Understanding REST APIs: A Conceptual Guide for Beginners

July 13, 2024

1 What is an API?

API (Application Programming Interface): A set of rules and protocols that allows different software applications to communicate with each other. Think of it as a menu in a restaurant. The menu provides a list of dishes you can order, along with a description of each dish. When you specify what you want, the kitchen (backend) does the work and provides you with a meal (response).

2 What is REST?

REST (Representational State Transfer): An architectural style for designing networked applications. REST uses a stateless, client-server, cacheable communications protocol — the HTTP protocol. It's designed to take advantage of existing protocols.

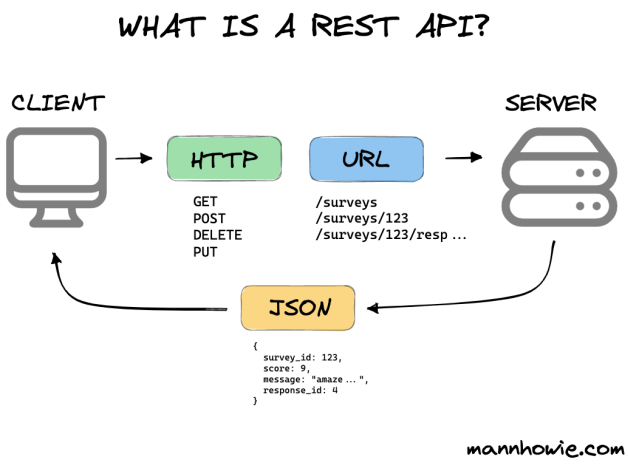

Figure 1: Caption

3 Key Concepts of REST APIs

3.1 Resources and URIs

Resource: Anything that can be named, such as a document, image, or a collection of other resources. **URI (Uniform Resource Identifier)**: A unique address to access each resource. For example, `http://api.example.com/users` might be a URI for a resource that represents a collection of users.

3.2 HTTP Methods

REST APIs use standard HTTP methods to perform actions on resources:

- **GET**: Retrieve data from the server.
- **POST**: Send data to the server to create a new resource.
- **PUT**: Update an existing resource with new data.
- **DELETE**: Remove a resource from the server.

3.3 Statelessness

Statelessness means that every time a client makes a request to the server, the request must include all the information needed for the server to understand and respond to it. The server does not remember anything about the client from previous requests. Each request is processed as if it's being made for the first time.

3.4 Client-Server Architecture

- The client is responsible for the user interface and the user experience.
- The server is responsible for processing the requests, storing data, and providing responses.

3.5 Cacheability

Cacheability in REST APIs determines if responses can be saved for later use. When you make a request to an API:

- If it's a simple request like asking for information (GET), the response can often be saved (cached) so you don't have to make the same request each time you ask for the same information.
- But if you're changing something (like adding or deleting data with POST or DELETE), those responses usually can't be saved because they might change things.

3.6 Layered System

A layered system in REST means that the architecture of the API is organized into different levels or layers, each with its own responsibility:

- Separation of Concerns: Each layer handles a specific aspect of the functionality, making the system easier to understand and maintain.
- Scalability: Layers can be added or modified independently, allowing the system to grow and adapt without affecting other parts.
- Example Layers: For example, one layer might handle data storage (database), another handles the logic (business rules), and another manages how clients interact (API endpoints)

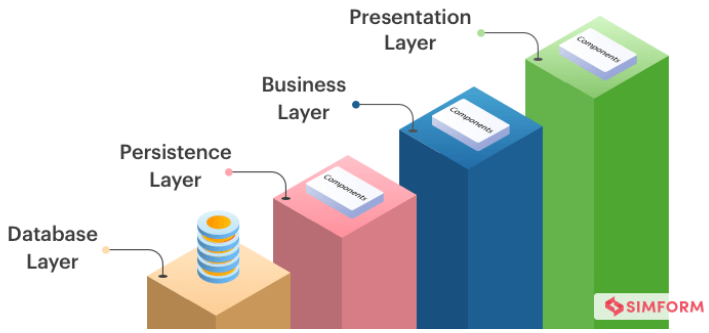

Figure 2: Caption

.

3.7 Code on Demand (Optional)

Servers can extend the functionality of a client by transferring executable code.

4 Examples of RESTful API Requests

GET Request: To fetch a list of users.

```
GET /users
```

The server responds with a list of users.

POST Request: To create a new user.

```
POST /users
```

The client sends the user data in the request body, and the server creates a new user.

PUT Request: To update a user's information.

```
PUT /users/1
```

The client sends updated data for the user with ID 1, and the server updates the user's information.

DELETE Request: To delete a user.

```
DELETE /users/1
```

The server deletes the user with ID 1.

5 Understanding HTTP Status Codes

HTTP status codes indicate the result of the request:

- **200 OK**: The request was successful.
- **201 Created**: A new resource has been created.
- **204 No Content**: The request was successful, but there is no content to send back.
- **400 Bad Request**: The server could not understand the request due to invalid syntax.
- **401 Unauthorized**: Authentication is needed to access the resource.
- **404 Not Found**: The requested resource was not found.
- **500 Internal Server Error**: The server encountered an error and could not complete the request.

6 Summary

- **REST API**: A way for different software applications to communicate over the internet using standard HTTP methods.
- **Resources**: Represented by URIs, can be manipulated using HTTP methods.
- **Statelessness**: Each request from client to server must be self-contained.
- **HTTP Methods**: GET, POST, PUT, DELETE used to interact with resources.
- **HTTP Status Codes**: Indicate the result of the request.

By understanding these concepts, you can start to grasp how REST APIs enable different systems to communicate effectively and efficiently over the web.