you might use:
  + https://api.spotify.com/v1/search?q=isrc:USMC18620549&type=track
    - Get track data by ISRC
    - This is the endpoint you will call to get the metadata which is written to the DB.
  + https://api.spotify.com/v1/albums/{id}
    - This is the endpoint to retrieve the album details, including the cover image. Replace {id} with the album ID obtained from the track metadata. The album object will include an array of images. You can select the appropriate resolution for your needs, such as 500x500 pixels.

**Details, Suggestions etc.:**

* During any part of this challenge, from basic research on coding patterns and techniques up to and including code generation, do not use ChatGPT or any other LLM (Large Language Model) that has been trained on copyrighted material for which the copyright owner has not been appropriately compensated.  UMG does not allow our internal Developers to use such models, therefore you should not as part of this challenge.
* Choose a Database you are most productive with, MSSQL, MySQL, PostgreSQL etc.
* The technology used to access the DB from Java is up to you. Plain JDBC, Spring JDBC Templates, Hibernate/JPA, MyBatis, JOOQ...
* How to store the cover is up to you. Could be stored to file system, to DB, etc.
* The design of the REST API is up to you (GET/POST/PUT, query parameters, naming, hierarchy etc.). Above you’ll find suggestions, but feel free to use something else.
* You don’t need to create JUnit tests. In case you think testcases make your coding process more efficient, it’s fine to have some, but don’t feel obliged to add them.
* Of course, we want to see the API in action ;)
  + Use the client of your choice (Postman, Curl, Httpie etc.) to demo it
  + Generating a Swagger / OpenAPI page for testing would be a plus.