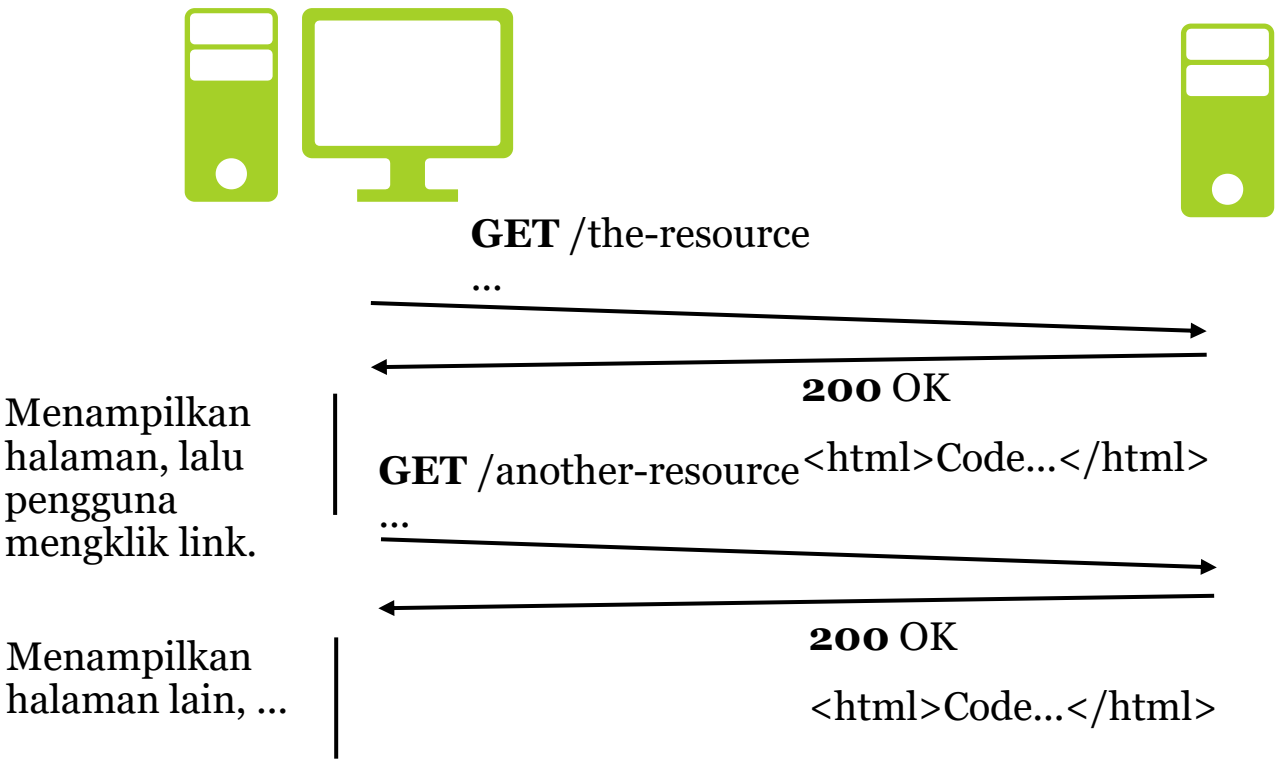
REST API

Mochammad Zen Samsono Hadi, ST. MSc. Ph.D

Traditional web applications

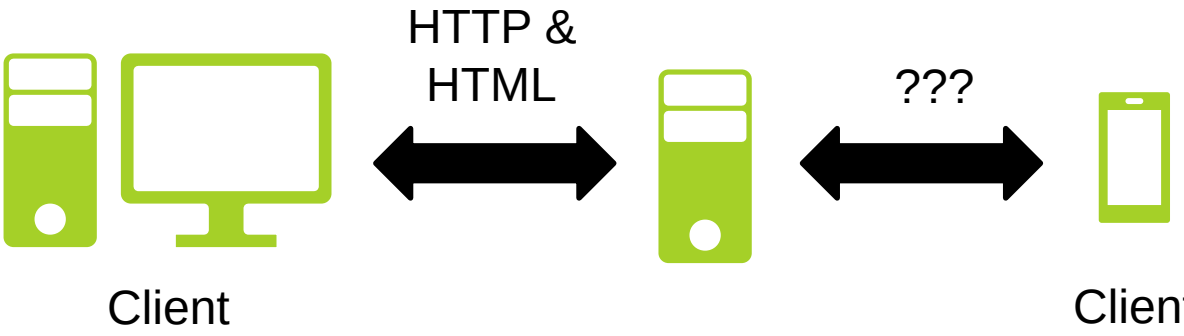


Traditional web applications

Interface dibangun di atas HTML & HTTP.

- Kekurangan:
  - Client harus memahami HTTP dan HTML.
  - Seluruh halaman web diganti dengan yang lain.
    - Tidak ada cara untuk menganimasikan transisi antar halaman web.
  - Data yang sama biasanya dikirim dalam beberapa tanggapan.
    - Misalnya. Kode HTML untuk layout.

Traditional web applications



- HTTP & HTML dapat digunakan, tetapi tidak optimal.
  - GUI pada smartphone tidak menggunakan HTML
  - Misal: `GET /users/3`:

Nama

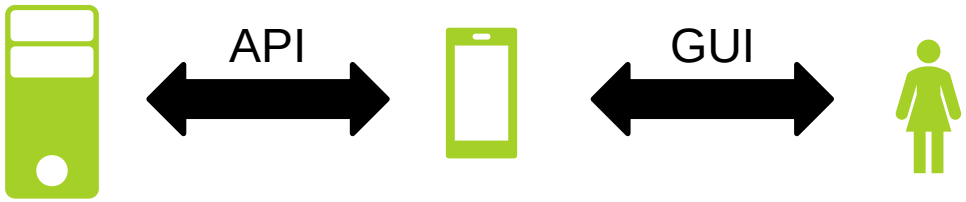
Umur

Kota

```
<h1>Claire</h1>
<p>Claire is 24 years old and lives in Boston.</p>
```

Application Programming Interface

GUI adalah interface untuk komunikasi Manusia ⇿ Mesin.



API adalah antarmuka untuk komunikasi Mesin ⇿ Mesin.

- API yang menggunakan HTTP disebut *Web API*.

Berbagai jenis Web API

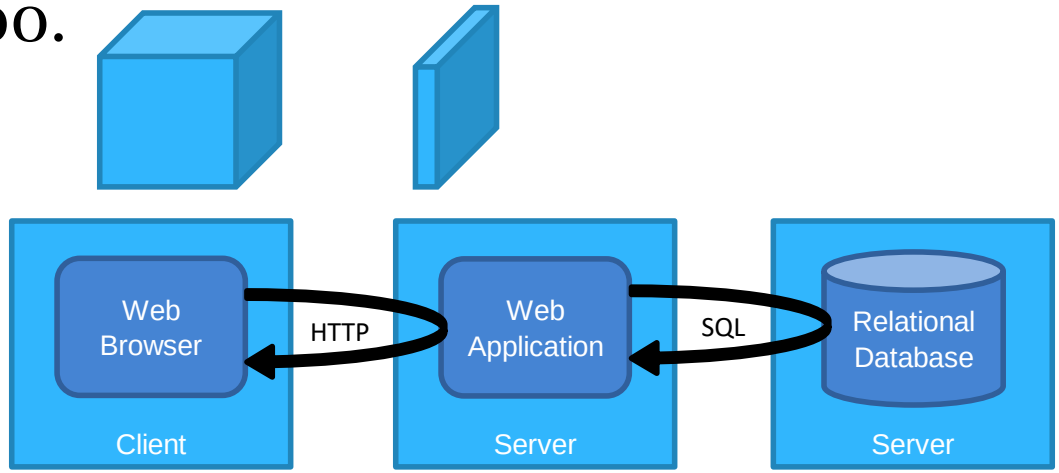
- *Remote Procedure Call*, RPC.
  - Client dapat memanggil fungsi di server.
- *Remote Method Invocation*, RMI.
  - Client dapat memanggil metode pada objek di server.
- *Representational State Transfer*, REST.
  - Client dapat menerapkan operasi CRUD (create, read, update, delete) pada sumber daya di server.

Apa itu REST?

Gaya arsitektur untuk sistem hypermedia terdistribusi dijelaskan oleh Roy Thomas Fielding dalam disertasi doktoralnya tahun 2000.

Terdiri dari batasan:

1. Client - Server
2. Stateless
3. Cache
4. Uniform Interface
5. Layered System
6. Code-On-Demand



Stateless artinya server tidak perlu mengetahui apa pun tentang status klien dan sebaliknya. Sehingga, baik server maupun klien dapat memahami pesan apa pun yang diterima, bahkan tanpa melihat pesan sebelumnya.

Apa artinya REST?

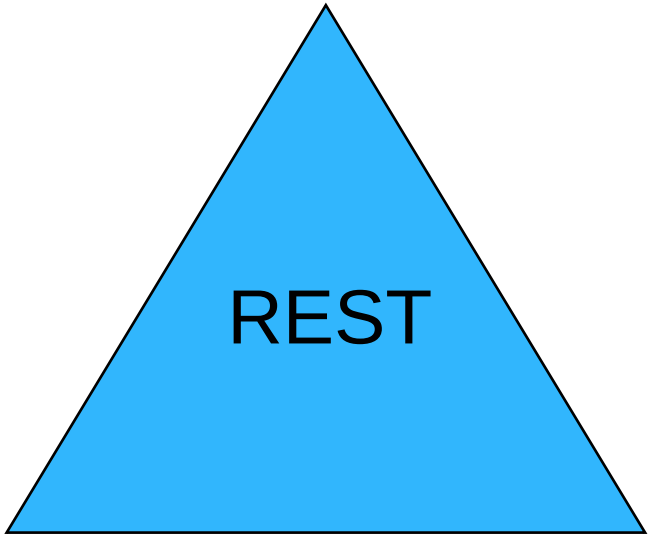
Nama "Representational State Transfer" dimaksudkan untuk membangkitkan gambaran tentang bagaimana aplikasi Web yang dirancang dengan baik berperilaku: jaringan halaman web (virtual state-machine), di mana pengguna melanjutkan melalui aplikasi dengan memilih link (state transition) , sehingga halaman berikutnya (mewakili status aplikasi berikutnya) ditransfer ke pengguna dan direnderer untuk digunakan.

Dari disertasi Roy.

REST merupakan gaya arsitektur untuk menyediakan standar antara sistem komputer di web, sehingga memudahkan sistem untuk berkomunikasi satu sama lain.

Konsep REST

Nouns (Resources)
unconstrained
Contoh http://example.com/employees/12345

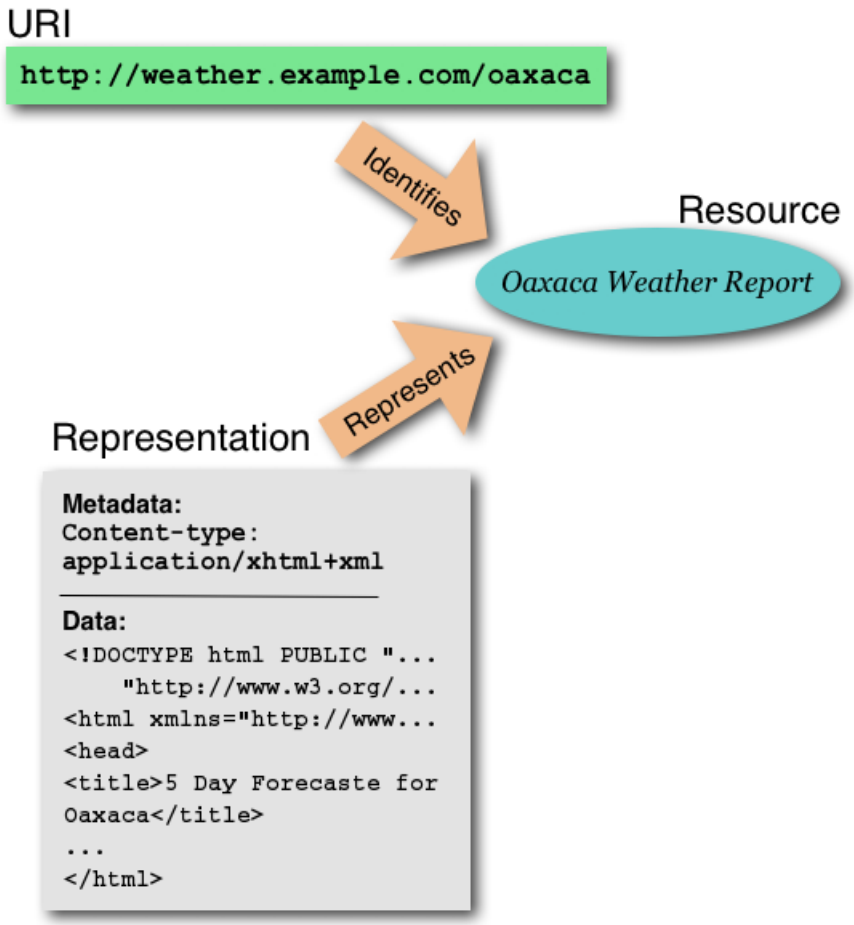


Verbs
constrained
Contoh, GET

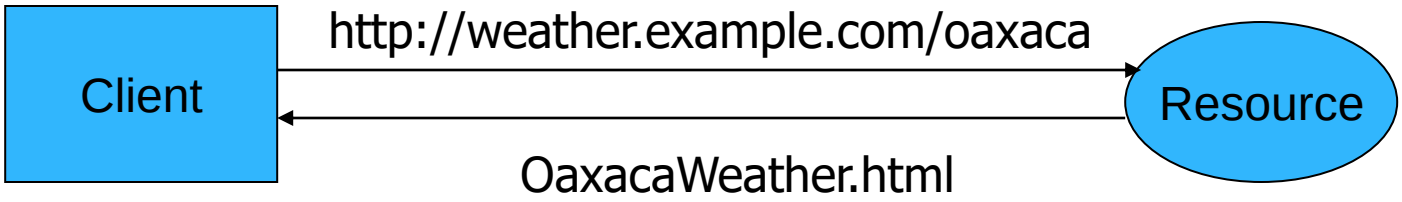
Representations
constrained
Contoh, XML, JSON

Konsep REST

- **Resource**
- **URI-Uniform Resource Identifier (or URL)**
- **Web Page (HTML Page)**



Konsep REST



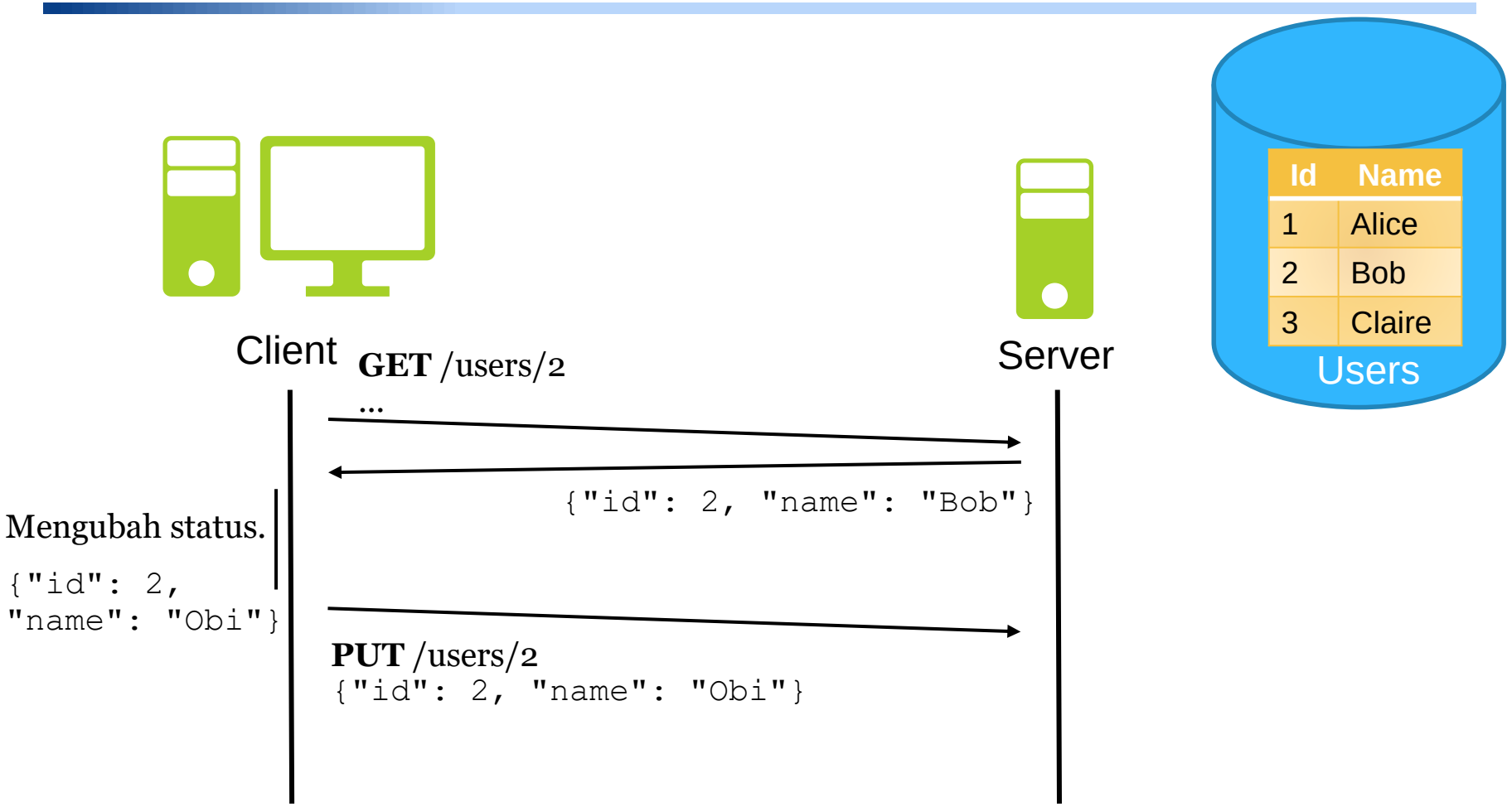
Dua format utama:

- JavaScript Object Notation (JSON)
- XML

Representations

- XML
  - <COURSE>
    - <ID>CS2650</ID>
    - <NAME>Distributed Multimedia Software</NAME>
  - </COURSE>
- JSON
  - {course
    - {id: CS2650}
    - {name: Distributed Multimedia Sofware}
  - }

Apa artinya REST?



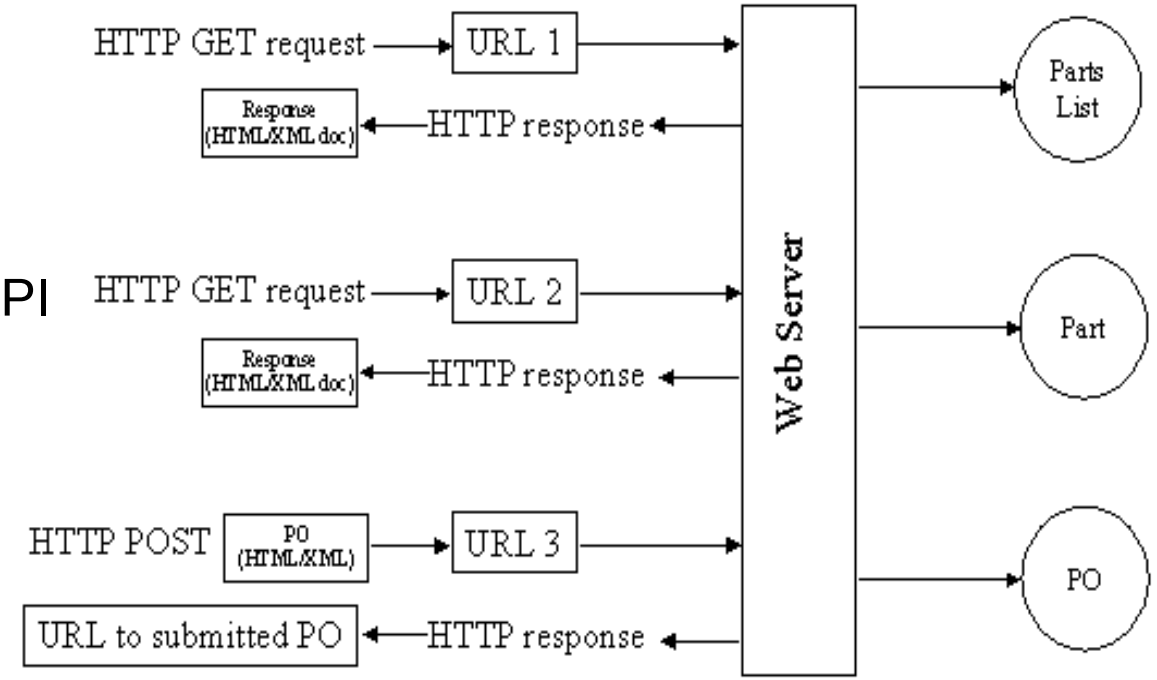
Aplikasi REST

REST based web services

- Online shopping
- Search services
- Dictionary services

Contoh REST:

- Google Maps
- Google AJAX Search API
- Yahoo Search API
- Amazon WebServices



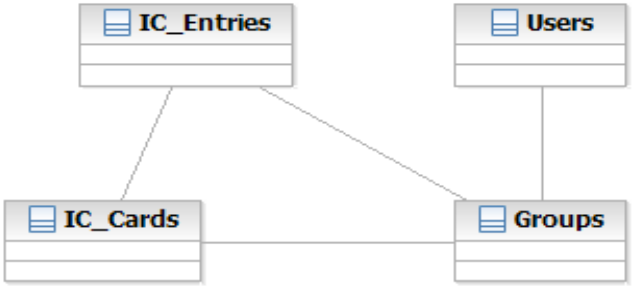
Invocation Model

Client

http

Channels

http



REST Service

SQL

Database

Menggunakan HTTP sebagai interface yang seragam

- Gunakan URI untuk mengidentifikasi sumber daya.
- Gunakan metode HTTP untuk menentukan operasi:
  - Create: POST (atau PUT)
  - Retrieve: GET
  - Update: PUT (atau PATCH)
  - Delete: DELETE

Bad	**Good**
`POST /login`	`POST /login-sessions`
`POST /create-book`	`POST /books`
`GET /get-top-10-books`	`GET /top-10-books`

- Gunakan header HTTP `Content-Type` dan `Accept` untuk menentukan format data untuk resources.
- Gunakan kode status HTTP untuk menunjukkan keberhasilan/kegagalan.

Menggunakan HTTP sebagai interface yang seragam

REST adalah gaya arsitektur, bukan spesifikasi.

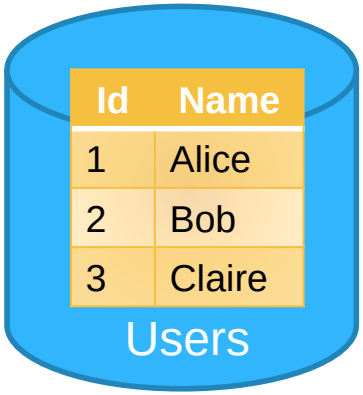
- Dalam praktiknya, ini dapat digunakan dengan berbagai cara.
  - Tetapi beberapa lebih baik dari yang lain.

Rekomendasi tentang Web API Design:

- Web API Design - Crafting Interfaces that Developers Love
  - https://pages.apigee.com/rs/apigee/images/api-design-ebook-2012-03.pdf

Contoh REST

Sebuah server dengan informasi tentang pengguna.



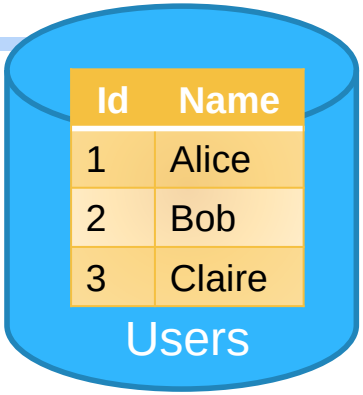
- Metode GET digunakan untuk mengambil resource.
  - `GET /users`
  - `GET /users/2`
  - `GET /users/pages/1`
  - `GET /users/gender/female`
  - `GET /users/age/18`
  - `GET /users/???`
  - `GET /users/2/name`
  - `GET /users/2/pets`

```
GET /users?page=1
GET /users?gender=female
GET /users?age=18
GET /users?gender=female&age=18
```

Contoh REST

Sebuah server dengan informasi tentang pengguna.

- Metode GET digunakan untuk mengambil resources.
  - Format data yang mana? Ditentukan oleh header `Accept`



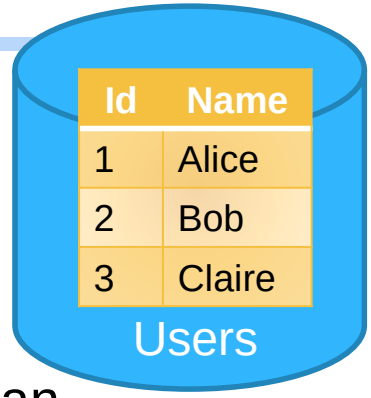

```
GET /users HTTP/1.1
Host: the-website.com
Accept: application/json
```

application/xml
pernah populer sebelum JSON.

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: 66

[
  {"id": 1, "name": "Alice"},
  {"id": 2, "name": "Bob"}
]
```

Contoh REST



Sebuah server dengan informasi tentang pengguna.

- Metode POST digunakan untuk membuat resource.
  - Format data yang mana? Ditentukan oleh header `Accept` dan `Content-Type`

```
POST /users HTTP/1.1
Host: the-website.com
Accept: application/json
Content-Type: application/xml
Content-Length: 49

<user>
  <name>Claire</name>
</user>
```

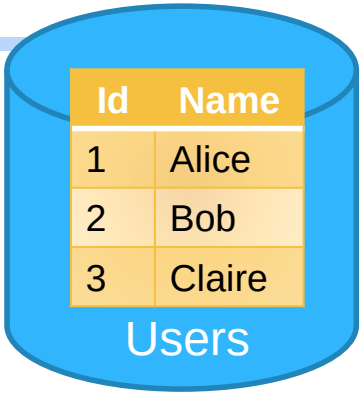
```
HTTP/1.1 201 Created
Location: /users/3
Content-Type: application/json
Content-Length: 28

{"id": 3, "name": "Claire"}
```

Contoh REST

Sebuah server dengan informasi tentang pengguna.

- Metode PUT digunakan untuk memperbarui seluruh resource.



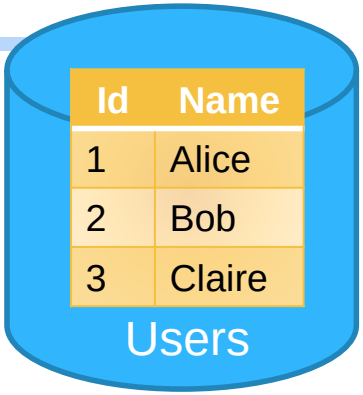

```
PUT /users/3 HTTP/1.1
Host: the-website.com
Content-Type: application/xml
Content-Length: 52

<user>
  <id>3</id>
  <name>Cecilia</name>
</user>
```

```
HTTP/1.1 204 No Content
```

PUT juga dapat digunakan untuk membuat sumber daya jika diketahui URI mana yang harus dimiliki selanjutnya.

Contoh REST



Sebuah server dengan informasi tentang pengguna.

- Metode DELETE digunakan untuk menghapus resource.

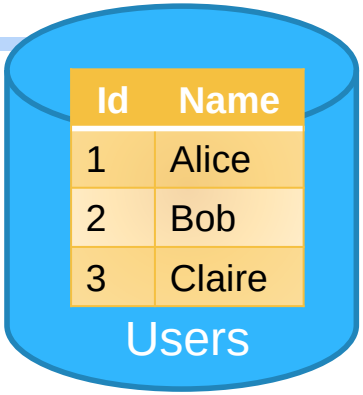
```
DELETE /users/2 HTTP/1.1
Host: the-website.com
```

```
HTTP/1.1 204 No Content
```

Contoh REST

Sebuah server dengan informasi tentang pengguna.

- Metode PATCH digunakan untuk memperbarui bagian dari resource.



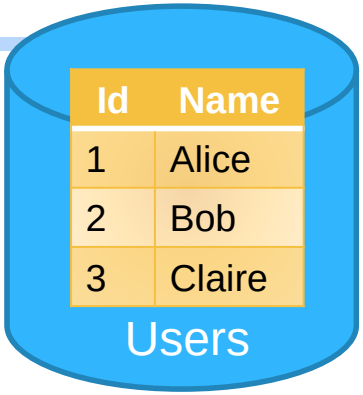

```
PATCH /users/1 HTTP/1.1
Host: the-website.com
Content-Type: application/xml
Content-Length: 37

<user>
  <name>Amanda</human>
</user>
```

```
HTTP/1.1 204 No Content
```

Metode PATCH hanyalah standar yang diusulkan.

Contoh REST



Sebuah server dengan informasi tentang pengguna.

- Bagaimana jika terjadi kesalahan?
  - Gunakan kode status HTTP untuk menunjukkan keberhasilan/kegagalan.

```
GET /users/999 HTTP/1.1
Host: the-website.com
Accept: application/json
```

```
HTTP/1.1 404 Not Found
```

- Baca lebih lanjut tentang berbagai kode status di:
  - http://www.restapitutorial.com/httpstatuscodes.html
- Opsional: disertakan pesan kesalahan *response body*.

Mendesain REST API

Bagaimana seharusnya?

- Buatlah semudah mungkin untuk digunakan oleh programmer lain.

Facebook:

- Always return 200 OK.
- `GET /v2.7/{user-id}`
- `GET /v2.7/{post-id}`
- `GET /v2.7/{user-id}/friends`
- `GET /v2.7/{object-id}/likes`

Mendesain REST API

Twitter:

- Only use GET and POST.
- `GET  /1.1/users/show.json?user_id=2244994945`
- `POST /1.1/favorites/destroy.json?id=243138128959913986`