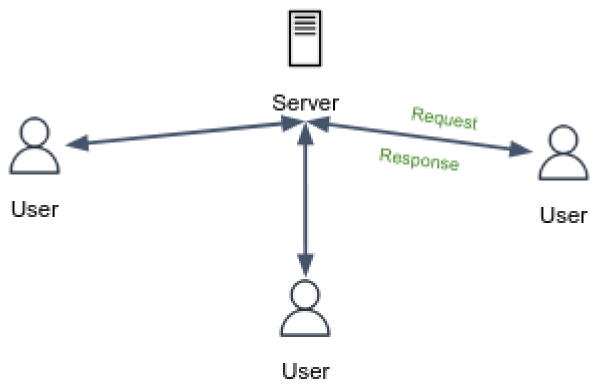
HTTP Protocol and APIs

The HTTP Protocol

Overview

The Hypertext Transfer Protocol (HTTP) is an application-level protocol that is being widely used over the Web. HTTP is a request/response protocol, which means, the client sends a request to the server (request method, URI, protocol version, followed by headers, and possible body content). The server responds (status code line, a success or error code, followed by server headers information, and possible entity-body content).



HTTP is said to be a stateless protocol. The server sends the requested content to clients without storing any information about the client. If a particular client sends the same request twice in a period of a few seconds, the server does not respond by saying that it just served the same request to the client. Instead, the server re-sends the data, as it has completely forgotten what it did earlier.

Nevertheless, although HTTP itself is stateless by design, most modern servers have a complex backend logic that stores information about logged-in clients, and by this means, we can say that those servers are stateful.

HTTP Request and Response

We can learn a lot by taking a closer look on a raw HTTP request and response that sent over the network:

curl -v http://httpbin.org/html

Below is the actual raw HTTP request sent by curl to the server:

GET /html HTTP/1.1

Host: httpbin.org

User-Agent: curl/7.58.0

Accept: */*

The server response is:

HTTP/1.1 200 OK

```
Date: Wed, 05 Apr 2023 12:43:42 GMT

Content-Type: text/html; charset=utf-8

Content-Length: 3741

Connection: keep-alive

Server: gunicorn/19.9.0

Access-Control-Allow-Origin: *

Access-Control-Allow-Credentials: true


<!DOCTYPE html>

<html>

  <head>

....
```

The MDN web docs specify the core components of request and response objects, review this resource.

Request Method

The request method specifies the type of HTTP action to perform. Here are the most common HTTP methods:

Method	Action
GET	Retrieves resources
POST	Creates resources
PUT	Changes and/or replaces resources or collections
DELETE	Deletes resources

Status code

HTTP response status codes indicate how a specific HTTP request was completed.

Responses are grouped in five classes:

- Informational (100–199)
- Successful (200–299)
- Redirection (300–399)
- Client error (400–499)
- Server error (500–599)

Try yourself to perform the below two HTTP requests, and read about the meaning of each status code.

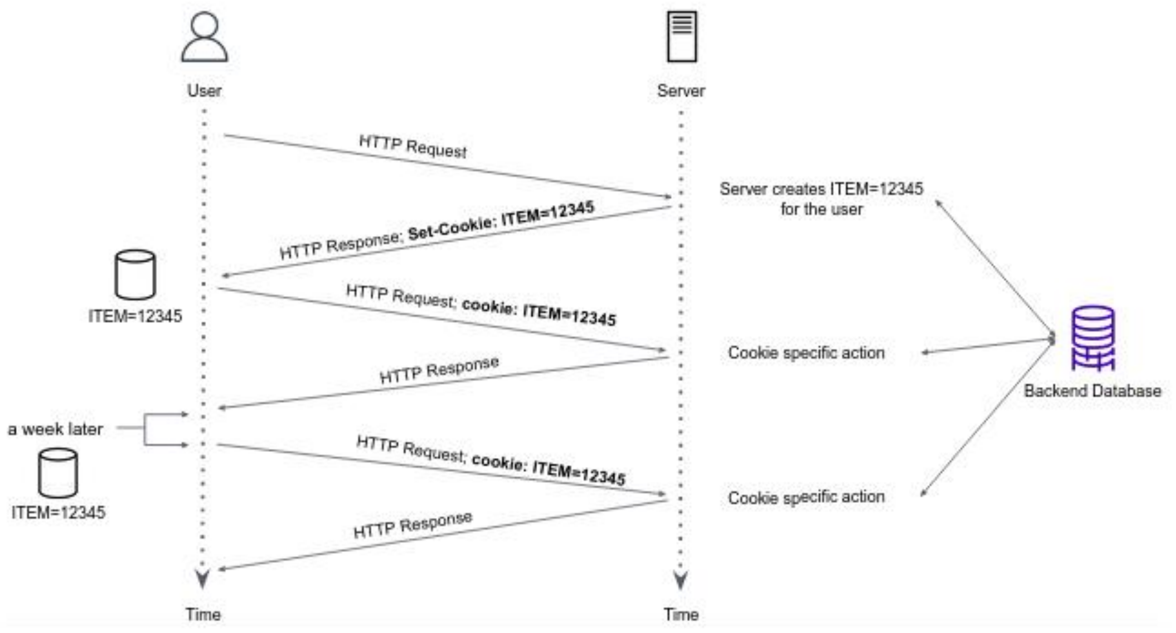
```
curl -i httpbin.org/status/0
```

```
curl -v -X PUT httpbin.org/path/to/nowhere
```

Cookies

We mentioned that an HTTP server is stateless. However, it is often desirable for a website to remember stateful information of users. For this purpose, HTTP uses cookies.

An HTTP cookie is a small piece of data that a server sends to a user's web browser. The browser may store the cookie and send it back to the same server with later requests. Most major commercial websites use cookies today.



As shown in the figure above, a cookie has four components:

1. Set-Cookie header line in the HTTP response message, usually in the form of key=value.
2. Cookie header line in the HTTP request message.
3. A cookie file kept on the user's end system and managed by the user's browser.
4. A back-end database at the website.

Suppose Berta contacts Amazon.com for the first time. When the request comes into the Amazon server, the server creates a unique id and creates an entry in its back-end database that is indexed by the id. Amazon server then responds to Berta's browser, including in the HTTP response a Set-cookie header, which contains the id value. For example, the header line might be:

```
Set-cookie: ITEM=12345
```

When Berta's browser receives the HTTP response message, it sees the Set-cookie header, and appends a line to the special cookie file that it manages locally.

As Berta continues to browse the Amazon site, each time she requests a web page, her browser consults her cookie file, extracts her id value for this site, and puts a cookie header line that includes the id value in the HTTP request. Specifically, each of her HTTP requests to the Amazon server includes the header line:

Cookie: ITEM=12345

In this manner, Amazon server is able to track Berta's activity, it knows exactly which pages she visited, in which order, and at what times!

Warning

The dark side of cookies

Although cookies often simplify and improve user experience, they are controversial because they can also be considered as an invasion of privacy. As we just saw, using a combination of cookies and user-supplied account information, a website can learn a lot about a user and potentially sell this information to a third party. The GDPR website includes extensive information on cookies compliance requirements.

APIs

An API (Application Programming Interface) can be thought of as a collection of all server endpoints that together define the functionality that is exposed to the client. Each endpoint typically accepts input parameters in a specific format and returns output data in a standard format such as JSON or XML.

For example, a web API for a social media platform might include endpoints for retrieving a user's profile information, posting a new status update, or searching for other users. Each endpoint would have a unique URL and a specific set of input parameters and output data.

Many platforms expose both API, and GUI. Like Spotify.

Introducing Postman

Postman is a powerful and user-friendly tool for testing, debugging, and documenting APIs.

1. Download from: https://learning.postman.com/docs/getting-started/first-steps/get-postman/.
2. Create a Postman account to unlock some important features.

Exercises

✏️ Practicing the HTTP protocol

The Accept header

1. Use curl to perform an HTTP GET request to http://httpbin.org/image. Add an Accept header to your requests to indicate that you anticipate a png image.
2. Read carefully the Warning message written by curl at the end of the server response, follow the instructions to save the image on the file system.

3. Execute another curl to save the image in the file system.

Which animal appears in the served image?

Status code

1. Perform an HTTP GET request to google.com
2. What does the server response status code mean? Follow the response headers and body to get the real Google's home page.
3. Which HTTP version does the server use in the above response?

✏️ Return informational errors for the CatalogController controller

In this exercise you'll modify your SpotifyCatalogAPI to better handle unsuccessful requests and responses.

When a user perform and unsuccessful request to your API, you'll return proper response code, along with a JSON object containing the following information:

Key	Value Type	Value Description
status	integer	The HTTP status code that is also returned in the response header.
message	string	A short description of the cause of the error.

Here, for example is the error that occurs when trying to fetch information for a non-existent album:

```
$ curl -i "http://localhost:8080/albums/%%@#$D86ARO5beq7Q0Drfqa"


HTTP/1.1 400 Bad Request

{

  "error": {

    "status": 400,

    "message": "invalid id"

  }

}
```

Modify your CatalogController controller's endpoints according to the below specifications:

endpoint	status code	scenario
/albums/{id}	404	When there is not album with the requested ID
/albums/{id}	400	When the album ID is invalid
/albums/0000000000000000000000	403	When the requested album id is all 0s (imagine this id is used for internal purposes in your API)
All endpoints	500	Any internal error. E.g. unable to read the JSON files from /data dir.
All endpoints	503	When list read from /data/popular_songs.json is empty (imagine this scenario happens 2-3 times a year due to temporary backup work)

This example can help you understand how to return a custom status code and JSON from your app controllers.

```
@GetMapping("/example")

public ResponseEntity<JsonNode> getExample() {

  ObjectNode errorResponse = objectMapper.createObjectNode();

  errorResponse.put("message", "Resource not found");

  errorResponse.put("status", HttpStatus.BAD_REQUEST.value());


  return new ResponseEntity<>(errorResponse, HttpStatus.BAD_REQUEST);

}
```

1. In your pom.xml file modify the app version to reflect your changes.
2. Run mvn test to run your tests suite to make sure that you didn't break any previous functionality. Fix bugs if needed.
3. Commit & push to GitHub.

✏️ Postman collections and simple API testing

- Run your SpotifyCatalogAPI in the background
- Select + in the workspace page to open a new tab.

- Enter localhost:8080/ for the request URL
- Select Send.

Postman displays the response data sent from the server in the lower pane.

Every request you send in Postman appears under the History tab of the sidebar. On a small scale, reusing requests through the history section is convenient. As your Postman usage grows, it can be time-consuming to find a particular request in your history. Instead of scrolling through your history section, you can use Collections.

A Postman collection is a set of saved requests used to interact and test APIs. Collections allow you to group, organize, and execute multiple API calls easily.

- In your request builder and select Save.
- Create a new collection by selecting New Collection. Enter a collection name, and then select Create.
- Select Save to save the request in the new collection.
- Add to the created collection multiple requests to different CatalogController's endpoints (both successfully and unsuccessful requests, as described in the previous exercise).

API tests are a way to ensure that your API is behaving as you expect it to. For example, you might write a test to validate your API's error handling by sending a request with incomplete data or wrong parameters.

You can create and configure and execute API tests using Postman and JavaScript.

- Go to one of your requests tab.
- In the request, go to the Scripts tab, then select the Post-response tab.
- In the snippet section to the right, select the snippet Status code: Code is 200. This will enter the following test code:
- pm.test("Status code is 200", function () {
-   pm.response.to.have.status(200);
- });
- Create a test for one of your request that should return a non-ok status code.
- You can utilize a built-in AI agent by clicking on the button in the top right corner of the code textbox.

✏️ Working with the real Spotify API

1. Create a free Spotify account if you don't have it yet.
2. Login to the Spotify Developer Dashboard. If necessary, read the latest Developer Terms of Service to complete your account set up.
3. Follow the Getting started guide till you're able to communicate with the API.

4. **Use Portman and the Spotify API reference docs to perform the following tasks:**
   - **Use the /search endpoint to get the artist id of your favorite artist (take a look in the .**
   - **Get the artist top tracks (search for a dedicated endpoint to fetch such information).**
   - **Start follow some artist.**
   - **Unfollow the artist.**