



# Power Platform

## COMMUNITY CONFERENCE

SEPTEMBER 18–20, 2024 • *Workshops: Sept 16, 17 & 21*  
MGM GRAND • *Las Vegas, NV*



# Power Platform COMMUNITY CONFERENCE

## **REST Assured: Powering Up the SharePoint REST API with Power Automate**

Fausto Capellan, Jr



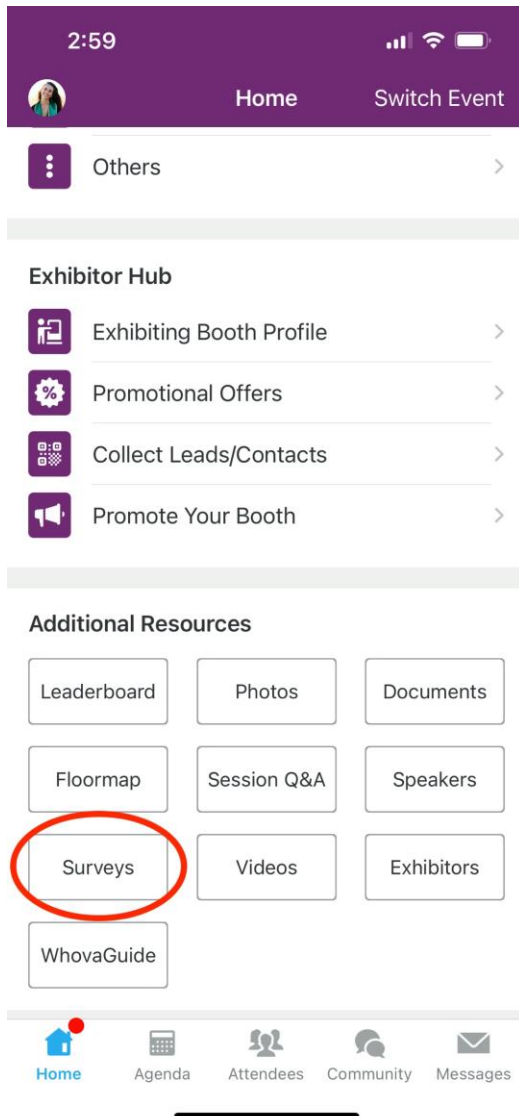
The official event app for the **Power Platform Community Conference**



Join the event app to access:

- ➔ Event announcements
- ➔ Personalized agenda, session details
- ➔ Speaker & attendee profiles
- ➔ Networking, meet-ups, messages
- ➔ Event documents

**Event Invitation  
Code: PPCConf2024**



# Session Feedback Surveys

*We really want to hear from YOU!*

*In the pursuit of making next year's Power Platform Community Conference even better, we want to hear your feedback about this session.*

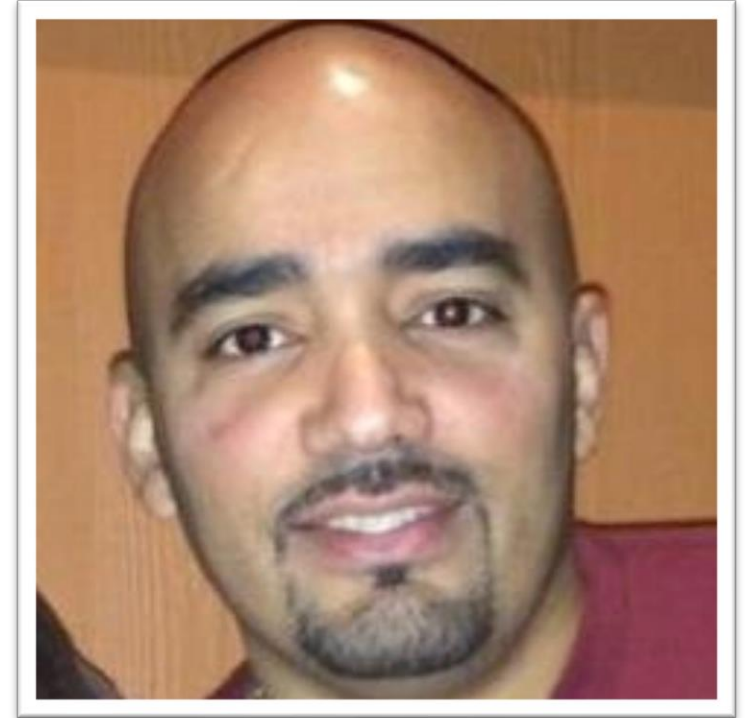
## ***Here's How -***

- *Simply go to the Whova App on your smartphone*
- *Scroll down on the Power Platform Community Conference Homepage to 'Additional Resources' to click "Surveys'.*
- *Click Session Feedback.*
- *Scroll down to find this session title.*
- *Complete the session feedback survey.*
- *Finally, click 'Submit'*

*It's just that easy!*

# About Me

- Consultant @ PowerApps911
- Microsoft MVP – Business Applications
- Power Platform Advocate
- Active Community Member
- Traveler



# Agenda

- Overviews
- How the SharePoint REST service works
- HTTP Request Methods
- Determine SharePoint REST service endpoints
- OData Query Operations
- Browser Extensions
- Navigate data structure
- Why use Send an HTTP Request to SharePoint action over pre-built actions
- Demo data structure and OData queries
- Demos

A close-up photograph of a group of people gathered around a silver laptop. One person's hand is resting on the keyboard, while another person's hand is pointing at the screen. The background is blurred, showing other people in a professional setting. A semi-transparent white banner is overlaid across the middle of the image, containing the word "Overviews" in a bold, black, sans-serif font.

# Overviews

# REST Overview

- Stands for REpresentational State Transfer
- It is an architectural style for designing networked applications
- It is neither a protocol nor a standard
- RESTful systems use standard HTTP methods (GET, POST, PUT, DELETE) to Create, Read, Update, and Delete resources (CRUD)
- Resources are represented in JSON or XML format
- RESTful systems are very popular because they promote
  - Scalability
  - Simplicity
  - Modifiability



# SharePoint REST API Overview

- It is a powerful and efficient method to interact with SharePoint data remotely
- It allows developers to effectively integrate SharePoint functionality into their own applications
- Developers can use it to perform Create, Read, Update, and Delete operations on SharePoint entities such as sites, lists, and document libraries

# OData Overview

- Stands for Open Data Protocol
- It is a standard protocol for building and consuming RESTful APIs
- It enables the creation of REST-based services where resources are identified using Uniform Resource Locators (URLs) and defined in a data model
- It allows developers to perform Create, Read, Update, and Delete operations
- Web clients can publish and edit these resources using HTTP messages



# How the SharePoint REST Service Works

# How the SharePoint REST Service Works

- It provides the ability to interact with SharePoint resources using technology that supports standard REST capabilities
  - Power Automate
- SharePoint resources are accessed by constructing a RESTful HTTP request using OData standard

## CSOM

C#

```
var items = List.GetByTitle(listname).GetItems();
```

## REST

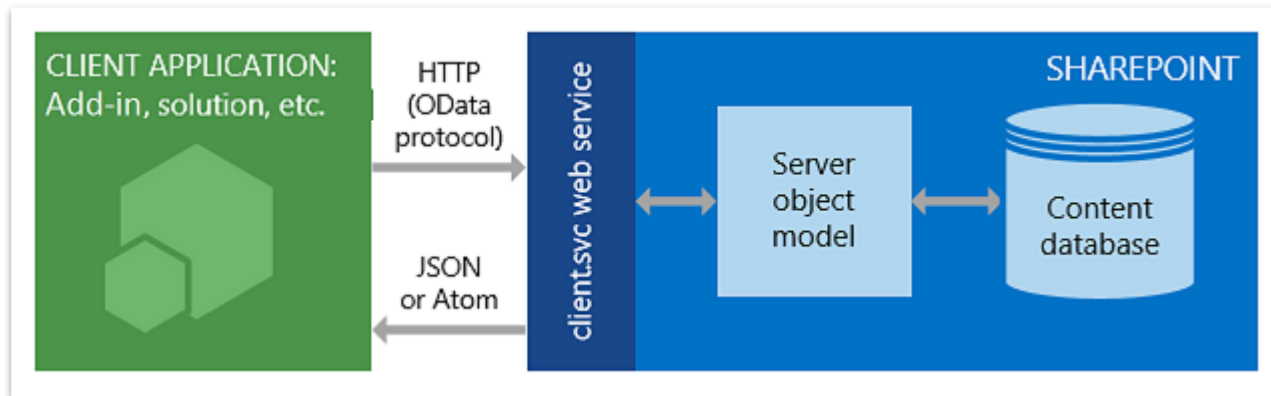
HTTP

```
GET https://{site_url}/_api/lists/getbytitle('{list_name}')/items
Authorization: "Bearer " + accessToken
Accept: "application/json;odata=verbose"
```

# How the SharePoint REST Service Works

- The client.svc web service in SharePoint handles the HTTP request and serves the response in either Atom(XML) or JavaScript Object Notation (JSON) format
- The response must be parsed by the client application

## SharePoint REST Service Architecture



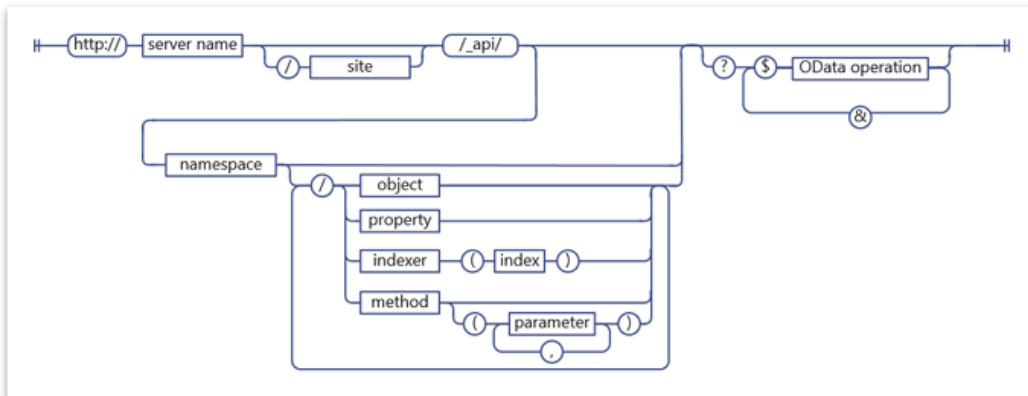
# HTTP Request Methods

HTTP Request	Action
GET	Read a resource
POST	Create or update a resource
PUT	Update or insert a resource
DELETE	Delete a resource

# Determine SharePoint REST Service Endpoints

- Must figure out the URI endpoint for the desired resource
- The URI endpoints closely match the SharePoint Client Object Model counterparts whenever possible
- In some cases, the URI endpoints are different from their SharePoint Client Object Model counterparts to comply with REST or OData conventions

## SharePoint REST Service Architecture





# Determine SharePoint REST Service Endpoints

Constructing a REST endpoint for a SharePoint resource

1. Accessing a specific site collection

[http://{site\\_url}/\\_api/site](http://{site_url}/_api/site)

2. Accessing a specific site

[http://{site\\_url}/\\_api/web](http://{site_url}/_api/web)

3. Accessing a specific list

[http://{site\\_url}/\\_api/web/lists/getbytitle\('ListDisplayName'\)](http://{site_url}/_api/web/lists/getbytitle('ListDisplayName'))





# OData Query Operators

# OData Query Operators

SharePoint REST service supports a wide range of OData query operators that enable selecting, filtering, and order the requested data

- **\$select** returns the selected fields for a given list or list item  
[http://{site\\_url}/\\_api/web/lists/getbytitle\('ListDisplayName'\)/items?\\$select=Title,Name](http://{site_url}/_api/web/lists/getbytitle('ListDisplayName')/items?$select=Title,Name)
- **\$filter** returns the selected item based on the specified filter  
[http://{site\\_url}/\\_api/web/lists/getbytitle\('ListDisplayName'\)/items?\\$filter=Title eq 'String'](http://{site_url}/_api/web/lists/getbytitle('ListDisplayName')/items?$filter=Title eq 'String')
- **\$orderby** sorts the returned items based on the specified field  
[http://{site\\_url}/\\_api/web/lists/getbytitle\('ListDisplayName'\)/items?\\$orderby=Title asc](http://{site_url}/_api/web/lists/getbytitle('ListDisplayName')/items?$orderby=Title asc)

# OData Query Operators

SharePoint REST service supports a wide range of OData query operators that enable selecting, filtering, and order the requested data

- **\$top** returns the first number of items indicated

[http://{site\\_url}/\\_api/web/lists/getbytitle\('ListDisplayName'\)/items?\\$top=5](http://{site_url}/_api/web/lists/getbytitle('ListDisplayName')/items?$top=5)

- **\$skiptoken** skips over the number of items indicated

[http://{site\\_url}/\\_api/web/lists/getbytitle\('ListDisplayName'\)/items?\\$skiptoken=Paged=TRUE  
&p\\_ID=5](http://{site_url}/_api/web/lists/getbytitle('ListDisplayName')/items?$skiptoken=Paged=TRUE&p_ID=5)

# OData Query Operators

## Supported OData query operators

### Numeric Comparisons

- lt (Less Than)
- le (Less Than or Equals To)
- gt (Greater)
- ge (Greater Than or Equals To)
- eq (Equals)
- ne (Not Equals)

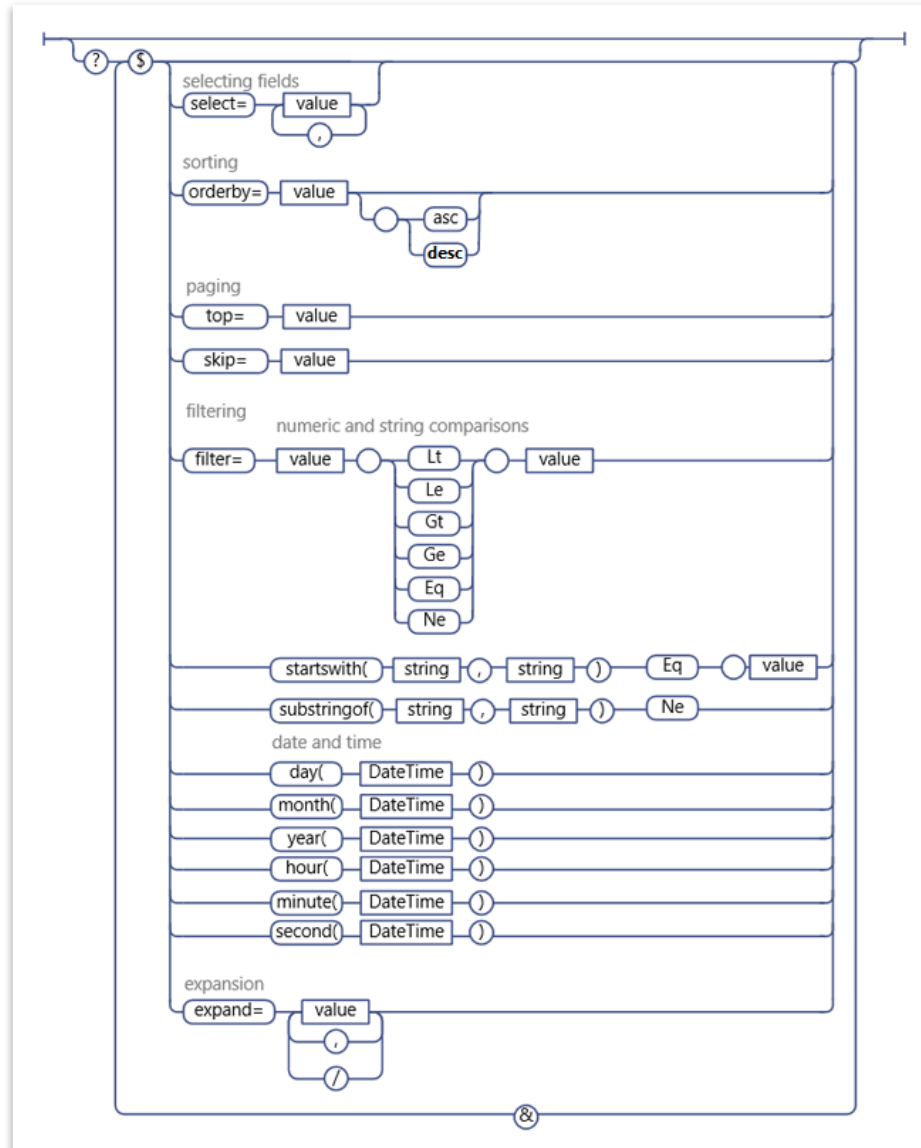
### String Comparisons

- eq (Equals)
- ne (Not Equals)
- startswith(colToQuery,'StringToCheck')
- substringof('StringToCheck',colToQuery)

### Date and Time

- day()
- month()
- year()
- hour()
- minute()
- second()

# OData Query Operators





# Browser Extensions

## XML Tree

Featured 4.0 ★ (554 ratings)

Extension Developer Tools 100,000 users

Click entity for XPath. Double-click to collapse/expand. Enter XPath or XML string then click XPath1/Parse for results or to XML Tree-

XPath1/Parse  
XSL 1.0

```
<author>
  <name/>
</author>
<content type="application/xml">
  <m:properties>
    <d:FileSystemObjectType m:type="Edm.Int32">1</d:FileSystemObjectType>
    <d:Id m:type="Edm.Int32">4</d:Id>
    <d:ServerRedirectedEmbedUri m:null="true"/>
    <d:ServerRedirectedEmbedUri/>
    <d:ContentTypeId>0x0120008915EE226E4DBC4A9E56B684DCC55DAC</d:ContentTypeId>
    <d:ComplianceAssetId m:null="true"/>
    <d:Title>Number 1</d:Title>
    <d:DocumentSetDescription m:null="true"/>
    <d:FlowUserId m:null="true"/>
    <d:FlowUserStringId m:null="true"/>
    <d:Category>Number 22</d:Category>
    <d:ColumnFormattingV1 m:null="true"/>
    <d:MMD m:type="SP.Taxonomy.TaxonomyFieldValue">
      <d:Label>1</d:Label>
      <d:TermGuid>371abe21-a93e-4b81-8248-f7a781fea00d</d:TermGuid>
      <d:WssId m:type="Edm.Int32">1</d:WssId>
    </d:MMD>
    <d:OData__ColorTag m:null="true"/>
    <d:MediaServiceImageTags m:type="Collection(SP.Taxonomy.TaxonomyFieldValue)"/>
    <d:MediaServiceOCR m:null="true"/>
    <d:ID m:type="Edm.Int32">4</d:ID>
    <d:Created m:type="Edm.DateTime">2019-06-17T14:49:59Z</d:Created>
    <d:AuthorId m:type="Edm.Int32">6</d:AuthorId>
    <d:Modified m:type="Edm.DateTime">2021-07-21T10:40:24Z</d:Modified>
```

## {≡} JSON Viewer

Featured 4.5 ★ (1.1K ratings)

Extension Developer Tools 2,000,000 users

```
"body": {
  "value": [
    {
      "@odata.etag": "\"1\"",
      "ItemInternalId": "7",
      "ID": 7,
      "Title": "_PrimaryRed",
      "ColorHex": "#e60000",
      "Modified": "2020-06-05T15:13:15Z",
      "Created": "2020-06-05T15:13:15Z",
      "Author": {
        "@odata.type": "##Microsoft.Azure.Connectors.SharePoint.SPListExpandedUser",
        "Claims": "i:0#.f|membership|fausto@vecaent.com",
        "DisplayName": "Fausto Capellan Jr",
        "Email": "fausto@vecaent.com",
        "Picture": "https://[redacted].sharepoint.com/sites/flow/_layouts/15/UserPhoto.aspx?Size=L&AccountName=fausto@vecaent.com",
        "Department": null,
        "JobTitle": "Owner"
      },
      "Author#Claims": "i:0#.f|membership|fausto@vecaent.com",
      "Editor": {
        "@odata.type": "##Microsoft.Azure.Connectors.SharePoint.SPListExpandedUser",
        "Claims": "i:0#.f|membership|fausto@vecaent.com",
        "DisplayName": "Fausto Capellan Jr",
        "Email": "fausto@vecaent.com",
        "Picture": "https://[redacted].sharepoint.com/sites/flow/_layouts/15/UserPhoto.aspx?Size=L&AccountName=fausto@vecaent.com",
        "Department": null,
        "JobTitle": "Owner"
      },
      "Editor#Claims": "i:0#.f|membership|fausto@vecaent.com",
      "{Identifier}": "Lists%252fMoreColors%252f7_.000",
      "{IsFolder}": false,
      "{Thumbnail}": {
        "Large": null,
        "Medium": null,
        "Small": null
      }
    }
  ]
}
```

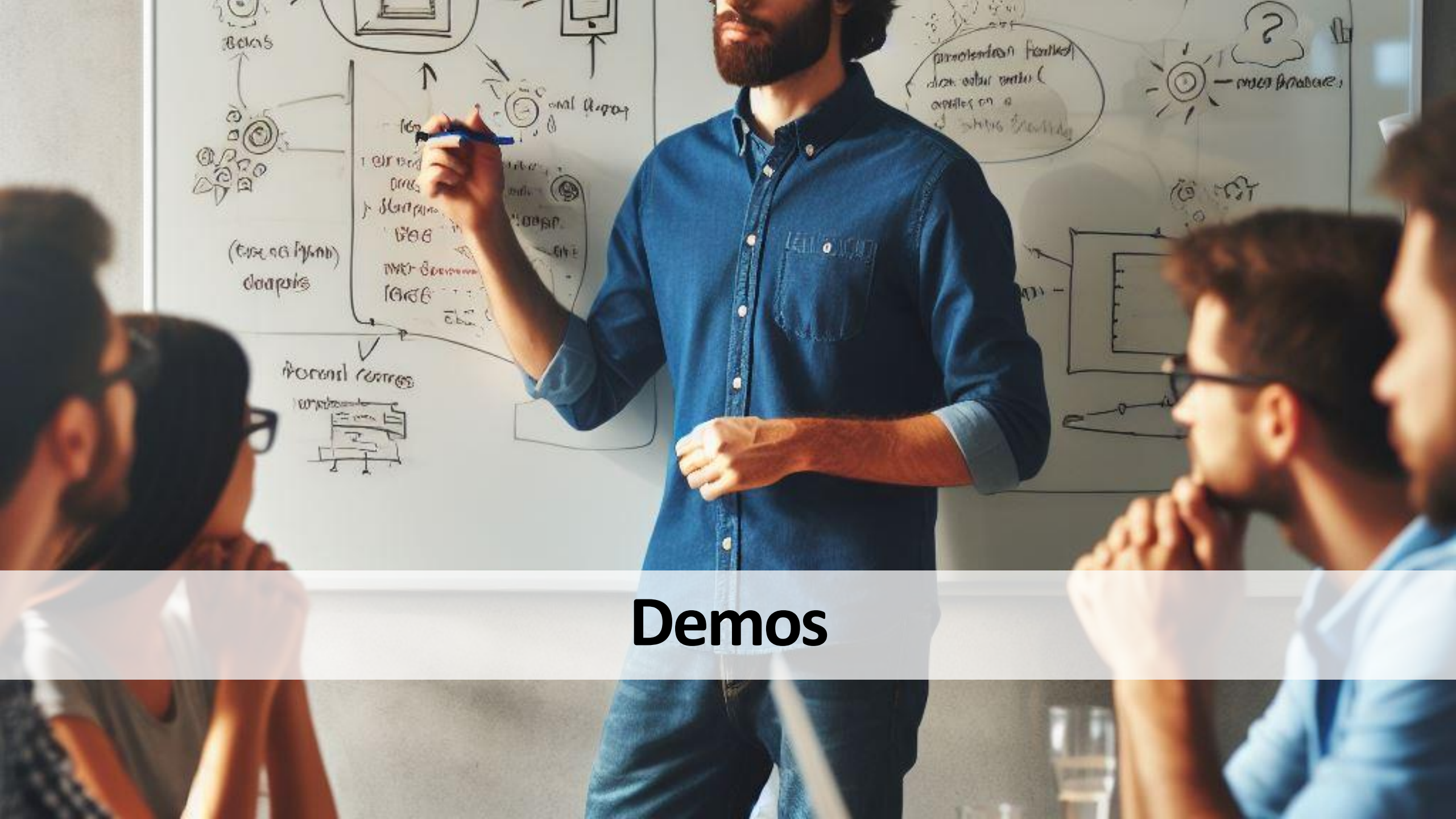
A man with brown hair and a beard, wearing a brown long-sleeved shirt, stands in front of a chalkboard. He is pointing with a piece of chalk at a box labeled 'COMPARE' which contains a checklist. The chalkboard is filled with various hand-drawn diagrams and text boxes. To the left of 'COMPARE' is a box labeled 'FEATURE' containing a list of items with checkboxes. To the right is a box labeled 'PUT TO TEST EASY' containing a list of items. Below 'COMPARE' is a box with a heart icon and the word 'COMPARE'. To the left of the man is a box labeled 'SACTIONS' containing a list of items. Below that is a box labeled 'SOFTWARE PRODUCE' containing a list of items. At the bottom right is a box with a gear icon. The background is a stone wall.

# Why use Send an HTTP Request to SharePoint action over pre-built actions

# Send an HTTP Request to SharePoint

- Use endpoints not exposed via pre-built actions
  - ☐ Create SharePoint sites
  - ☐ Create Lists
  - ☐ Create Document Libraries
  - ☐ Create Site Columns
  - ☐ Update Choice Column Values
  - ☐ Move Files and Folders (CopyJobs)
  - ☐ Dynamic Information
    - ☐ SharePoint Site URLs
    - ☐ List Names
    - ☐ Document Library Names





# Demos

# Resources

[Get to know the SharePoint REST service](#)

[REST \(REpresentational State Transfer\)](#)

[Welcome to OData](#)

[Complete basic operations using SharePoint REST endpoints](#)

[Fields REST API reference](#)

# Contact Me

 [fausto@powerapps911.com](mailto:fausto@powerapps911.com) (Work)

 [fausto@vecaent.com](mailto:fausto@vecaent.com) (Personal)

 [@fcapellanjr](https://twitter.com/fcapellanjr)

 <https://www.linkedin.com/in/fausto-capellan-jr/>

 <https://faustocapellan.com>