REST APIs

References:
1. https://www.codecademy.com/article/what-is-rest
2. ChatGPT

REST (Representational State Transfer)

- REST API is a type of Web API
- More specifically, REST (Representational State Transfer) is an architectural style for designing networked applications. It uses HTTP requests to perform CRUD (Create, Read, Update, Delete) operations on resources.

KEY CONCEPTS:

1. Resources

In REST, everything is considered a resource. Resources are identified by URIs (Uniform Resource Identifiers). For example, in a RESTful API for a library system, books could be resources, and their URIs might look like /books/123 for a specific book with ID 123.

2. HTTP Methods

REST APIs use standard HTTP methods to perform operations on resources:
- **GET**: Retrieve information about a resource.
- **POST**: Create a new resource.
- **PUT**: Update an existing resource.
- **DELETE**: Remove a resource.

3. Statelessness

Each request from a client to a server must contain all the information needed to understand and process the request. The server does not store any state about the client between requests. This means every request is independent.

4. Data Representation

Resources can be represented in various formats, such as JSON or XML. For example, a GET request to /books/123 might return a JSON object representing the book with ID 123.

5. Uniform Interface

A RESTful API should have a consistent and uniform interface, which simplifies the architecture and makes it easier for developers to interact with the API. This typically includes using standard conventions for resource URIs and HTTP methods.

6. HATEOAS (Hypermedia As The Engine Of Application State)

In a fully RESTful API, responses should include hyperlinks to related resources, allowing clients to navigate the API dynamically. For instance, if you retrieve a book resource, the response might include links to update or delete that book.

Example

Here’s an example of how RESTful operations might look for a resource representing a book:

- **GET** /books/123 - Retrieve information about the book with ID 123.
- **POST** /books - Create a new book. The request body would contain the book details.
- **PUT** /books/123 - Update the book with ID 123. The request body would contain the updated book details.
- **DELETE** /books/123 - Delete the book with ID 123.

Benefits

- **Scalability**: Stateless interactions and a uniform interface help APIs scale effectively.
- **Simplicity**: REST uses standard HTTP methods and status codes, making it straightforward to use.
- **Flexibility**: Clients and servers can evolve independently as long as they adhere to the API contract.