Google Cloud

Edge REST Design Fundamentals

Response Codes & Pagination

Response Codes

Communicating enough, but never too much.

Response codes should communicate only what the developer consuming an API needs

Consider this scenario:

An actor requests a user profile by ID.
Users that don't exist at all are returned 404.
Users that do exist but aren't accessible return 403.

```
GET /users/U124

404 Not Found

GET /users/U123

403 Forbidden
```

Inadvertent leakage.

A malicious actor can use this information to determine which user ids are valid, and which are not assigned.

This information can be used to launch additional attacks.

A better approach:

All resources not accessible by the current user return `404 Not Found`.

Consider this scenario:

An actor requests a resource by Object ID. Resources that don't exist at all are returned 404. Resources that do exist but aren't accessible return 401.

```
GET /product/P124

404 Not Found

GET /product/P123

401 Not Authorized
```

Inadvertent leakage.

A malicious actor can use this information to determine which resource ids are valid, and which are not assigned.

This information can be used to launch additional attacks.

A better approach:

All resources not accessible by the current user return `404 Not Found`.

Seven Minimal Error Codes

- `200 OK` — Resource found, accessible and available. Action taken and response in payload.
- `201 Created` — Resource was created; payload contains new object (or reference to).
- `304 Not Modified` — No change since last request. Use cached copy.
- `400 Bad Request` — Request was missing required elements. Repair and resubmit.
- `401 Unauthorized` — User has not obtained any token. Include authorization instructions in return body.
- `404 Not Found` — No such resource accessible to current client / user.
- `500 Server Error` — Something went wrong behind the scenes. No change necessary on client.

Response Payloads: Only what the consumer needs to know.

Poor Examples

```
{ "400":"Bad Request" }
```

Provides no context for repair of the query

```
{ "500":"Internal Error",
  "message":"Tibco failure in
querying MySQL database on
10.3.4.33"}
```

Leaks information about internal network configuration

Better Examples

```
{ "400":"Bad Request",
  "message":"See http://docs/api/
the-resource" }
```

```
{ "500":"Internal Error",
  "message":"Report error by
mailing errors@mycorp.com",
  "correlationId":"C4444-23"}
```

Pagination

Avoiding linear memory overflow since the 5th Century.

When should pagination be used?

When to use pagination:

- A large number of results in a list.
- An unknown number of results in a list.
- Results are of the same resource type.
- Queries or listings.

When not to use pagination:

- When the payload is not a list of objects.
- Small, known number of results never to exceed a handful (e.g. 5-10)

Best Practice for Pagination.

Paged resources should have a metadata section that describes, at a minimum, the current page, the number of resources requested, and the number of resources in the current page listing.

```
{
 "pagination": {
  "page": 4,
  "page_size": 10,
  "items": 10
 },
 "items": []
}
```

A Helpful Idea

Include a URL for the next/previous page, so the developer intuitively knows how to fetch them.

```
{
 "pagination": {
  "page": 4,
  "page_size": 10,
  "items": 10,
  "next": "https://.../resource?page=5&page_size=10",
  "prev": "https://.../resource?page=3&page_size=10"
 },
 "items": []
}
```

Best Practice for Pagination.

The item list need not be the full item. Instead, consider what summary information is required for use & display. Include that, plus a link to the full item.

```
{
 "pagination": {
  "page": 4,
  "page_size": 10,
  "items": 10,
  "next": "https://.../resource?page=5&page_size=10",
  "prev": "https://.../resource?page=3&page_size=10"
 },
 "items": [
  { "id":"fjones123", "first_name":"Fred", "last_name":"Jones",
    "href":"https://.../people/fjones123" },
  {...},{...},...
 ]
}
```

THANK YOU