ory.screen?cen ory.screen?cen

hopping.com/d een?category ic

Windows NT 6; 0; a-, ?action=re,

1.1 oduct\_id=AV-SB-

"Mozītīd, T.O 200 1901 "ht (V

11. 89 82.245.228.36

EDDY&JSESS.

een?category,

SHOPPINT

200

## Resource Modeling with REST

Tips for Building a Better REST API

DeAndré Carroll

Sr. Software Engineer, Splunk



### REST Refresher

```
### 1-5W-01" "Open to a serious of the serious of t
```

ory.screen?cen popping.com/a een?category io

d=AA-2B-05"0;

Windows NT

1.1 oduct\_id=AV-033.243 - SB-

82.245.228.36

"Mozīlla" id 200 1901 "http://

PENDAPPENDAPPENDEDER

EDDY&JSESI?

intosh; U; Intel

"Mozilla/4;

een?category

SHOPPINT

200

do?action=re-

APPIEWEDK

#### What is REST?

- Representational State Transfer
- Defined by T. Roy Fielding in 2000 for his PhD dissertation.
- Resource oriented architectural style reflected in the structure of the world wide web.
- Usually implemented in HTTP but does not have to be.



#### Properties of REST?

- Client-Server
- Stateless (Communication)
- Cache

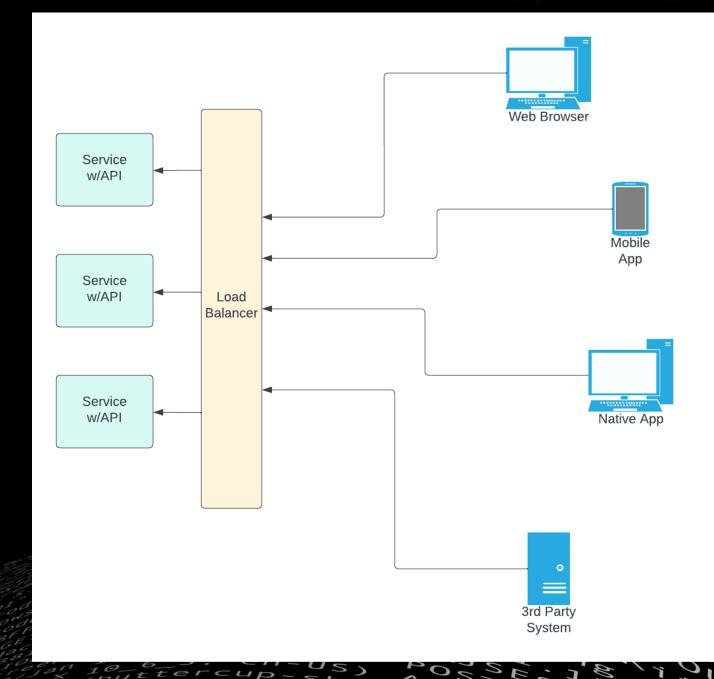
splunk>

- Uniform Interface
- Layered System
- Code-On-Demand



#### **Client-Server**

- Separating the user interface concerns from the data storage concerns
  - Improves portability of the user interface
  - Improves scalability by simplifying the server components.
- Separation allows the components to evolve independently





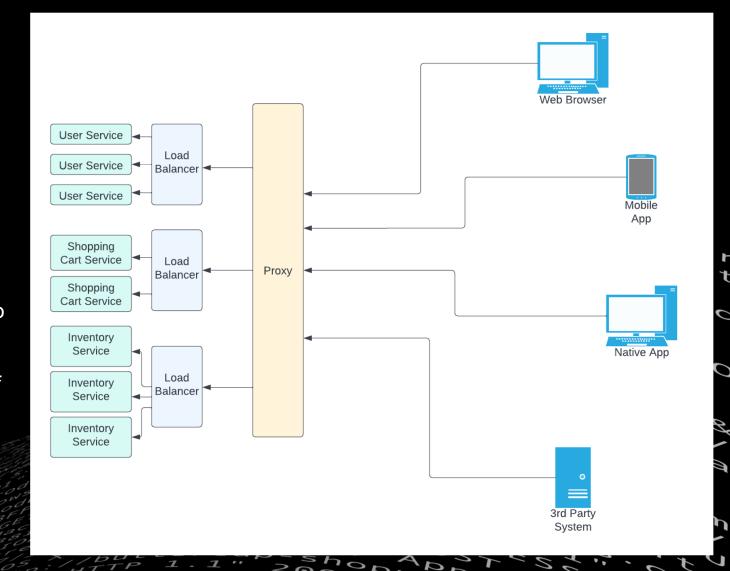
#### Stateless (Communication)

- Induces the properties of:
  - Visibility
  - Reliability
  - Scalability



#### Stateless (Communication): Visibility

- Visibility is "the ability of a component to monitor or mediate the interaction between two other components."
  - "Visible" because each component of the system can "see" details of the request and resource.
  - Allows additional components (e.g. proxies, caches, etc.) to be added to the system because it isn't necessary to examine several requests to determine the context of the current request.





u; Intel

Http

82.245.228.36

gcategory.

SHOPPINT

200

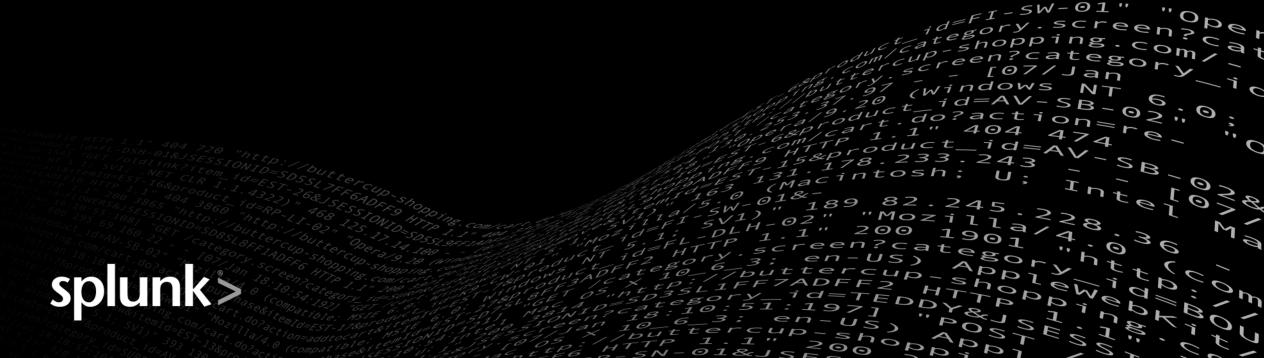
#### Stateless (Communication): Reliability

- Reliability is improved because it eases the task of recovering from partial failures
  - Any failure on the network and server can recover in deterministic way
    - If the problem happens on the server, it can be compensated w- @1" using last state on the client
    - pping gcategory\_io If the problem happens on the client, it can be compensated ws by using the state on the server



#### Stateless (Communication): Scalability

- Scalability is improved by allowing the addition of more servers that don't have to "know" client application state.
  - Application (session) state is therefore kept entirely on the client (No cookies)



id=AV-SB-05:0; a-, ?action=re,

1.1 oduct\_id=AV-oduct\_id=AV-SB-

82.245.228.36

AOZ OO 1901 ?category iccb: ?boppingbki hopping in

"Mozilla, T.O 200 1901 "http://

PCOURT APPLE WEDK

200 POSS P-SHOPPINI

,78.233.243 1ntosh; U; Intel

#### Stateless (Communication): Concerns

- May decrease network performance by increasing the repetitive data (per-interaction overhead) sent in a series of requests (removes shared context)
  - Reduces the server's control over consistent application behavior, since the application becomes dependent on the correct implementation of semantics across multiple client versions ory.
    nopping.com/a egory io



action=re-

ego, Applewed Br Applewebki APPIEW Shopping. 1.1".

ntosh; U; Intel

82.245.228.36

"Mozītes 200 1901 "httl 3catesory id

EDDY & JSEST SEST

SHOPPINT

200

#### Cache

- Used to improve network performance
- Can be cached on client side or server side
- Data within a response to a request is implicitly or explicitly labeled as cacheable or non-cacheable
- Caching can decrease reliability if stale data within the hopping. Company of the cache differs significantly from server data GOM2 NT 6:0;



#### **Uniform Interface**

- REST is defined by four interface constraints:
  - Identification of resources; \* self-descriptive messages
  - Hypermedia as the engine of application state
  - Implementations are decoupled from the services they provide, which encourages independent evolvability
    - The trade-off, though, is that a uniform interface degrades efficiency, since information is transferred in a standardized ws form rather than one which is specific to an application's 404 107 1000 needs

category

SHOPPINT

200

Manipulation of resources through representations



82.245.228.36

"Mozīlla" id 200 1901 "http 2category id

200 PARAL

#### Layered System

Allows an architecture to be composed of hierarchical layers by constraining component behavior such that each component cannot "see" beyond the immediate layer with which they are interacting

#### Code-On-Demand

- Allows client functionality to be extended by downloading and properties of secretary and secretary
- Reduces visibility, and thus is only an optional constraint within the second constraint within the s REST.



## Good REST API Design

```
### 1-5W-01" "Open to a temporal property of the property of t
```

ory.screen?cen

nopping.com/a een?category io

(Windows NT 6; 0; id=AV-SB-02; 0;

do?action=re do?action=re 404 474 1.1 oduct\_id=AV-SB-

intosh; U; Intel

" "MC
" 200 1901
" 200 1901
" een?category icto:
en-us) Applewides
on-up-shopping
ing "Mozīlla" T.O 200 1901 "ht (c

EZ HTTP PRINEDK I "POSTESS"

"Mozilla/4.

7209 (2U-n: 720A (2U-n: 744A (2U-n:

82.245.228.36

#### Start with your API

- API design should not dictate your architecture, but it may likely inform it
- If you are working with an existing system, establish your API as a "facade"
- Think about the capabilities that need to be exposed to your client



# "It's all the same. Only the names will change." -Jon Bon Jovi

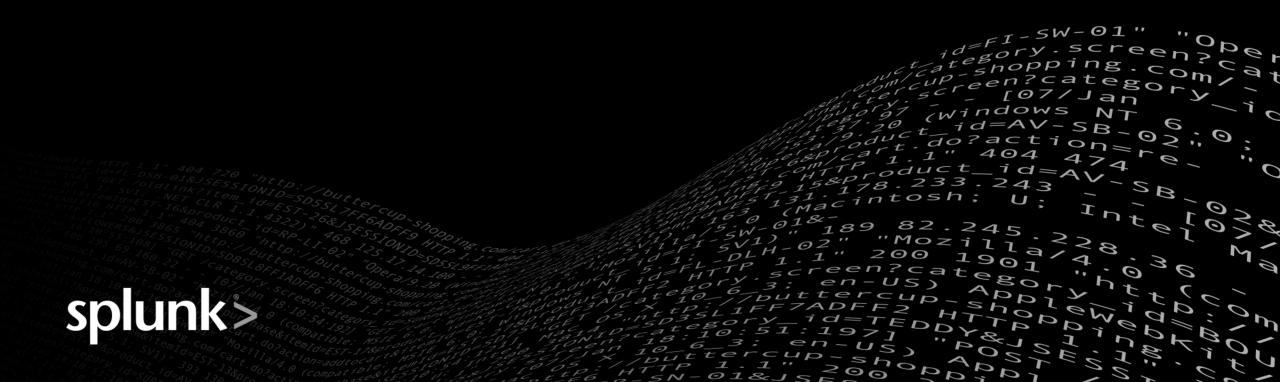
```
Splunk>

Splunk

Splu
```

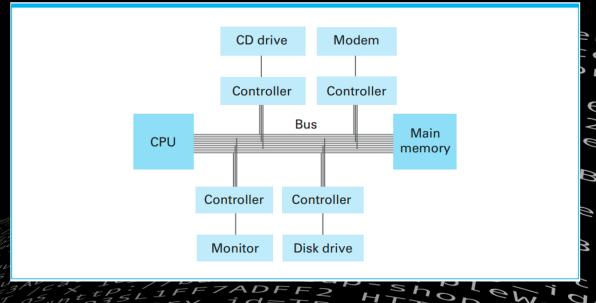
#### Computational System Design Models

- Microcomputer architecture
- Unix operating system



#### Computational System Design Models

- Microcomputer architecture
  - The core processing unit (CPU/ALU)
     accesses all resources through an address
     bus.
    - Memory
    - Controllers





ory.screen?cen ory.screen?cen ory, nopping.com/a

een?category io7/Jan

d=AV-SB-05:0;

egory Applewedelec Bobbinebki

(Windows NT

1.1 oduct\_id=AV-SB.

82.245.228.36 82.243. "Mozilla/4.036 200 1901 ".0 (6

SCONDER APPLICATION OF THE STATE OF THE STAT

intosh; U; Intel

een?category

SHOPPINT

200

do?action=re-

#### Computational System Design Models

- Unix operating system
  - Everything is a file
    - Directories (/home/user)
    - Devices (/dev)
    - Links (made with 'ln' command)
    - Sockets and named pipes (special files for passing information)



ory.screen?cen nopping.com/a

een?category io7/Jan

(Windows NT 6; 0;

egory AppleWebRBC Bobbinski

do?action=re-

1.1 oduct\_id=AV.SB.

"Mozīlla, f. 0 200 1901 "ht (,

US) APEROPOREDE CUP-SHOPPINEDE F2 HTTP 1 NE F2 LCE: 1"

HTTP SPACION TO SECULAR SECULA

intosh; U; Intel

"Mozilla/4.036

een?category,

SHOPPINT

200

#### REST Design Tips

- Model your process not the implementation
- Design resources and representations that reflect your business logic
- Design resources as nouns (not verbs or prepositions)
- Design resources and representations to be composable



y.screen?cat

eping eategory io

Intel

action=re-

82.245.228.36

"Mozīlea" i. 0 200 1901 "http 3category id

EDDY

200

SHOPPIN

#### Model your process not the implementation

- Critical processes (i.e. resource ID assignment) should be isolated to the server
- Errors should reflect business logic problems (only implementation/system problems when absolutely necessary)



y.screen? y.screen? on.com/at popping.com/d

een?category ic

d=AA-2B-05:0

n: Infel

action=re-

Windows NT

82.245.228.36

"Mozīlla" id 200 1901 "http

"Mozilla/4

EDDY

200

SHOPPIN

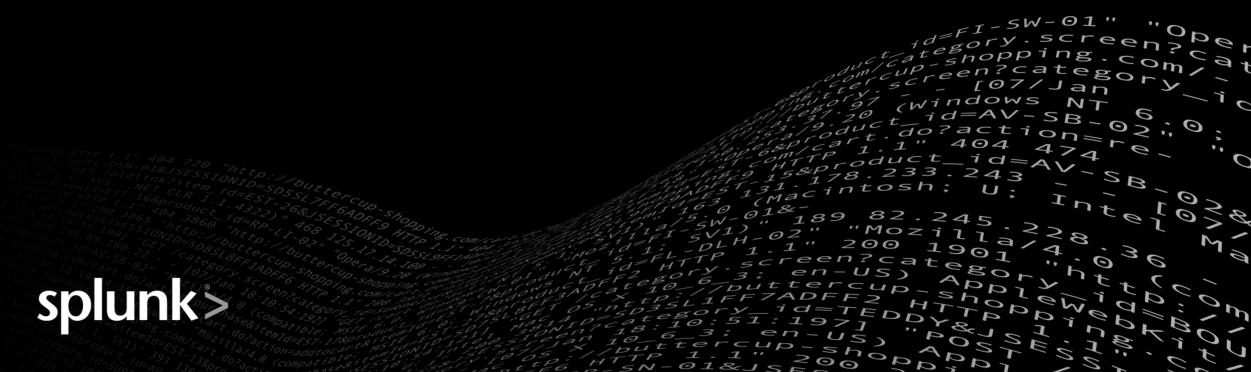
### Design resources and representations that reflect your business logic

- Keep in mind
  - Creation and change properties
    - Atomic transaction/immediate consistency vs eventual consistency
    - Synchronous vs asynchronous
  - Network efficiency
  - Size of representations
  - Client convenience to guide resource granularity



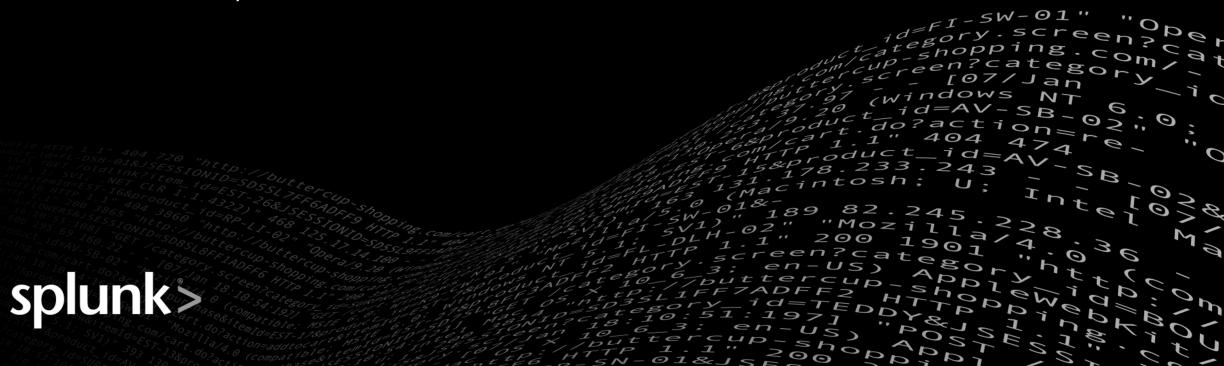
# Design resources and representations that reflect your business logic

- Avoid modeling
  - Your persistence schema
  - Other traits of your implementation



## Design resources as nouns (not verbs or prepositions)

 Resources can be designed in ways similar to objects in OOP (not necessarily nouns in the strict sense)

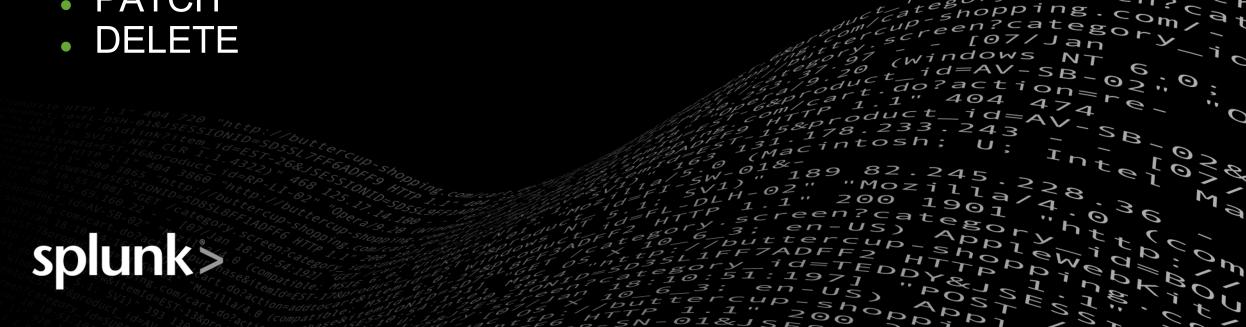


ory.screen?cen

nopping.com/a

#### Know your verbs, their design, and their capabilities (the Uniform Interface)

- POST
- GET
- PUT
- PATCH
- DELETE



ory.screen?cer

nopping.com/a een?category ieen?category ic

d=AA-2B-05:0;

egory Applewed Br Book in Chi

APPLE WELL SHOP IN THE SHOP IN

(Windows NT

1.1 oduct\_id=AV-SB.

82.245.228 MOZILIA/4.036

"Mozīlla" i o 200 1901 "http://

ZONING TESSING TO STATE TO STA

ntosh; U; Intel

"Mozilla/4.

een?category

SHOPPINT

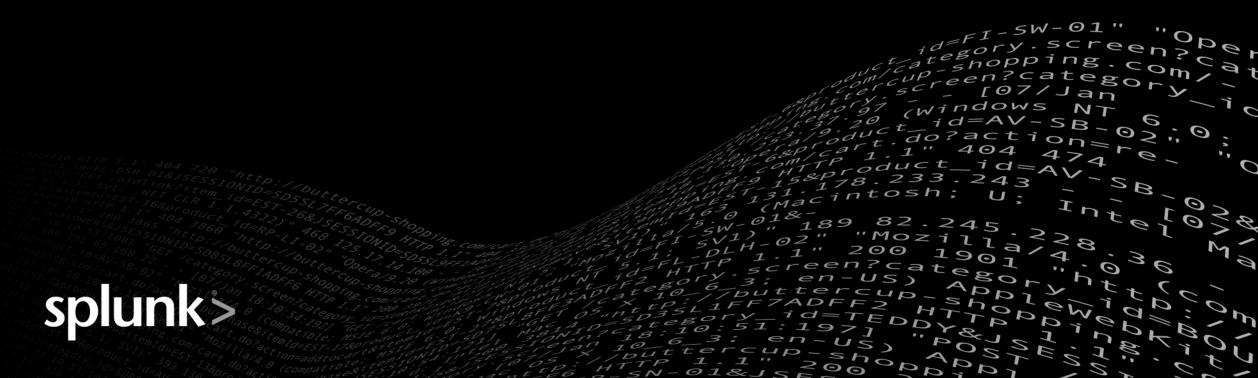
200

do?action=re-

- To create a new resource, using the resource as a factory
- To submit input to any process that would be asynchronous
- To run queries with large inputs
- To perform any unsafe or non-idempotent operation when no other HTTP method seems appropriate
- Basically, anything POSTed to a resource becomes a subordinate of that resource



- To create a new resource, using the resource as a factory
  - System specific values (i.e. unique IDs) should be generated by the resource server to prevent business logic from implemented in the client



ory.screen?cen

Intel

popping.com/d een?category ic

d=AA-2B-05"0

action=re-

82.245.228.36

category.

SHOPPINT

200

- To submit input to any process that would be asynchronous
  - Commands
  - CRUD for resources that are eventually consistent
    - Use a batch processing model
      - 1. POST to a resource that creates a progress resource
      - 2. The progress resource shows the completion state of the request
      - 3 Upon completion, a request against the progress resource redirects to a completed resource
      - 4. As an alternative, asynchronous notification can, be issued via a configured webbook or websocket



- To run queries with large inputs
  - Safer than using query variables in a URI
    - Used to be used as a server attack method, inducing a buffer overflow



- To perform any unsafe or non-idempotent operation when no other HTTP method seems appropriate
  - Processes that yield side effects are executed this way



ory.screen?cen

nopping.com/a een?category io7/Jan

d=AA-2B-05"0;

Windows NT

1.1 oduct\_id=AV-233.243 - SB-

82.245.228 30zilla/4.036

"Mozīled' † . 0 200 1901 "http://

ZONING TESSING TO STATE TO STA

intosh; U; Intel

"Mozilla/4.

SHOPPINT

200

do?action=re-

egoly debk

#### **Using GET**

- For safe and idempotent (no side effects) information retrieval of resources
- To fetch a representation containing the output of the processing function or data over a continuous domain
  - Map data, mathematical functions, etc
  - Use query parameters to supply inputs



#### **Using PUT**

- To update named individual resources in an idempotent (no side effects) way
  - Best for resources that already exist on the server
  - Process should be atomic (immediately consistent)



ory.screen?cen oning.com/

nopping.com/a een?category io7/Jan \io

(Windows NT 6; 0; id=AV-SB-02; 0;

egory AppleWedWeb Bobbincki

do?action=re-

intosh; U; Intel

82.243. "Mozilla/4.036 200 1901 ".0 (6

PER APPLE VER ES HTTP 1 NEW CORP. SHOP DINE TO SEC. 12:

EDDY&JSESI?

89 82.245.228 "Mozilla/4.0<sup>3</sup>6

een?category

SHOPPEN

200

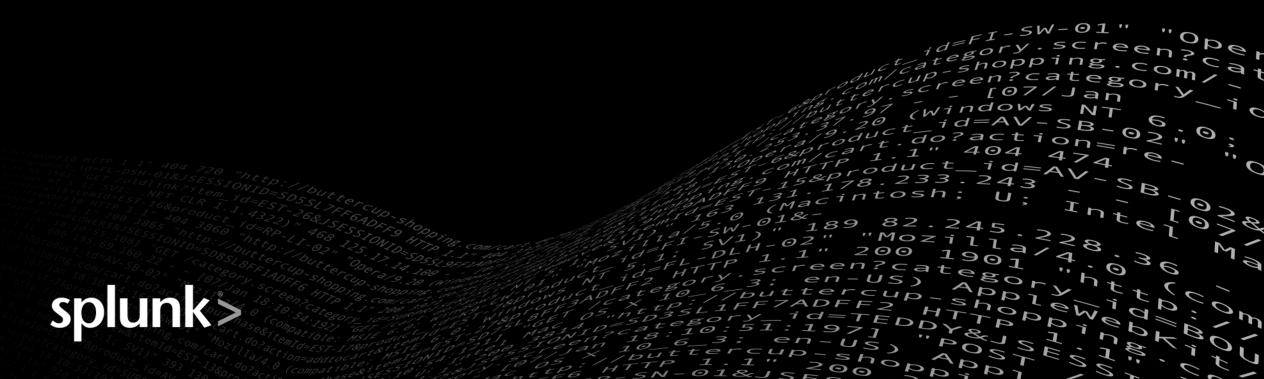
#### **Using PATCH**

- To update resources in an idempotent, nonconstrained way
- Do NOT use
  - If the underlying operation is not atomic/non transactional
  - If the input is constrained to a well-defined, unchanging representation
    - Use PUT instead



#### Using DELETE

- To remove individual resources from visibility
  - Can be removed from the system
  - Can be archived
- May be implemented asynchronously



ory.screen?cen

popping.com/a

Http

een?category ic

=AV-SB-02:

action=re-

82.245.228.36

"Mozilla/4

?category

200 1901

EDDY

200

SHOPPINT

#### Let's Build an API

We're building a system that keeps track of users and their mailing addresses. The user information consists of...

- First name
- Last name
- Email address
- Phone
- Mailing address
  - Street 1
  - Street 2
  - State/Province
  - Postal code
  - Country

#### Part 1:

Implement CRUD (Create, Retrieve, Update, Delete) for the data resource as represented here.

What does the URI look like for a collection? An individual resource?

What does the representation look like?



ory.screen?cen

Intel

Http

nopping.com/a

een?category io7/Jan

d=AV-SB-02:

action=re-

id=AV-SB-

APPIEWEDK

Windows NT

82.245.228.36

"Mozilla/4

?category

200 1901

EDDY

200

SHOPPINT

#### Let's Build an API

We're building a system that keeps track of users and their mailing addresses. The user information consists of...

- First name
- Last name
- Email address
- Phone
- Mailing address
  - Street 1
  - Street 2
  - State/Province
  - Postal code
  - Country

#### Part 2:

Implement CRUD (Create, Retrieve, Update, Delete) for two data resources, one for the user and one for mail.

How are the two linked?



y.screen?cat

pping ?category\_io

=AV-SB-05:0

action=re-

8.233.2.3 ntosh; U; Intel

"Mozītega" i o (, 200 1901 "http://ocategory.io

EDDY

200

SHOPPIN

82.245.228.36

### Design resources and representations to be composable

- Resource representations can be composed into a larger resource for easier access
  - This should be driven by monitoring client usage patterns
- Remember the user/mail example

