Test plan for API Route

Introduction

The purpose of this test plan is to verify the functionality of the
GET /v4/catalog/artists/ API route.
The test plan outlines a comprehensive API route testing strategy, covering positive and negative (&edge) scenarios. This route returns a list of artists in the Beatport catalog, filtered by various parameters.

Scope

This plan focuses on the functionality of the API route, including status codes, response format, and data returned. It does not include security testing or performance testing.

- Positive Tests: Verify the API with valid inputs.
- Negative Tests: Verify the API with invalid or edge case inputs.

Test Objectives

- Validate that the API returns the expected list of artists based on the query parameters;
- Ensure that the API handles edge cases and erroneous inputs gracefully;
- Verify the structure and data types of the returned JSON response;

Test cases

Positive tests:

1. Get artists with "No Parameters"
- Description: Send a GET request to the base URL without any parameters.
- Expected Result: HTTP 200 OK; Contains "Count" property with the total number of artists (may be higher than returned results). Contains "Results" array with artist objects. Each artist object has all expected properties (created, id, image, name, slug, updated, URL).

2. Valid name filter -> Send a GET request with the "name" parameter containing a search term.
- Description: Filter artists by a valid name containing the specified string.
- Input: name=kobe
- Expected Result: HTTP 200 OK, results contain artists with names containing "kobe".

3. Exact name match -> Send a GET request with the "name_exact" parameter containing an artist's name
- Description: Filter artists by an exact name match.
- Input: name_exact=Kobe
- Expected Result: HTTP 200 OK, results contain artists with the exact name "Kobe".

4. Created date filter (Exact)
- Description: Filter artists by an exact created date.
- Input: created=2023-03-23
- Expected Result: HTTP 200 OK, results contain artists created on 2023-04-24

5. Created date filter (Range)
- Description: Filter artists created within a date range.
- Input: created=2023-01-01:2023-03-31
- Expected Result: HTTP 200 OK, results contain artists created between 2023-01-01 and 2023-03-31.

6. Pagination(get artists with pagination) -> Send a GET request with "page" and "per_page" parameters to retrieve a specific page of results.
- Input: page=2&per_page=10
- Expected Result: HTTP 200 OK, results contain the second page of artists, with 10 artists per page (contains "Count" property with the total number of artists; contains "Next" and/or "Previous" links for pagination (if applicable); contains "Results" array with the requested number of artist objects for the specified page.)

Negative tests:

7. Invalid URL
- Description: Send a GET request to a non-existent URL endpoint.
- Expected Result: Status code: 404 Not Found

8. Invalid name filter
- Description: Filter artists by an invalid name.
- Input: name=
- Expected Result: HTTP 400 Bad Request, an error message indicating an invalid name parameter.

9. Invalid date format -> Send a GET request with the "created" or "updated" parameter in an invalid date format.
- Description: Filter artists by an incorrectly formatted date.
- Input: created=2023-13-01
- Expected Result: HTTP 400 Bad Request, an error message indicating invalid date format.

10. Invalid page number
- Description: Use a negative page number.
- Input: page=-1
- Expected Result: HTTP 400 Bad Request, an error message indicating invalid page number.

11. Excessive page size -> Send a GET request with a very high "page" parameter value for pagination.
- Description: Use an excessively high value for per_page.
- Input: per_page=1000
- Expected Result: HTTP 400 Bad Request, an error message indicating invalid page size.

12. Non-existent Artist ID
- Description: Send a GET request with the "id" parameter containing a non-existent artist ID.
- Expected Result: HTTP 404 Not Found (or 200 OK (possible)), Empty "Results" array, or a specific error message indicating the artist is not found.

13. Invalid enabled filter
- Description: Filter artists by an invalid enabled value.
- Input: enabled=maybe
- Expected Result: HTTP 400 Bad Request, an error message indicating invalid enabled parameter.

Edge case tests:

14. Empty search results
- Description: Send a GET request with a search parameter (name or name_exact) that matches no artists.
- Expected Result: HTTP 200 OK; "Count" property may be greater than zero (total artists); Empty "Results" array indicating no matching artists found.

15. Special characters in search
- Description: Send a GET request with the "name" or "name_exact" parameter containing special characters or symbols.
- Expected Result: HTTP 200 OK; "Results" array may contain matching artists based on how the API handles special characters. Empty the "Results" array if the special characters prevent a proper search.

16. Empty string for search parameters
- Description: Send a GET request with the "name" or "name_exact" parameter set to an empty string.
- Expected Result: HTTP 200 OK (possible) or 400 Bad Request (may return all artists (depending on API implementation). An error message indicating an empty search parameter (if applicable).

17. Date filter edge cases
- Description: Send a GET request with the "created" or "updated" parameter using various edge cases for date formats.
- Examples: future dates, dates outside a supported range, single-digit year formats.
- Expected Result: HTTP 200 OK (possible) or 400 Bad Request; Response body: Empty "Results" array if the date filter excludes all artists; Error message indicating the invalid date format (if applicable).

18. Combining multiple filters
- Description: Send a GET request with a combination of filters (name, created, enabled, etc.) including edge cases.
- Examples: combining a name search with a date range filter exceeding available data.
- Expected Result: HTTP 200 OK; Response body: "Results" array reflecting the intersection of applied filters, potentially empty. Error message if the combination of filters is invalid (if applicable).

Tools and Techniques

- Automated testing tools like Postman can be used to send requests and validate responses.
- Unit testing frameworks can be used to write tests for specific functionalities.

Pass/Fail Criteria

- A test case passes if the actual result matches the expected result for each scenario.
- Any deviation from the expected behavior is considered a failure.

Reporting

- Test results should be documented and reported clearly, including details of passed and failed test cases.

- Bugs identified during testing should be logged and communicated to the development team.

<u>**References:**</u> API Documentation ***Route*** **GET /v4/catalog/artists/**

Artists are the producers or performers of music in the Beatport catalog. This route returns a list of artists whose music is in the Beatport catalog.
Parameters
Name string (query)
Filter name by case-insensitive text containment.
Name_exact string (query)
Filter by name exact match.
Created string (query)
Filter by exact, less/greater than equal, and range. Supports slice syntax: `date=1970-01-01` (exact) `date=:1971-01-01` (less than equal) `date=1970-01-01:` (greater than equal) `date=1970-01-01:1971-01-01` (range)
Updated string (query)
Filter by exact, less/greater than equal, and range. Supports slice syntax: `date=1970-01-01` (exact) `date=:1971-01-01` (less than equal) `date=1970-01-01:` (greater than equal) `date=1970-01-01:1971-01-01` (range)
Id number (query)
Filter by artist ID exact match. Supports `OR` lookup: `param=value1,value2`
Enabled string (query)
Filter by enabled.
Page integer (query)
A page number within the paginated result set.
Per_page integer (query)
The number of results to return per page.
Response
A successful request returns the HTTP 200 OK status code and a JSON response body with the following properties.
Count integer
Next string | $uri
Previous string | $uri
Results array
The "Results" array contains artist objects with the following structure

```
{
"created": "2023-03-21T15:15:48-06:00",
"id": 1095995,
"image": {
"id": 5539565,
"Uri":
"https://geo-stage-media.beatport.com/image_size/590x404/0dc61986-bccf-49d4-8fa
d-6b147ea8f327.jpg",
"Dynamic_uri":
"https://geo-stage-media.beatport.com/image_size/{w}x{h}/0dc61986-bccf-49d4-8fa
d-6b147ea8f327.jpg"
},
"name": "Kobe",
"slug": "kobe",
"updated": "2023-03-21T15:21:23-06:00",
"website": null,
"url": "https://api.beatportstage.com/v4/catalog/artists/1095995/",
"dj_association": null
}
```