

POSTMAN REQUESTS

IDM	2
1.1 Register user with invalid password	2
1.2 Register valid user	3
1.3 Successfully login	3
1.4 Unsuccessfully login	4
1.5 Authorize → expired token	4
1.6 Authorize → valid token	5
1.7 Admin can update user roles (token for admin)	5
1.8 Any other user cannot update user roles	6
1.9 Admin can also remove user roles	6
1.10 Admin can remove users	7
1.11 Get user info	8
1.12 Logout	8
1.13 Update user password	9
2. SONGS_ARTISTS	11
2.1 Get all artists(pageable)	11
2.2 Get artist by UUID	12
2.3 Get artist by UUID -> NOT FOUND	12
2.4 Get all songs - with valid query params	13
2.5 Get all songs with invalid query params	13
2.6 Get songs for an artist	14
2.7 Create artist without auth header	14
2.7 Create artist without being content manager	15
2.8 Create artist being content manager	15
2.9 Replace artist	16
2.10 Delete artist	16
2.11 Delete nonexistent artist	16
2.12 Add new song	17
2.13 Assign song to artist	18
2.14 Delete album	19
2.15 Delete song	19
2.16 Update song	20
2.17 Invalid song update	20
3. PLAYLISTS	20
3.1 Create playlist	21

3.2 Add song to created playlist	21
3.3 Get playlist by id	22
3.4 Delete playlist	22
4. API GATEWAY	23
4.1 Login	23
4.2 Invalid login	23
4.3 Register	23
4.4 Logout	24

IDM

1.1 Register user with invalid password

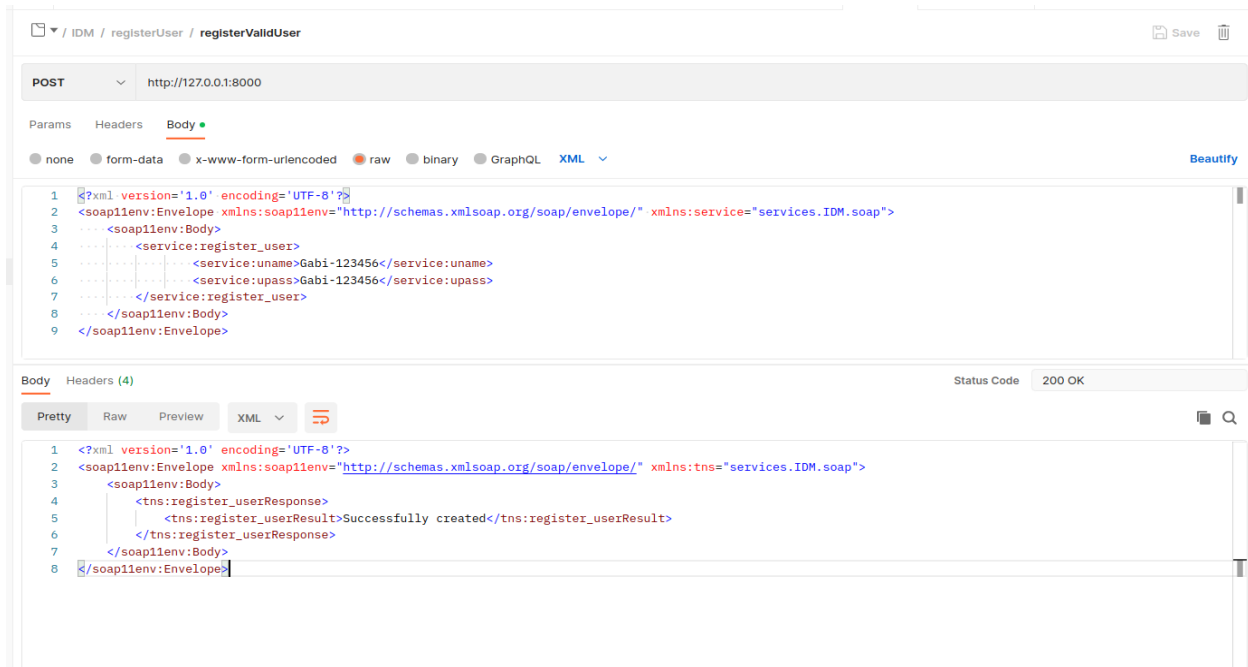
The screenshot displays a REST client interface for a POST request to `http://127.0.0.1:8000`. The request is a SOAP message with the following XML body:

```
<?xml version='1.0' encoding='UTF-8'?>
<soap11env:Envelope xmlns:soap11env="http://schemas.xmlsoap.org/soap/envelope/" xmlns:service="services.IDM.soap">
  <soap11env:Body>
    <service:register_user>
      <service:uname>Ana2</service:uname>
      <service:upass>Ana</service:upass>
    </service:register_user>
  </soap11env:Body>
</soap11env:Envelope>
```

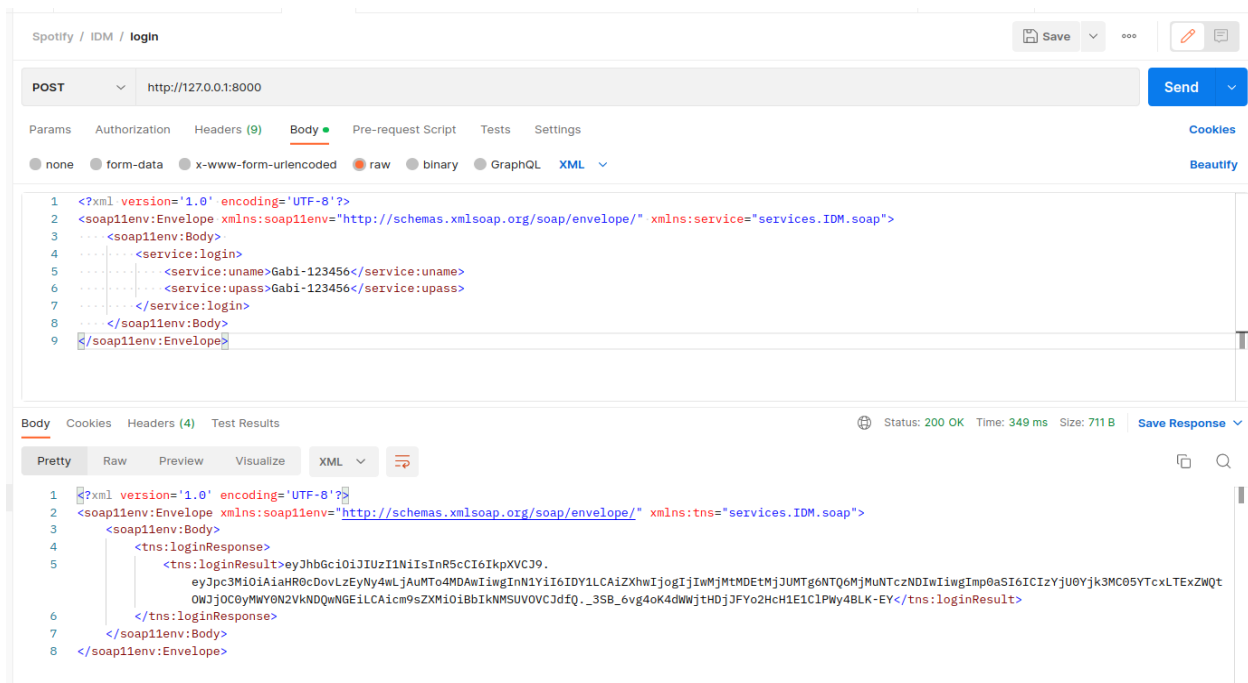
The response, shown in "Pretty" format, indicates a "500 Internal Server Error" and contains the following XML body:

```
<?xml version='1.0' encoding='UTF-8'?>
<soap11env:Envelope xmlns:soap11env="http://schemas.xmlsoap.org/soap/envelope/">
  <soap11env:Body>
    <soap11env:Fault>
      <faultcode>soap11env:Client</faultcode>
      <faultstring>Too weak password</faultstring>
      <faultactor></faultactor>
    </soap11env:Fault>
  </soap11env:Body>
</soap11env:Envelope>
```

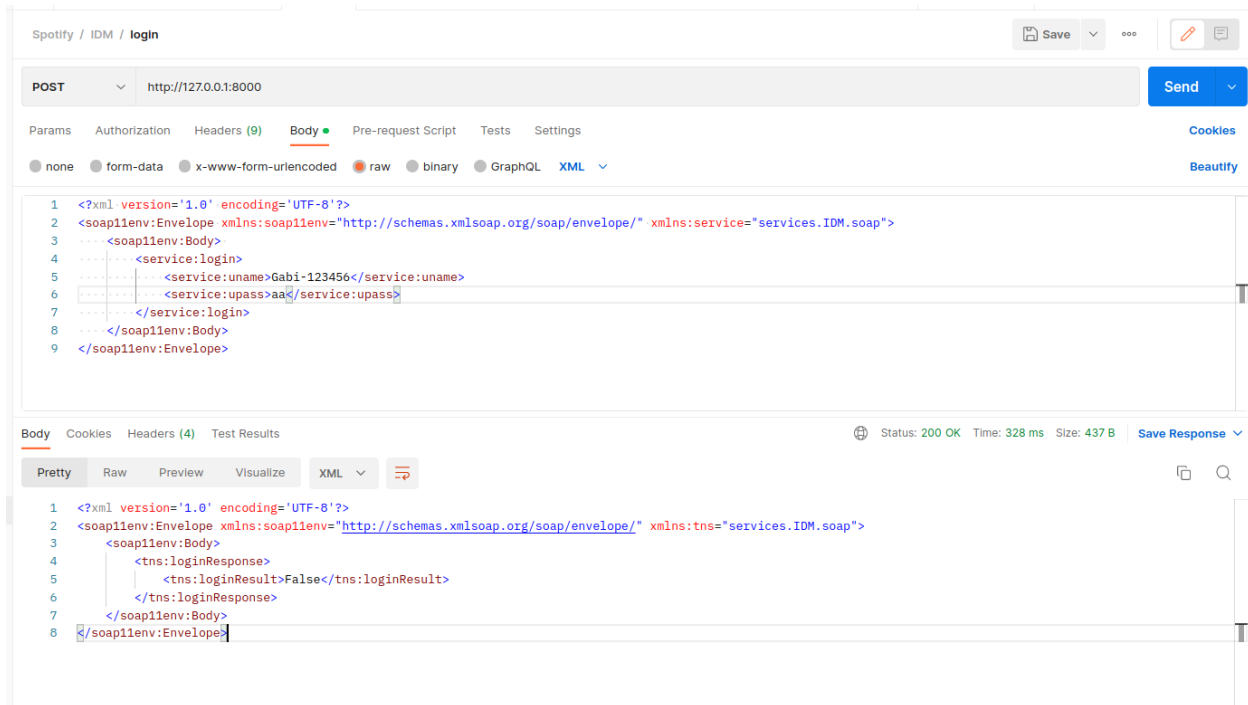
1.2 Register valid user



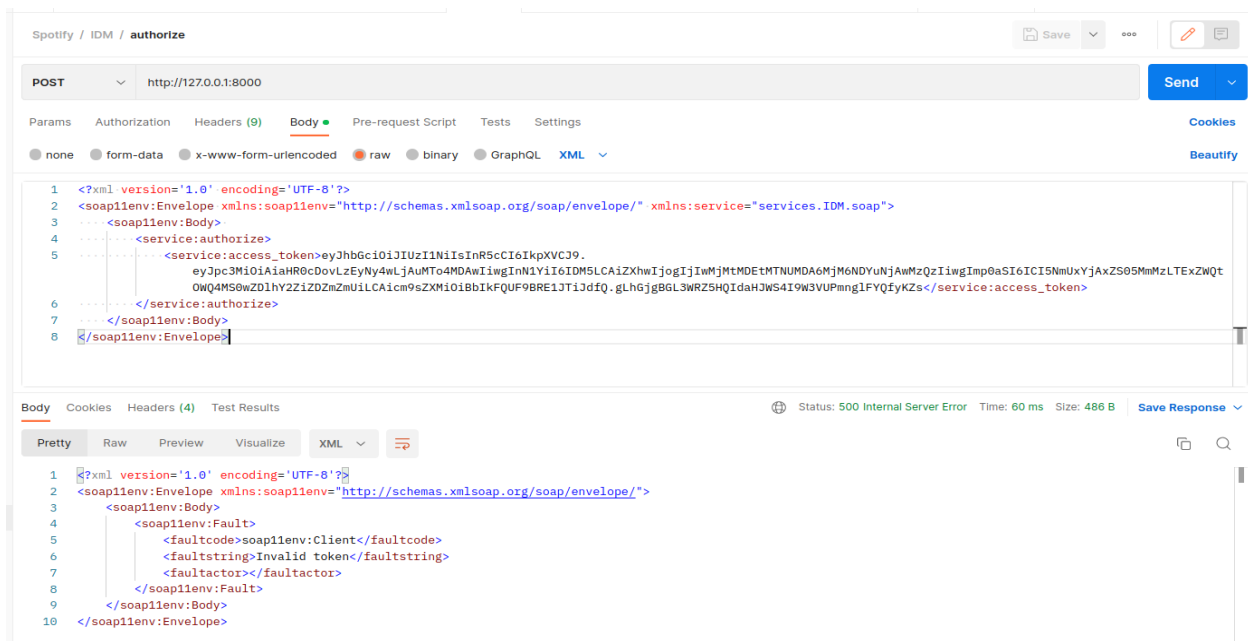
1.3 Successfully login



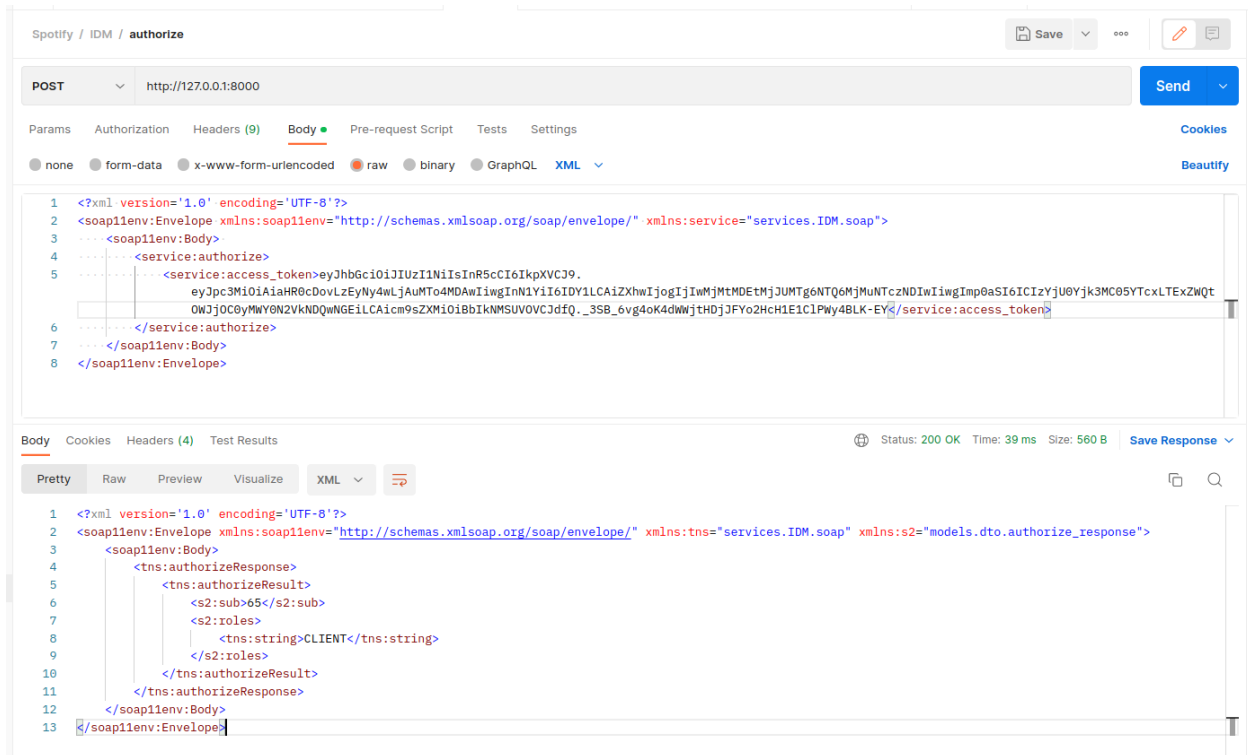
1.4 Unsuccessfully login



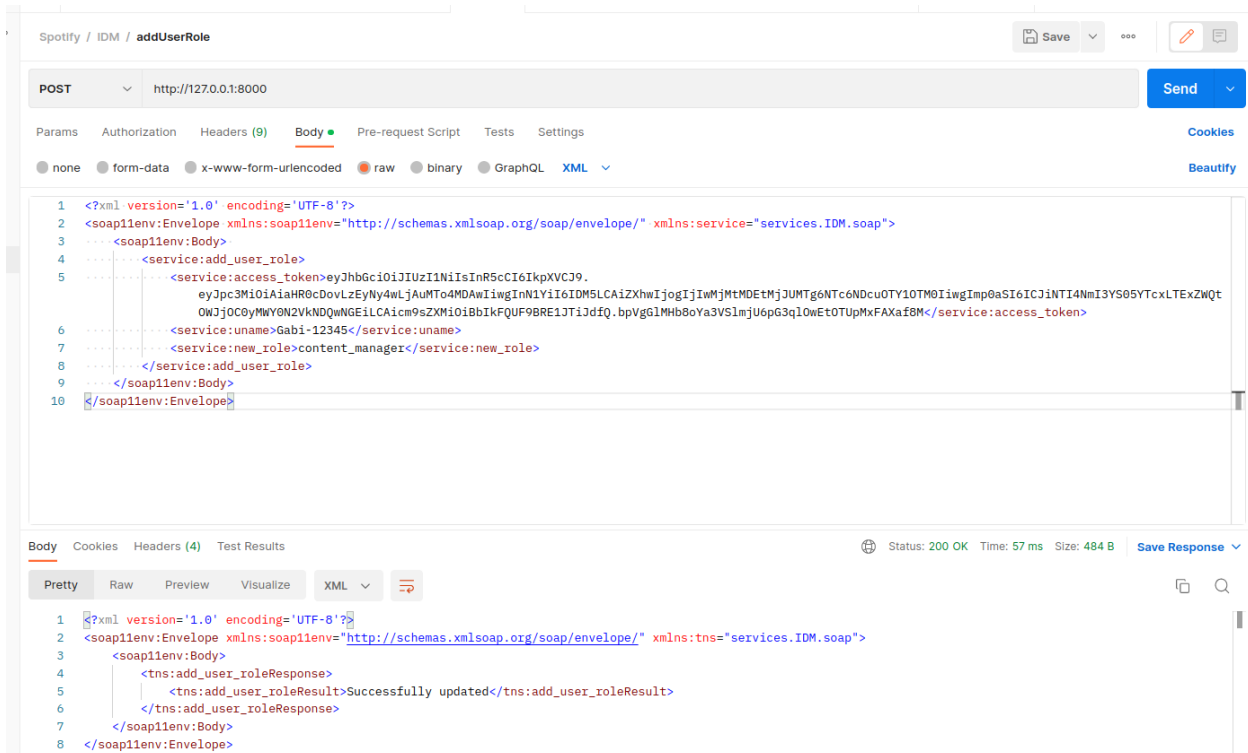
1.5 Authorize → expired token



1.6 Authorize → valid token



1.7 Admin can update user roles (token for admin)



1.8 Any other user cannot update user roles

The screenshot shows a REST client interface for a POST request to `http://127.0.0.1:8000`. The request body is an XML SOAP message intended to add a user role. The response status is **500 Internal Server Error** with a time of 44 ms and a size of 482 B. The response body is an XML SOAP fault message indicating a 'Forbidden' error.

```
1 <?xml version='1.0' encoding='UTF-8'?>
2 <soap11env:Envelope xmlns:soap11env="http://schemas.xmlsoap.org/soap/envelope/" xmlns:service="services.IDM.soap">
3   <soap11env:Body>
4     <service:add_user_role>
5       <service:access_token>eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ3Zjpc3MiOiAiHR0cDovLzEyNy4wLjAuMTQ0MDAwIiwiaWF0IjE6IDM4LCAiZXhwIjogIjIwMjMtMDEtMjUUMTk6MDI6NTYyNzU0NTY1IiwiaWp0aSI6ICI2ZDM1ZDNmNi05YTcyLTExZWQtOWJjOCByMmY0N2VkdQwNGEiLCJmcm9sZXMiOiBbIkNPTlRFTlRlTUFOQudFUiIsICJBUiRlUjU1LCAiQ0xJRUSU1I19.1Bf6c6-SBRBxEskxDJ118XXJIBKWWY1N5rGykaC8BmI</
6       <service:access_token>
7       <service:username>Gabi-12345</service:username>
8       <service:new_role>artist</service:new_role>
9     </service:add_user_role>
10  </soap11env:Body>
11 </soap11env:Envelope>
```

```
1 <?xml version='1.0' encoding='UTF-8'?>
2 <soap11env:Envelope xmlns:soap11env="http://schemas.xmlsoap.org/soap/envelope/">
3   <soap11env:Body>
4     <soap11env:Fault>
5       <faultcode>soap11env:Client</faultcode>
6       <faultstring>Forbidden</faultstring>
7       <faultactor></faultactor>
8     </soap11env:Fault>
9   </soap11env:Body>
10 </soap11env:Envelope>
```

1.9 Admin can also remove user roles

The screenshot shows a REST client interface for a POST request to `http://127.0.0.1:8000`. The request body is an XML SOAP message intended to remove a user role. The response status is **200 OK** with a time of 64 ms and a size of 496 B. The response body is an XML SOAP message indicating successful deletion.

```
1 <?xml version='1.0' encoding='UTF-8'?>
2 <soap11env:Envelope xmlns:soap11env="http://schemas.xmlsoap.org/soap/envelope/" xmlns:service="services.IDM.soap">
3   <soap11env:Body>
4     <service:remove_user_role>
5       <service:access_token>eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ3Zjpc3MiOiAiHR0cDovLzEyNy4wLjAuMTQ0MDAwIiwiaWF0IjE6IDM5LCAiZXhwIjogIjIwMjMtMDEtMjUUMTk6NTc6NDcuOTY1OTM0IiwiaWp0aSI6ICI3IjI0NTI4NmIzYS05YTcyLTExZWQtOWJjOCByMmY0N2VkdQwNGEiLCJmcm9sZXMiOiBbIkFQUF9BRE1JTjJldjQ.1bVgG1Mhb8oYa3VslmjU6pG3q10wEtUUpMxFAf8M</service:access_token>
6       <service:username>Gabi-12345</service:username>
7       <service:removed_role>content_manager</service:removed_role>
8     </service:remove_user_role>
9   </soap11env:Body>
10 </soap11env:Envelope>
```

```
1 <?xml version='1.0' encoding='UTF-8'?>
2 <soap11env:Envelope xmlns:soap11env="http://schemas.xmlsoap.org/soap/envelope/" xmlns:tns="services.IDM.soap">
3   <soap11env:Body>
4     <tns:remove_user_roleResponse>
5       <tns:remove_user_roleResult>Successfully deleted</tns:remove_user_roleResult>
6     </tns:remove_user_roleResponse>
7   </soap11env:Body>
8 </soap11env:Envelope>
```

1.10 Admin can remove users

The screenshot displays a REST client interface for a POST request to `http://127.0.0.1:8000`. The request is a SOAP message with the following XML body:

```
1 <?xml version='1.0' encoding='UTF-8'?>
2 <soap11env:Envelope xmlns:soap11env="http://schemas.xmlsoap.org/soap/envelope/" xmlns:service="services.IDM.soap">
3   <soap11env:Body>
4     <service:remove_user>
5       <service:access_token>eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
6         eyJpc3MiOiAiaHR0cDovLzEyNy4wLjAuMTc0MDAwIiwiaWF0IjE6IDM5LCAiZXhwIjogIjEwMjM0MDU0OTY1OTM0IiwiaWp0aSI6IChJNTI4NmI3YS05YVt0c0w0WjJ
7         OC0yMmY0N2VkdQwNGEiLCAicm9sZXMiOiBbIkFQUF9BRE1JTjJdIiwiaWF0IjE6IDM5LCAiZXhwIjogIjEwMjM0MDU0OTY1OTM0IiwiaWp0aSI6IChJNTI4NmI3YS05YVt0c0w0WjJ
8       </service:access_token>
9     </service:remove_user>
10  </soap11env:Body>
11 </soap11env:Envelope>
```

The response shows a status code of `500 Internal Server Error`. The response body is a SOAP fault:

```
1 <?xml version='1.0' encoding='UTF-8'?>
2 <soap11env:Envelope xmlns:soap11env="http://schemas.xmlsoap.org/soap/envelope/">
3   <soap11env:Body>
4     <soap11env:Fault>
5       <faultcode>soap11env:Client</faultcode>
6       <faultstring>This user doesn't exist</faultstring>
7       <faultactor></faultactor>
8     </soap11env:Fault>
9   </soap11env:Body>
10 </soap11env:Envelope>
```

Swagger UI interface showing a POST request to `http://127.0.0.1:8000`. The request body is a SOAP XML payload. The response status is 200 OK, time is 66 ms, and size is 818 B. The response body is also shown in XML format.

Request:

```

3 <...<soap11env:Body>
4 <...<service:get_user_info>
5 <...<service:access_token>eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
  eyJ3Mj0iOiAiaHR0cDovLzEyNy4wLjAuMT04MDAwIiwiaWwiOiI6IDM5LCAiZXRhIjogIiwiaWJmMDEtMTJUMTg0NTc6NDcuOTY1OTM0IiwiaWImp0aSI6ICJlNTI4NmI3YS05YTcxLXEtZWQt
  OWJ3JC0yMyY0N2VkMDQwNGEiLCAicm9sZXN1OiBBIkFQUF9BRE1JT1Jdf0. bpVg6lMhB8oYa3VSlmjU6pG3ql0wEt0TupMxFAxif8M</service:access_token>
6 <...<service:uname>Gabi-12345</service:uname>
7 <...</service:get_user_info>
8 <...</soap11env:Body>
9 </soap11env:Envelope>

```

Response:

```

3 <...<soap11env:Body>
4 <...<tns:get_user_infoResponse>
5 <...<tns:get_user_infoResult>
6 <...<s1:uid>38</s1:uid>
7 <...<s1:uname>Gabi-12345</s1:uname>
8 <...<s1:uroles>
9 <...<s0:RoleDTO>
10 <...<s0:rid>2</s0:rid>
11 <...<s0:iname>CONTENT_MANAGER</s0:iname>
12 <...</s0:RoleDTO>
13 <...<s0:RoleDTO>
14 <...<s0:rid>3</s0:rid>
15 <...<s0:iname>ARTIST</s0:iname>
16 <...</s0:RoleDTO>
17 <...<s0:RoleDTO>
18 <...<s0:rid>4</s0:rid>
19 <...<s0:iname>CLIENT</s0:iname>
20 <...</s0:RoleDTO>
21 <...</s1:uroles>

```

Spotify / IDM / logout

Save

POSThttp://127.0.0.1:8000

Send

ParamsAuthorizationHeaders (9)BodyPre-request ScriptTestsSettings

CookiesBeautify

noneform-datax-www-form-urlencodedrawbinaryGraphQLXML

```
1 <?xml version='1.0' encoding='UTF-8'?>
2 <soap11env:Envelope xmlns:soap11env="http://schemas.xmlsoap.org/soap/envelope/" xmlns:service="services.IDM.soap">
3   <!--<soap11env:Body>--
4   <!--<service:logout>
5     <!--<service:access_token>eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
        eyJpc3MiOiAiaHR0cDovLzEyNy4wLjAuMTQMDAwIiwiaWF0IjE6IDM4LCAiZXhwIjogIjEwMjItMTItMzBUMTK6Mzc6NTIuNzQIMTM3IiwgImp0aSI6ICI3ZjA0NWwiZS04ODY0LTExZWQt
        YTMxNS05OTZhYWZhOdkxNTAiLCaicaic9sXMioiBBikNPt1RFt1RfTUFOQUdFUiIsICJBURJU1QiLCaIQ8XRJU5UIl19.1oIWQzgHPK5be9uPLHv1qpp3phJukP-1rv_zDFMDvw</
        service:access_token>
6   <!--</service:logout>
7   <!--</soap11env:Body>
8 </soap11env:Envelope>
```

BodyCookiesHeaders (4)Test Results

Status: 200 OKTime: 33 msSize: 440 BSave Response

PrettyRawPreviewVisualizeXML

```
1 <?xml version='1.0' encoding='UTF-8'?>
2 <soap11env:Envelope xmlns:soap11env="http://schemas.xmlsoap.org/soap/envelope/" xmlns:tns="services.IDM.soap">
3   <soap11env:Body>
4     <tns:logoutResponse>
5       <tns:logoutResult>true</tns:logoutResult>
6     </tns:logoutResponse>
7   </soap11env:Body>
8 </soap11env:Envelope>
```


Spotify / IDM / updateUserPassword

POST

http://127.0.0.1:8000

Send

Params

Authorization

Headers (9)

Body

Pre-request Script

Tests

Settings

Cookies

Beautify

noneform-datax-www-form-urlencodedrawbinaryGraphQLXML

<?xml version='1.0' encoding='UTF-8'?>
<soap11env:Envelope xmlns:soap11env='http://schemas.xmlsoap.org/soap/envelope/' xmlns:service='services.IDM.soap'>
 <soap11env:Body>
 <service:change_upass>
 <service:access_token>eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
 eyJpc3MiOiAiAHR0cDovLzEyNy4wLjAuMTQ4MDAwIiwiaWF0IjE6IDM4LCAiZXhwIjojIjwzMtMDEtMjUMTkwMDYNTyZlOTU0NTUyZWQt
 OWJjOCcyMWYN2VkNDQmNGEILCAicm9sZXMI01BkIkNPTlRFTlRftUFQudFUiICJBUIRJUIQLCAIQ0xJRUSUll19.zBfc6-SBRBxrEskxD3118XXJIBKWY1N5rGykaC8BmI</
 service:access_token>
 <service:uname>Gabi-12345</service:uname>
 <service:old_pass>Gabi-12345</service:old_pass>
 <service:new_pass>Gabi-123456</service:new_pass>
 </service:change_upass>
 </soap11env:Body>
</soap11env:Envelope>

Status: 200 OKTime: 622 msSize: 489 BSave Response

PrettyRawPreviewVisualizeXML

1<?xml version='1.0' encoding='UTF-8'?>
2<soap11env:Envelope xmlns:soap11env='http://schemas.xmlsoap.org/soap/envelope/' xmlns:tns='services.IDM.soap'>
3 <soap11env:Body>
4 <tns:change_upassResponse>
5 <tns:change_upassResult>Password successfully updated</tns:change_upassResult>
6 </tns:change_upassResponse>
7 </soap11env:Body>
8 </soap11env:Envelope>

Spotify / IDM / updateUserPassword

POST

http://127.0.0.1:8000

Send

Params

Authorization

Headers (9)

Body

Pre-request Script

Tests

Settings

Cookies

Beautify

noneform-datax-www-form-urlencodedrawbinaryGraphQLXML

<?xml version='1.0' encoding='UTF-8'?>
<soap11env:Envelope xmlns:soap11env='http://schemas.xmlsoap.org/soap/envelope/' -xmlns:service='services.IDM.soap'>
 <soap11env:Body>
 <service:change_upass>
 <service:access_token>eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
 eyJpc3MiOiAiAHR0cDovLzEyNy4wLjAuMTQ4MDAwIiwiaWF0IjE6IDM4LCAiZXhwIjojIjwzMtMDEtMjUMTkwMDYNTyZlOTU0NTUyZWQt
 OWJjOCcyMWYN2VkNDQmNGEILCAicm9sZXMI01BkIkNPTlRFTlRftUFQudFUiICJBUIRJUIQLCAIQ0xJRUSUll19.zBfc6-SBRBxrEskxD3118XXJIBKWY1N5rGykaC8BmI</
 service:access_token>
 <service:uname>Gabi-12345</service:uname>
 <service:old_pass>Gabi-123456</service:old_pass>
 <service:new_pass>Gabi-123456</service:new_pass>
 </service:change_upass>
 </soap11env:Body>
</soap11env:Envelope>

Status: 500 Internal Server ErrorTime: 333 msSize: 520 BSave Response

PrettyRawPreviewVisualizeXML

1<?xml version='1.0' encoding='UTF-8'?>
2<soap11env:Envelope xmlns:soap11env='http://schemas.xmlsoap.org/soap/envelope/'>
3 <soap11env:Body>
4 <soap11env:Fault>
5 <faultcode>soap11env:Client</faultcode>
6 <faultstring>New password cannot be the same as old password</faultstring>
7 <faultactor></faultactor>
8 </soap11env:Fault>
9 </soap11env:Body>
10 </soap11env:Envelope>

1.14 Admin can create new content managers

Spotify / IDM / createUser

POST http://127.0.0.1:8000

Params Authorization Headers (9) **Body** Pre-request Script Tests Settings Cookies Beautify

none form-data x-www-form-urlencoded raw binary GraphQL XML

```
1 <?xml version='1.0' encoding='UTF-8'?>
2 <soap11env:Envelope xmlns:soap11env='http://schemas.xmlsoap.org/soap/envelope/' xmlns:service='services.IDM.soap'>
3   <soap11env:Body>
4     <service:create_user>
5       <service:access_token>eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
6         eyJpc3MiOiAiaHR0cDovLzE5Ny4wLjAuMTQ4MDAwIiwiaWF0IjE6IDM5LCAiZXhwIjogIjEwMjMtMDEtMjUUMTQ6MjI6MjQuMjc5NTM3IiwgImp0aSI6ICIyNTFhZWYwZS05YTc1LTExZWQtOWJj
7         OC0yMmY0N2VkdQwNGEiLCAicm9sZXMiOiBbIkFQUF9BRE1JTjJdFQ.FQzfkaoVKNVjh3_xtoa98XCg60aAwSCjfeRzqEOotnI</service:access_token>
8       <service:uname>Ana-1234567</service:uname>
9       <service:upass>Ana-1234567</service:upass>
10      <service:urole>content_manager</service:urole>
11    </service:create_user>
12  </soap11env:Body>
13 </soap11env:Envelope>
```

Body Cookies Headers (4) Test Results Status: 200 OK Time: 339 ms Size: 471 B Save Response

Pretty Raw Preview Visualize XML

```
1 <?xml version='1.0' encoding='UTF-8'?>
2 <soap11env:Envelope xmlns:soap11env='http://schemas.xmlsoap.org/soap/envelope/' xmlns:tns='services.IDM.soap'>
3   <soap11env:Body>
4     <tns:create_userResponse>
5       <tns:create_userResult>68: Ana-1234567</tns:create_userResult>
6     </tns:create_userResponse>
7   </soap11env:Body>
8 </soap11env:Envelope>
```

2. SONGS_ARTISTS

2.1 Get all artists(pageable)

The screenshot displays a REST client interface with a GET request to `http://localhost:8080/api/songcollection/artists?page=0&size=9`. The response is a JSON object with the following structure:

```
1 {
2   "_embedded": {
3     "artistResponseList": [
4       {
5         "uuid": "72dc2e44-2d3a-487b-a166-30548b14a793",
6         "name": "Scorpions",
7         "active": true,
8         "hasSongs": true,
9         "_links": {
10          "self": {
11            "href": "http://localhost:8080/api/songcollection/artists/72dc2e44-2d3a-487b-a166-30548b14a793"
12          },
13          "parent": {
14            "href": "http://localhost:8080/api/songcollection/artists?page, size, name, match]",
15            "templated": true
16          },
17          "songs": {
18            "href": "http://localhost:8080/api/songcollection/artists/72dc2e44-2d3a-487b-a166-30548b14a793/songs"
19          }
20        }
21      },
22      {
23        "uuid": "77a0340b-068d-4bc7-96ed-ed6372f096af",
24        "name": "Bonnie Tyler",
25        "active": true,
26        "hasSongs": true,
27        "_links": {
28          "self": {
29            "href": "http://localhost:8080/api/songcollection/artists/77a0340b-068d-4bc7-96ed-ed6372f096af"
30          },
31          "parent": {
32            "href": "http://localhost:8080/api/songcollection/artists?page, size, name, match]",
33            "templated": true
34          },
35          "songs": {
36            "href": "http://localhost:8080/api/songcollection/artists/77a0340b-068d-4bc7-96ed-ed6372f096af/songs"
37          }
38        }
39      }
40    ]
41  }
42 }
```

The response status is 200 OK, with a time of 58 ms and a size of 3.19 KB. The response is saved and can be viewed in the Body, Cookies, Headers (8), or Test Results tabs. The response is also available in the Pretty, Raw, Preview, Visualize, or JSON tabs.

```
59 },
60 "_links": {
61   "first": {
62     "href": "http://localhost:8080/api/songcollection/artists?page=0&size=3"
63   },
64   "self": {
65     "href": "http://localhost:8080/api/songcollection/artists?page=0&size=3"
66   },
67   "next": {
68     "href": "http://localhost:8080/api/songcollection/artists?page=1&size=3"
69   },
70   "last": {
71     "href": "http://localhost:8080/api/songcollection/artists?page=2&size=3"
72   }
73 },
74 "page": {
75   "size": 3,
76   "totalElements": 7,
77   "totalPages": 3,
78   "number": 0
79 }
80 }
```

2.2 Get artist by UUID

Spotify / SONG_ARTISTS / get one artist by id

GET http://localhost:8080/api/songcollection/artists/9e8e15f6-f8c4-430d-9682-67ae49438545

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION
-----	-------	-------------

Body Cookies Headers (8) Test Results Status: 200 OK Time: 14 ms Size: 688 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "uuid": "9e8e15f6-f8c4-430d-9682-67ae49438545",
3   "name": "Led Zeppelin",
4   "active": false,
5   "hasSongs": true,
6   "_links": {
7     "self": {
8       "href": "http://localhost:8080/api/songcollection/artists/9e8e15f6-f8c4-430d-9682-67ae49438545"
9     },
10    "parent": {
11      "href": "http://localhost:8080/api/songcollection/artists?page,size,name,match",
12      "templated": true
13    },
14    "songs": {
15      "href": "http://localhost:8080/api/songcollection/artists/9e8e15f6-f8c4-430d-9682-67ae49438545/songs"
16    }
17  }
18 }
```

2.3 Get artist by UUID -> NOT FOUND

Spotify / SONG_ARTISTS / get one artist by id

GET http://localhost:8080/api/songcollection/artists/9e8e15f6-f8c4-430d-9682-67ae49438522

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION
-----	-------	-------------

Body Cookies Headers (8) Test Results Status: 404 Not Found Time: 21 ms Size: 365 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "error": "NOT_FOUND",
3   "status": 404,
4   "details": "Could not find artist 9e8e15f6-f8c4-430d-9682-67ae49438522"
5 }
```

2.4 Get all songs - with valid query params

The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** `http://localhost:8080/api/songcollection/songs?page=0&size=6&searchBy=title&searchedValue=Greatest`
- Status:** 200 OK
- Time:** 10 ms
- Size:** 1.07 KB
- Body (JSON):**

```
1 {
2   "_embedded": {
3     "songResponseList": [
4       {
5         "id": 91,
6         "name": "Greatest Hits",
7         "genre": "POP",
8         "year": 1977,
9         "type": "ALBUM",
10        "songs": [
11          {
12            "id": 92,
13            "name": "It's a Heartache",
14            "_links": {
15              "self": {
16                "href": "http://localhost:8080/api/songcollection/songs/92"
17              }
18            }
19          }
20        ],
21        "_links": {
22          "self": {
23            "href": "http://localhost:8080/api/songcollection/songs/91"
24          },
25          "parent": {
26            "href": "http://localhost:8080/api/songcollection/songs?page=0&size=6&searchBy=title&searchedValue=Greatest",
27            "templated": true
28          },
29          "artists": {
30            "href": "http://localhost:8080/api/songcollection/songs/91/artists"
31          },
32          "add to playlist": {
33            "href": "http://localhost:8080/api/playlistscollection/playlists/{playlistId}/songs",
34            "type": "PATCH",
35            "templated": true
36          }
37        }
38      }
39    ]
40  },
41  "_links": {
42    "self": {
43      "href": "http://localhost:8080/api/songcollection/songs?page=0&size=6&searchBy=title&searchedValue=Greatest"
44    }
45  },
46  "page": {
47    "size": 6,
48    "totalElements": 1,
49    "totalPages": 1,
50  }
51 }
```

2.5 Get all songs with invalid query params

The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** `http://localhost:8080/api/songcollection/songs?page=0&size=6&searchBy=t&searchedValue=Greatest`
- Status:** 422 Unprocessable Entity (WebDAV) (RFC 4918)
- Time:** 10 ms
- Size:** 387 B
- Body (JSON):**

```
1 {
2   "error": "UNPROCESSABLE_ENTITY",
3   "status": 422,
4   "details": "getAllSongs.searchBy: Invalid criteria"
5 }
```

2.6 Get songs for an artist

Spotify / SONG_ARTISTS / get songs for an artist

GET <http://localhost:8080/api/songcollection/artists/72dc2e44-2d3a-487b-a166-30548b14a793/songs> Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	Bulk Edit
-----	-------	-------------	-----------

Body Cookies Headers (8) Test Results Status: 200 OK Time: 22 ms Size: 487 B Save Response

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "id": 89,
4     "name": "Wind of Change",
5     "links": [
6       {
7         "rel": "self",
8         "href": "http://localhost:8080/api/songcollection/songs/89"
9       }
10    ]
11  },
12  {
13    "id": 88,
14    "name": "Crazy World",
15    "links": [
16      {
17        "rel": "self",
18        "href": "http://localhost:8080/api/songcollection/songs/88"
19      }
20    ]
21  }
22 ]
```

2.7 Create artist without auth header

Spotify / SONG_ARTISTS / create artist

PUT <http://localhost:8080/api/songcollection/artists/c020fd12-6c07-44fa-83f2-4d4b4ead0049> Send

Params Authorization Headers (10) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify

```
1 {
2   "name": "Chris Moreno",
3   "active": true
4 }
```

Body Cookies Headers (8) Test Results Status: 401 Unauthorized Time: 340 ms Size: 302 B Save Response

Pretty Raw Preview Visualize Text

```
1 http://localhost:8082/spotify/api/login
```

2.7 Create artist without being content manager

The screenshot shows a REST client interface for a Spotify API. The request is a PUT to `http://localhost:8080/api/songcollection/artists/c020fd12-6c07-44fa-83f2-4d4b4ead0049`. The body is a JSON object: `{ "name": "Chris Moreno", "active": true }`. The status is 403 Forbidden, with a time of 85 ms and size of 219 B.

```
PUT http://localhost:8080/api/songcollection/artists/c020fd12-6c07-44fa-83f2-4d4b4ead0049

{
  "name": "Chris Moreno",
  "active": true
}
```

Status: 403 Forbidden Time: 85 ms Size: 219 B

2.8 Create artist being content manager

The screenshot shows a REST client interface for a Spotify API. The request is a PUT to `http://localhost:8080/api/songcollection/artists/c020fd12-6c07-44fa-83f2-4d4b4ead0049`. The body is a JSON object with more fields: `{ "uuid": "c020fd12-6c07-44fa-83f2-4d4b4ead0049", "name": "Chris Moreno", "active": true, "songs": [], "hasSongs": false, "_links": { "self": { "href": "http://localhost:8080/api/songcollection/artists/c020fd12-6c07-44fa-83f2-4d4b4ead0049" }, "parent": { "href": "http://localhost:8080/api/songcollection/artists?page,size,name,match", "templated": true }, "assign songs": { "href": "http://localhost:8080/api/songcollection/artists/c020fd12-6c07-44fa-83f2-4d4b4ead0049/songs", "type": "POST" }, "delete artist": { "href": "http://localhost:8080/api/songcollection/artists/c020fd12-6c07-44fa-83f2-4d4b4ead0049", "type": "DELETE" } } }`. The status is 201 Created, with a time of 148 ms and size of 854 B.

```
PUT http://localhost:8080/api/songcollection/artists/c020fd12-6c07-44fa-83f2-4d4b4ead0049

{
  "uuid": "c020fd12-6c07-44fa-83f2-4d4b4ead0049",
  "name": "Chris Moreno",
  "active": true,
  "songs": [],
  "hasSongs": false,
  "_links": {
    "self": {
      "href": "http://localhost:8080/api/songcollection/artists/c020fd12-6c07-44fa-83f2-4d4b4ead0049"
    },
    "parent": {
      "href": "http://localhost:8080/api/songcollection/artists?page,size,name,match",
      "templated": true
    },
    "assign songs": {
      "href": "http://localhost:8080/api/songcollection/artists/c020fd12-6c07-44fa-83f2-4d4b4ead0049/songs",
      "type": "POST"
    },
    "delete artist": {
      "href": "http://localhost:8080/api/songcollection/artists/c020fd12-6c07-44fa-83f2-4d4b4ead0049",
      "type": "DELETE"
    }
  }
}
```

Status: 201 Created Time: 148 ms Size: 854 B

2.9 Replace artist

Spotify / SONG_ARTISTS / create artist

PUT http://localhost:8080/api/songcollection/artists/c020fd12-6c07-44fa-83f2-4d4b4ead0049

Params Authorization Headers (10) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   ... "name": "Chris Moreno",
3   ... "active": false
4 }
```

Body Cookies Headers (6) Test Results

Status: 204 No Content Time: 84 ms Size: 201 B Save Response

Pretty Raw Preview Visualize Text

1

2.10 Delete artist

Spotify / SONG_ARTISTS / delete artist

DELETE http://localhost:8080/api/songcollection/artists/c020fd12-6c07-44fa-83f2-4d4b4ead0049

Params Authorization Headers (8) Body Pre-request Script Tests Settings

Body Cookies Headers (8) Test Results

Status: 200 OK Time: 143 ms Size: 589 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "uuid": "c020fd12-6c07-44fa-83f2-4d4b4ead0049",
3   "name": "Chris Moreno",
4   "active": false,
5   "songs": [],
6   "hasSongs": false,
7   "_links": {
8     "self": {
9       "href": "http://localhost:8080/api/songcollection/artists/c020fd12-6c07-44fa-83f2-4d4b4ead0049"
10    },
11    "parent": {
12      "href": "http://localhost:8080/api/songcollection/artists?page,size,name,match",
13      "templated": true
14    }
15  }
16 }
```

2.11 Delete nonexistent artist

Spotify / SONG_ARTISTS / delete artist

DELETE http://localhost:8080/api/songcollection/artists/c020fd12-6c07-44fa-83f2-4d4b4ead0049

Params Authorization Headers (8) Body Pre-request Script Tests Settings

Body Cookies Headers (8) Test Results

Status: 404 Not Found Time: 59 ms Size: 365 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "error": "NOT_FOUND",
3   "status": 404,
4   "details": "Could not find artist c020fd12-6c07-44fa-83f2-4d4b4ead0049"
5 }
```


2.12 Add new song

Spotify / SONG_ARTISTS / add new song

Save Send

POST http://localhost:8080/api/songcollection/songs

Params Authorization Headers (10) Body Pre-request Script Tests Settings Cookies Beautify

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   ... "name": "Don't Believe Her",
3   ... "genre": "ROCK",
4   ... "year": 1990,
5   ... "type": "SONG",
6   ... "parentId": 88,
7   ... "artists": ["Scorpions"]
8 }
```

Body Cookies Headers (8) Test Results Status: 201 Created Time: 486 ms Size: 928 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 97,
3   "name": "Don't Believe Her",
4   "genre": "ROCK",
5   "year": 1990,
6   "type": "SONG",
7   "parentId": 88,
8   "_links": {
9     "self": {
10      "href": "http://localhost:8080/api/songcollection/songs/97"
11     },
12     "parent": {
13      "href": "http://localhost:8080/api/songcollection/songs?page,size,searchBy,searchedValue,match",
14      "templated": true
15     },
16     "album": {
17      "href": "http://localhost:8080/api/songcollection/songs/88"
18     },
19     "artists": {
20      "href": "http://localhost:8080/api/songcollection/songs/97/artists"
21     },
22     "add to playlist": {
23      "href": "http://localhost:8081/api/playlistscollection/playlists/{playlistId}/songs",
24      "type": "PATCH",
25      "templated": true
26     },
27     "delete song": {
28      "href": "http://localhost:8080/api/songcollection/songs/97",
29      "type": "DELETE"
30     }
31   }
32 }
```

Cookies Capture requests Postman Runner Trac

2.13 Assign song to artist

Spotify / SONG_ARTISTS / assign songs to an artist

POST http://localhost:8080/api/songcollection/artists/72dc2e44-2d3a-487b-a166-30548b14a793/songs

Params Authorization Headers (10) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   "songsId": [97]
3 }
```

Body Cookies Headers (8) Test Results

Status: 200 OK Time: 79 ms Size: 126 KB Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "uuid": "72dc2e44-2d3a-487b-a166-30548b14a793",
3   "name": "Scorpions",
4   "active": true,
5   "songs": [
6     {
7       "id": 89,
8       "name": "Wind of Change",
9       "_links": {
10        "self": {
11          "href": "http://localhost:8080/api/songcollection/songs/89"
12        }
13      }
14    },
15    {
16      "id": 97,
17      "name": "Don't Believe Her",
18      "_links": {
19        "self": {
20          "href": "http://localhost:8080/api/songcollection/songs/97"
21        }
22      }
23    },
24    {
25      "id": 88,
26      "name": "Crazy World",
27      "_links": {
28        "self": {
29          "href": "http://localhost:8080/api/songcollection/songs/88"
30        }
31      }
32    }
33  ],
34   "hasSongs": true,
35   "_links": {
36     "self": {
37       "href": "http://localhost:8080/api/songcollection/artists/72dc2e44-2d3a-487b-a166-30548b14a793"
38     },
39     "parent": {
40       "href": "http://localhost:8080/api/songcollection/artists?page,size,name,match",
41       "templated": true
42     },
43     "songs": {
44       "href": "http://localhost:8080/api/songcollection/artists/72dc2e44-2d3a-487b-a166-30548b14a793/songs"
45     },
46     "assign songs": {
47       "href": "http://localhost:8080/api/songcollection/artists/72dc2e44-2d3a-487b-a166-30548b14a793/songs",
48       "type": "POST"
49     },
50     "delete artist": {
51       "href": "http://localhost:8080/api/songcollection/artists/72dc2e44-2d3a-487b-a166-30548b14a793",
52       "type": "DELETE"
53     }
54   }
55 }
```

Cookies Capture requests Bootcamp Runner Trash

2.14 Delete album

Spotify / SONG_ARTISTS / delete song

DELETE <http://localhost:8080/api/songcollection/songs/88> Send

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies

Body Cookies Headers (8) Test Results Status: 409 Conflict Time: 76 ms Size: 373 B Save Response

Pretty Raw Preview Visualize JSON

```
1  {
2    "error": "CONFLICT",
3    "status": 409,
4    "details": "You are not able to remove this album until you remove all its songs"
5  }
```

2.15 Delete song

Spotify / SONG_ARTISTS / delete song

DELETE <http://localhost:8080/api/songcollection/songs/75> Send

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies

Body Cookies Headers (8) Test Results Status: 200 OK Time: 85 ms Size: 761 B Save Response

Pretty Raw Preview Visualize JSON

```
1  {
2    "id": 75,
3    "name": "Most beautiful song",
4    "genre": "POP",
5    "year": 2019,
6    "type": "SONG",
7    "songs": [],
8    "_links": {
9      "self": {
10       "href": "http://localhost:8080/api/songcollection/songs/75"
11      },
12      "parent": {
13       "href": "http://localhost:8080/api/songcollection/songs?page,size,searchBy,searchedValue,match",
14       "templated": true
15      },
16      "artists": {
17       "href": "http://localhost:8080/api/songcollection/songs/75/artists"
18      },
19      "add to playlist": {
20       "href": "http://localhost:8081/api/playlistscollection/playlists/{playlistId}/songs",
21       "type": "PATCH",
22       "templated": true
23      }
24    }
25  }
```

2.16 Update song

Spotify / SONG_ARTISTS / update song

PATCH Save Send

Params Authorization Headers (10) **Body** Pre-request Script Tests Settings Cookies

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON** Beautify

```
1 {
2   ... "name": "Normal name for this song",
3   ... "genre": "ROCK",
4   ... "year": 2019
5 }
```

Body Cookies Headers (6) Test Results Status: 204 No Content Time: 70 ms Size: 201 B Save Response

Pretty Raw Preview Visualize Text Copy

```
1 |
```

2.17 Invalid song update

Spotify / SONG_ARTISTS / update song

PATCH Save Send

Params Authorization Headers (10) **Body** Pre-request Script Tests Settings Cookies

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON** Beautify

```
1 {
2   ... "name": "Normal name for this song",
3   ... "genre": "ROC",
4   ... "year": 2019
5 }
```

Body Cookies Headers (8) Test Results Status: 422 Unprocessable Entity (WebDAV) (RFC 4918) Time: 60 ms Size: 362 B Save Response

Pretty Raw Preview Visualize **JSON** Copy

```
1 {
2   "error": "UNPROCESSABLE_ENTITY",
3   "status": 422,
4   "details": "Invalid genre"
5 }
```

3. PLAYLISTS

3.1 Create playlist

The screenshot shows a REST client interface for a Spotify API. The request is a POST to `http://localhost:8081/api/playlistscollection/playlists/` with a JSON body: `{ "name": "My music" }`. The response is a 201 Created status with a JSON body containing playlist details: `{ "id": "63cd7dac016c9f3532815379", "name": "My music", "favSongs": [], "_links": { "self": { "href": "http://localhost:8081/api/playlistscollection/playlists/63cd7dac016c9f3532815379" }, "parent": { "href": "http://localhost:8081/api/playlistscollection/playlists?name=", "templated": true } } }`.

```
1 {
2   "name": "My music"
3 }
```

Body: Cookies Headers (8) Test Results Status: 201 Created Time: 115 ms Size: 613 B Save Response

```
1 {
2   "id": "63cd7dac016c9f3532815379",
3   "name": "My music",
4   "favSongs": [],
5   "_links": {
6     "self": {
7       "href": "http://localhost:8081/api/playlistscollection/playlists/63cd7dac016c9f3532815379"
8     },
9     "parent": {
10      "href": "http://localhost:8081/api/playlistscollection/playlists?name=",
11      "templated": true
12    }
13  }
14 }
```

3.2 Add song to created playlist

The screenshot shows a REST client interface for a Spotify API. The request is a PATCH to `http://localhost:8081/api/playlistscollection/playlists/63cd7dac016c9f3532815379/songs` with a JSON body: `{ "songId": 88 }`. The response is a 204 No Content status.

```
1 {
2   "songId": 88
3 }
```

Body: Cookies Headers (6) Test Results Status: 204 No Content Time: 188 ms Size: 201 B Save Response

```
1 {
2   "songId": 88
3 }
```

3.3 Get playlist by id

Spotify / PLAYLISTS / **get playlist by id**

GET <http://localhost:8081/api/playlistscollection/playlists/63cd7dac016c9f3532815379> Send

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies

Body Cookies Headers (8) Test Results Status: 200 OK Time: 63 ms Size: 856 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": "63cd7dac016c9f3532815379",
3   "name": "My music",
4   "favSongs": [
5     {
6       "resourceId": 88,
7       "name": "Crazy World",
8       "link": "http://localhost:8080/api/songcollection/songs/88"
9     },
10    {
11      "resourceId": 89,
12      "name": "Wind of Change",
13      "link": "http://localhost:8080/api/songcollection/songs/89"
14    }
15  ],
16  "_links": {
17    "self": {
18      "href": "http://localhost:8081/api/playlistscollection/playlists/63cd7dac016c9f3532815379"
19    },
20    "parent": {
21      "href": "http://localhost:8081/api/playlistscollection/playlists{name}",
22      "templated": true
23    }
24  }
25 }
```

3.4 Delete playlist

Spotify / PLAYLISTS / **delete playlist**

DELETE <http://localhost:8081/api/playlistscollection/playlists/63cd7eb9016c9f353281537a> Send

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies

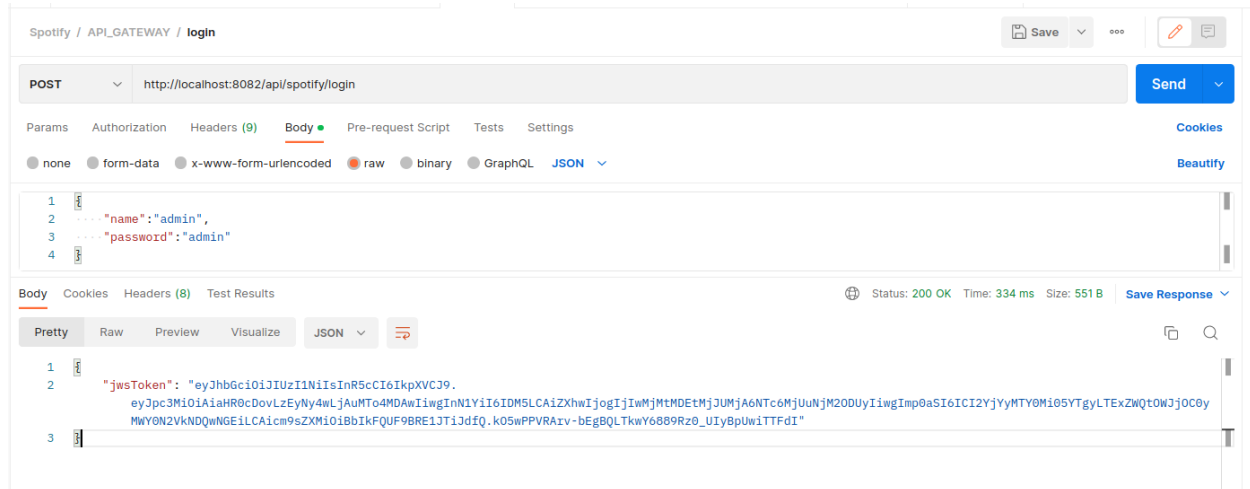
Body Cookies Headers (8) Test Results Status: 200 OK Time: 64 ms Size: 605 B Save Response

Pretty Raw Preview Visualize JSON

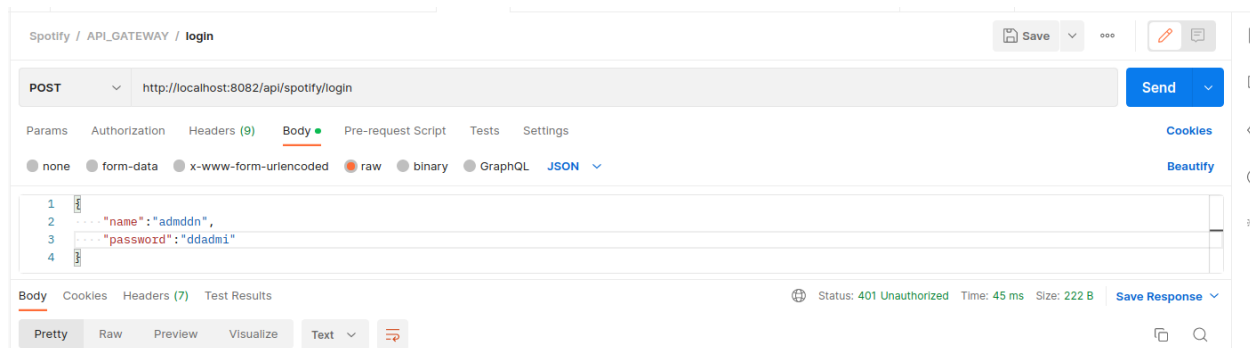
```
1 {
2   "id": "63cd7eb9016c9f353281537a",
3   "name": "Sport",
4   "favSongs": [],
5   "_links": {
6     "self": {
7       "href": "http://localhost:8081/api/playlistscollection/playlists/63cd7eb9016c9f353281537a"
8     },
9     "parent": {
10      "href": "http://localhost:8081/api/playlistscollection/playlists{name}",
11      "templated": true
12    }
13  }
14 }
```

4. API GATEWAY

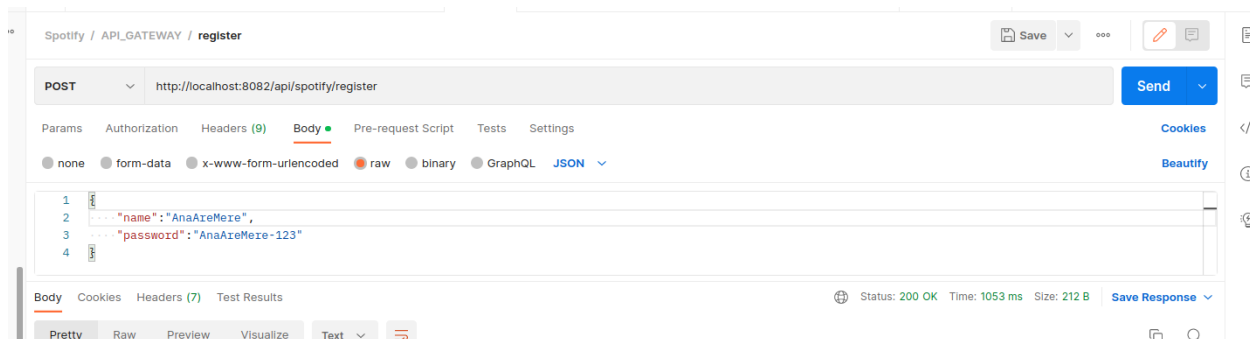
4.1 Login



4.2 Invalid login



4.3 Register



4.4 Logout

