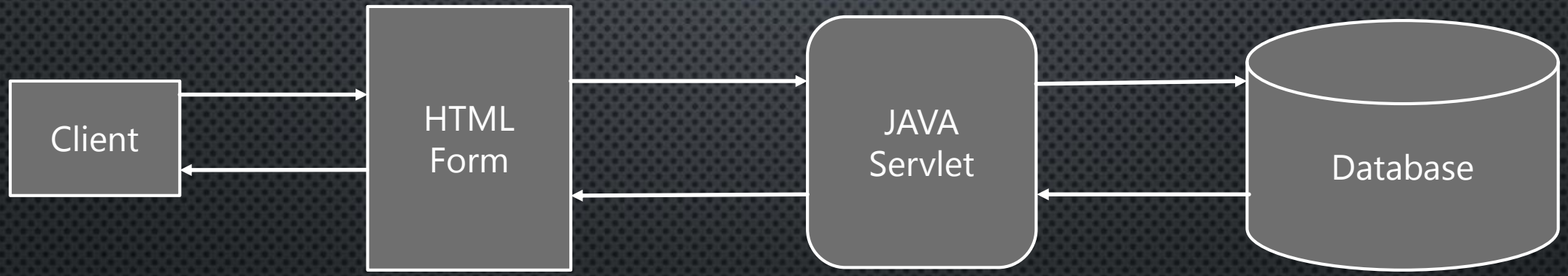
Representational State Transfer

RESTful Web Services and REST API

BEFORE REST

A typical client-server transaction to access something on the database could look like this



The user submits their data in a form, which will be fetched by the server and responds with the desired output from the database

BEFORE REST

Before the REST Architecture, Techniques like SOAP and XML-RPC were followed.

SOAP or Simple Object Access Protocol involves in sending XML requests and responses between clients and servers which contains the information.

SOAP is still used in a lot of enterprises because of they are more secure and provide ACID characteristics.

What is REST API?

- REST (Representational State Transfer) is an architectural style used for designing networked applications, particularly web services.
- It defines a set of principles for creating scalable, flexible, and easy-to-maintain APIs (Application Programming Interfaces) that communicate over the web.
- REST is based on the use of HTTP and leverages its methods for resource manipulation.
- RESTful Webservices is the practical implementation of the rules that are described into a live web service.

KEY CONCEPTS OF REST

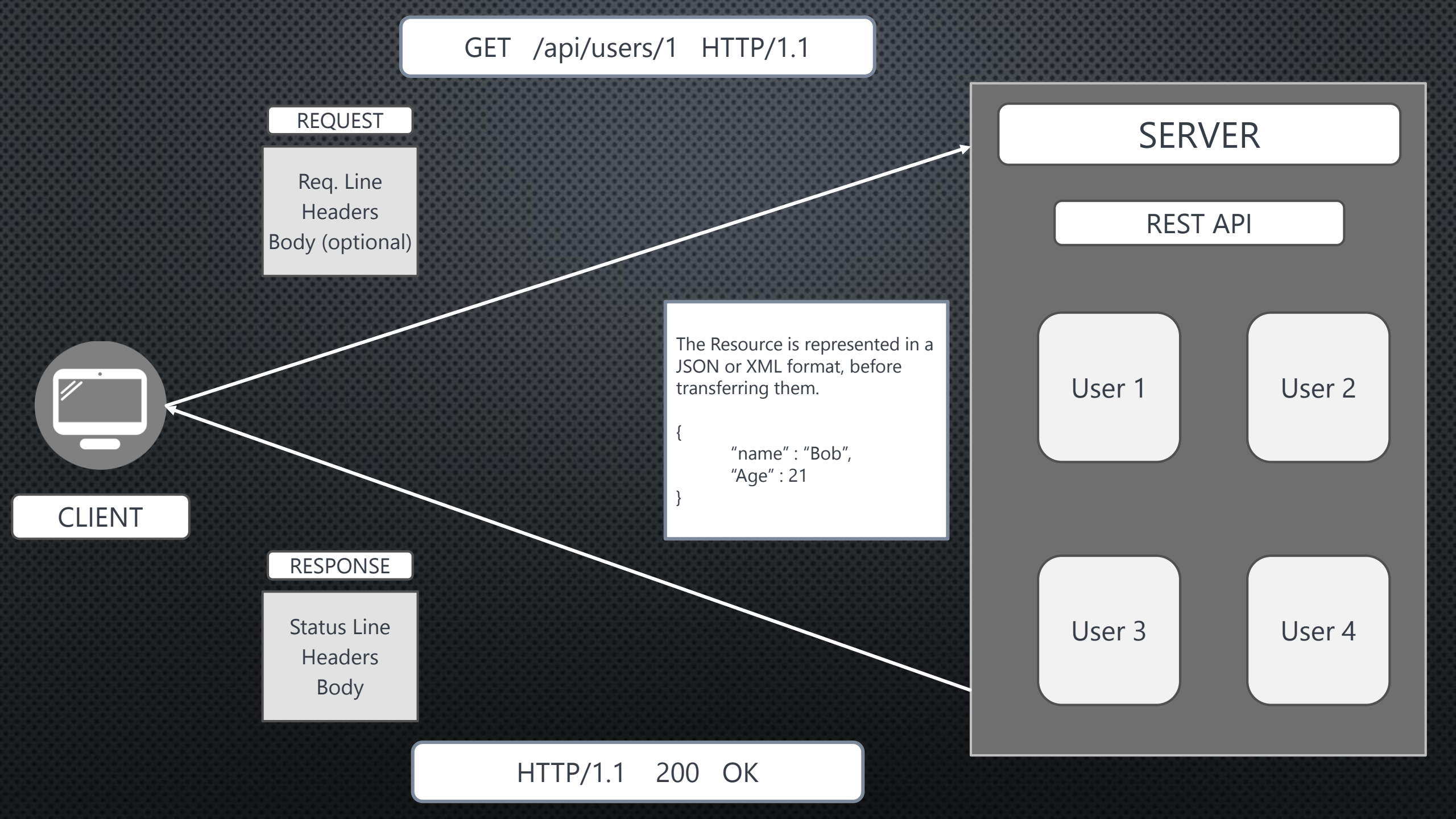
- Resource Based:
  Everything is considered as a resource.

- Client-Server Architecture:
  It follows the same principle as a Client-Server Architecture

- Stateless:
  Server will have no additional information about client

- HTTP Methods:
  REST uses standard HTTP methods to do operations.

- Cacheability:
  Responses from the server can be marked as cacheable which improves performance


GET /api/users/1 HTTP/1.1
REQUEST
Req. Line
Headers
Body (optional)
SERVER
REST API
User 1
User 2
User 3
User 4
The Resource is represented in a JSON or XML format, before transferring them.
{
"name" : "Bob",
"Age" : 21
}
CLIENT
RESPONSE
Status Line
Headers
Body
HTTP/1.1 200 OK

HTTP Request Properties

- HTTP METHODS:
  Methods like GET, POST, PUT, PATCH, etc.

- Uniform Resouce Identifier (URI):
  Each and Every Resource will be identified by an unique URI.
  `api/users/1`

- HTTP Version:
  Specifies the Version of HTTP that the client uses.

- Headers:
  Contains additional invormation in Key-Value Pairs.
  Host, Content-Type, Accept, Authorization, Cache-Control

- Body (optional):
  Contains the information which will be sent to the server.

```
GET /api/users/123

Host: server-domain.com
Accept: application/json

{
    “body”: “optional”
}
```

HTTP Response Properties

- STATUS CODES:
  Some of the status codes include:
  200 – OK
  404 – NOT FOUND
  500 – INTERNAL SERVER ERROR

- Description of the Status Code:
  Contains a short description of the Status code.

- Headers:
  Contains additional invormation in Key-Value Pairs.
  Host, Content-Type, Accept, Authorization, Cache-Control

- Body:
  Contains the information which will be sent from the server.

```
HTTP/1.1  200  OK

Content-Type: application/json
Content-length: 75
Cache-control: no-cache

{
    “Body” : “Asked info”,
    “Error” : “info”
}
```

THANKS FOR LISTENING