Module 2: Translating Functional Requirements for APIs
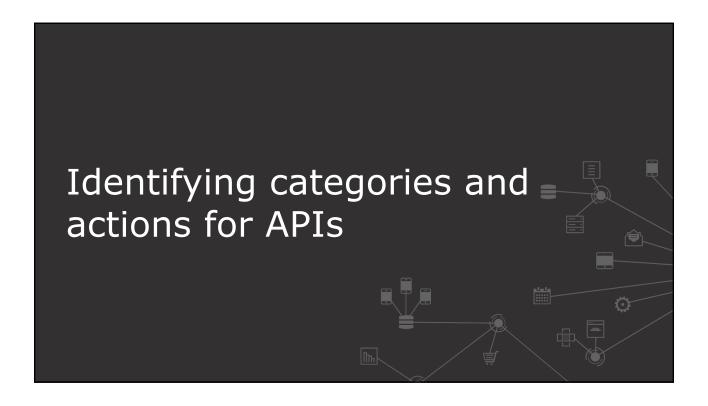
MuleSoft®

Goal

MuleSoft

ACME Banking API User Functionality

This document helps build API consumer stories to see how they will consume the API. API consumers can be involved at this stage to find new use cases for the API usage and prioritize features important to them.

ACME bank has customers(individuals, companies etc.) who want to perform actions with regards to their bank accounts. The accounts help them manage and handle their finances (using transactions).

As a developer, who is the API consumer at ACME Bank and is involved in building a web application for bankers to serve customers, what are the functions they should be able to perform with the API?

Categories:

CUSTOMERS — As a developer, one should be able to access and manipulate customer information.

i.   Get list of all customers in the bank
ii.   Register a new customer
iii.   Get customer information for a specific customer ID
iv.   Update customer information for a specific customer ID
v.   Delete a customer with a specific customer ID

ACCOUNTS — Since customers are associated with bank accounts, developers using the ACME Banking API should be able to work with account information.

Translating categories and actions into resources and methods —

CUSTOMERS: Resource /customers

i. Get list of all customers in the bank	Resource /customers	Method GET
ii. Register a new customer	Resource /customers	Method POST
iii. Get customer information for a specific customer ID	Resource /customers/{customer_id}	Method GET
iv. Update customer information for a specific customer ID	Resource /customers/{customer_id}	Method PATCH
v. Delete a customer with a specific customer ID	Resource /customers/{customer_id}	Method DELETE
vi. Get list of all accounts for a specific customer ID	Resource /customers/{customer_id}/accounts	Method GET

ACCOUNTS: Resource /accounts

i. Create a new account	Resource /accounts	Method POST
ii. Get account information for a specific account ID	Resource /accounts/{account_id}	Method GET
iii. Delete an account with a specific account ID	Resource /accounts/{account_id}	Method DELETE
iv. Update account information for a specific account ID	Resource /accounts/{account_id}	Method PUT
ii. Get transactions for a specific account ID	Resource /accounts/{account_id}/transactions	Method GET

TRANSACTIONS: Resource /transactions

i. Create a new transaction	Resource /transactions	Method POST
iii. Get transaction information for a specific transaction ID	Resource /transactions/{transaction_id}	Method GET

2

1

At the end of this module, you should be able to

MuleSoft

- Identify the different categories and actions for a REST API
- Convert categories to resources
- Select HTTP methods to support the actions on categories

3

Identifying categories and actions for APIs

Defining categories and actions for APIs

MuleSoft

- Categories
  - Define and group information into entities with common characteristics
  - Always a noun
    - Identified from the requirements and the functionality that the API should serve
- Actions
  - Do something with a category
  - Always a verb

5

Walkthrough 2-1: List the categories and actions for an API

MuleSoft

- Identify the categories for a banking API
- Define actions for the categories to decide how users will consume the API

ACME Banking API User Functionality

This document helps build API consumer stories to see how they will consume the API. API consumers can be involved at this stage to find new use cases for the API usage and prioritize features important to them.

ACME bank has customers(individuals,
The accounts help them manage and ha

As a developer, who is the API consu
customers, what are the functions th

TO-DO 1:
    Identify the categories in th

TO-DO 2:
    Add a one-line summary as to

TO-DO 3:
    Expand the one-line summary t

ACME Banking API User Functionality

This document helps build API consumer stories to see how they will consume the API. API consumers can be involved at this stage to find new use cases for the API usage and prioritize features important to them.

ACME bank has customers(individuals, companies etc.) who want to perform actions with regards to their bank accounts. The accounts help them manage and handle their finances (using transactions).

As a developer, who is the API consumer at ACME Bank and is involved in building a web application for bankers to serve customers, what are the functions they should be able to perform with the API?

Categories:

CUSTOMERS — As a developer, one should be able to access and manipulate customer information.

i.   Get list of all customers in the bank
ii.   Register a new customer
iii. Get customer information for a specific customer ID
iv.   Update customer information for a specific customer ID
v.   Delete a customer with a specific customer ID

ACCOUNTS — Since customers are associated with bank accounts, developers using the ACME Banking API should be able to work with account information.

i.   Get list of all accounts for a specific customer ID
ii.   Create a new account
iii. Get account information for a specific account ID
iv.   Delete an account with a specific account ID
v.   Update account information for a specific account ID

TRANSACTIONS — Bank accounts contain transactions and activities like withdrawal, deposit, interest earnings etc., and developers should be able to perform actions on the transaction information.

i.   Create a new transaction
ii.   Get transactions for a specific account ID
iii. Get transaction information for a specific transaction ID

6

Translating categories into resources

Introducing resources

MuleSoft

- Resources are the primary way a client interacts with an API
  - Resource names are derived from categories
  - Resources are represented with a / followed by the name
    - For example: /users is a resource that represents the collection of users and can be used to retrieve, create and modify user data
- Multiple actions can be performed on a single resource
- Nested resources are used to call a subset or member of the main resource
  - For example: /users/{userID} is used to retrieve the user information of a particular user with a specific ID
  - The curly braces around the userID nested resource represents a URI parameter with dynamic values for the field
  - Nested resources need not contain URI parameters always

8

Best practices for creating resources

Ⓜ MuleSoft

- Decoupled architecture
  - – Don't tie resources to specific methods in the backend
  - – Changes in the backend will not change consumer interactions with the API
- Nouns
  - – Using verbs as a part of the resource name indicates tight coupling with a specific action the resource might perform (similar to RPC) - avoid it
  - – Use plural form of nouns, if possible
    - Allows interaction with a collection (multiple records), referred to as collection resources
    - To access a member resource (single record), represent them as a nested resource
- Support multiple content types
  - – Results in an extremely flexible and usable API

All contents © MuleSoft Inc.

9

Translating actions into HTTP methods

Introducing HTTP methods

MuleSoft

- HTTP methods are derived from the actions described for categories
- HTTP methods transport resource information and instructions between systems and applications

11

Commonly used methods in HTTP/1.1

MuleSoft

- GET
  - Used to request data
- POST
  - Creates a new object within a collection
  - Transmits the object in the body of the request
  - Sends the URI of the object created, in the response
- PUT
  - Updates an object or creates a new one if it does not exist
  - Overwrites an existing object
    - Critical to send the entire object data in the request, else it replaces missing data with null
  - Should not be used as a POST method to create new objects

12

Other supported methods in HTTP/1.1

MuleSoft

- PATCH
  - Makes partial update to an object
  - Not required to send the entire object in the request
- DELETE
  - Deletes an object
  - Can be very dangerous to use on collections and thus should be avoided or greatly limited (similar to PUT and PATCH)
- OPTIONS
  - Returns the quick list of methods that are allowed on a specific resource
- HEAD
  - Used to get information about a particular data and not the data itself

All contents © MuleSoft Inc.

13

Walkthrough 2-2: Translate categories and actions into resources and methods

MuleSoft

- Translate categories into resources
- Identify the HTTP methods for the actions

```
Translating categories and actions into resources and methods —

CUSTOMERS:    Resource _____
i.   Get list of all customers in the bank                          Resource _____      Method _____
ii.  Register a new customer                                        Resource _____      Method _____
iii. Get customer information for a specific customer ID            Resource _____      Method _____
iv.  Update a customer information for a specific customer ID       Resource _____      Method _____
v.   Delete a customer  with a specific customer ID                 Resource _____      Method _____

ACCOUNTS:     Resource _____
i.   Get list of all accounts for a specific customer ID           Resource _____      Method _____
ii.  Create a new account                                          Resource _____      Method _____
iii. Get account information for a specific account ID             Resource _____      Method _____
iv.  Delete an account with a specific account ID                  Resource _____      Method _____
v.   Update account information for a specific account ID          Resource _____      Method _____

TRANSACTIONS: Resource _____
i.   Create a new transaction                                      Resource _____      Method _____
ii.  Get transactions for a specific account ID                    Resource _____      Method _____
iii. Get transaction information for a specific transaction ID     Resource _____      Method _____
```

All contents © MuleSoft Inc.

Summary

Summary

- Categories are nouns used to represent resources in REST APIs
- Actions are HTTP methods performed on resources
- Nest resources when you want to receive or manipulate a single member object data
- Decouple actions from the resource names to reduce dependencies with backend services

16