Platform as a Service (PaaS) and Software as a Service (SaaS)

Roch Glitho, PhD

Full Professor

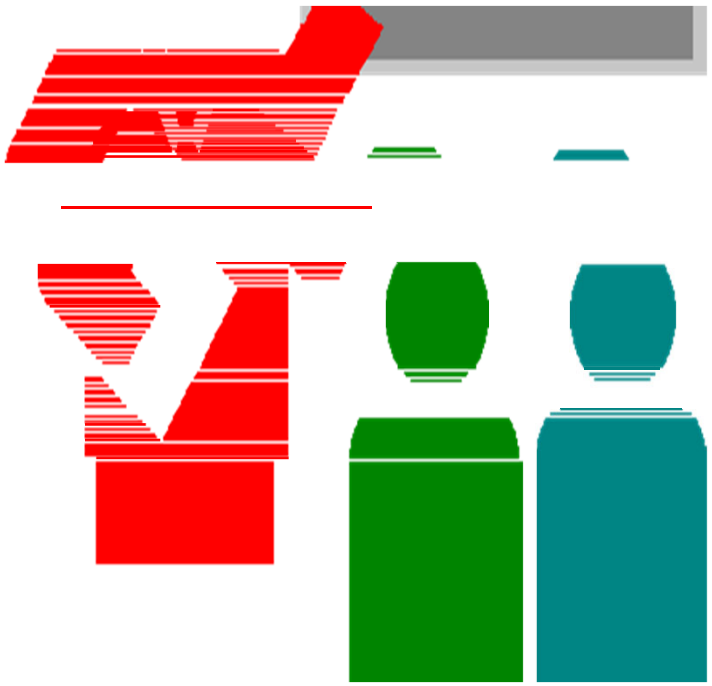
Ericsson / ENCQOR-5G Senior Industrial Research Chair

Cloud and Edge Computing for 5G and Beyond

My URL - http://users.encs.concordia.ca/~glitho

Concordia University
Engineering and Computer Science
Concordia Institute for Information Systems Engineering

PaaS and SaaS



- What is PaaS?
- What is an IaaS?
- Case studies
  - AWS Elastic Beanstalk Case Study
  - Software-as-a-Service
    A Case study on Zoom

References

1. L.M. Vaquero et al., A Break in the Clouds: Towards a Cloud Definition, ACM SIGCOM Computer Review, January 2009

2.Q. Zhang et al., Cloud Computing: State-of-the-Art and Research Challenges, Journal of Internet Services Applications (2010)

3.C. Vechiola et al., Aneka – Integration of Private and Public Cloud, Chapter 9, in “Cloud Computing: Principles and Paradigms”, (eds: R. Buyya et al.) Wiley 2011

Platform as a Service (PaaS)

What is a PaaS?

Platforms as a Service (PaaS):immersed part I (End-User perspective)

What is a PaaS?

Platforms as a Service (PaaS):immersed part I (Service provider perspective)

- Platforms used for the development and management of the applications offered as SaaS to end-users (and other applications)
  - Examples:
  - Google Cloud Engine
  - Microsoft Azur
  - Cloud Foundry

What is a PaaS?

Platforms as a Service (PaaS):immersed part I (Service provider perspective)

- Platforms used for the development and management of the applications offered as SaaS to end-users (and other applications)
  - Examples:
  - Google Apps Engine
  - Microsoft Azur
  - Cloud Foundry

What is a PaaS?

``Cloud systems can offer an additional abstraction level: instead of supplying a virtualized infrastructure, they can provide the software platform where systems run on. The sizing of the hardware resources demanded by the execution of the services is made in a transparent manner. This is denoted as Platform as a Service (PaaS)``

Reference 1.

What is a PaaS?

``The platform layer: Built on top of the infrastructure layer, the platform layer consists of operating systems and application frameworks. The purpose of the platform layer is to minimize the burden of deploying applications directly into VM containers. For example, Google App Engine operates at the platform layer to provide API support for implementing storage, database and business logic of typical web applications``

Reference 2.

What is a PaaS?

``They provide enterprises with a platform for creating, deploying and managing distributed applications on top of existing cloud infrastructures. They are in charge of monitoring and managing the infrastructure and acquiring new nodes and they rely on virtualization technologies in order to scale applications on demand”

Reference 3.

What is a PaaS?

PaaS handle application / service life cycle

- 4 phases in the early service life cycle models

1. Phase 1: Development –
   - Includes design, testing …
2. Phase 2: Deployment
3. Phase 3: Usage
   - Includes activation, execution ..
4. Phase 4: Removal

What is a PaaS?

On application / service life cycle

The 4 phases are sometimes collapsed in two phases:

1. Phase 1: Development –
   - Includes design, testing …
2. Phase 2: Management (i.e. everything that is not development)
   - Deployment
   - Usage
   - Removal

What is a PaaS?

A PaaS might re-use existing application / service life cycle frameworks / tools for some of the phases:

- Microsoft Azure re-uses .NET for development phase
- Cloud Foundry does not come with frameworks for development phase (The user can choose)

What is a PaaS?

A PaaS might be bound to a given IaaS or allow the user to select within a pre-defined set:

- Google Apps Engine comes with Google IaaS
  - The infrastructure can be bypassed to some extend
- Cloud Foundry the user to select within a pre-defined set (e.g. Openstack, Amazon WS)

What is a PaaS?

A PaaS might allow or not allow auto-scaling:

- Horizontal and/or
- Vertical

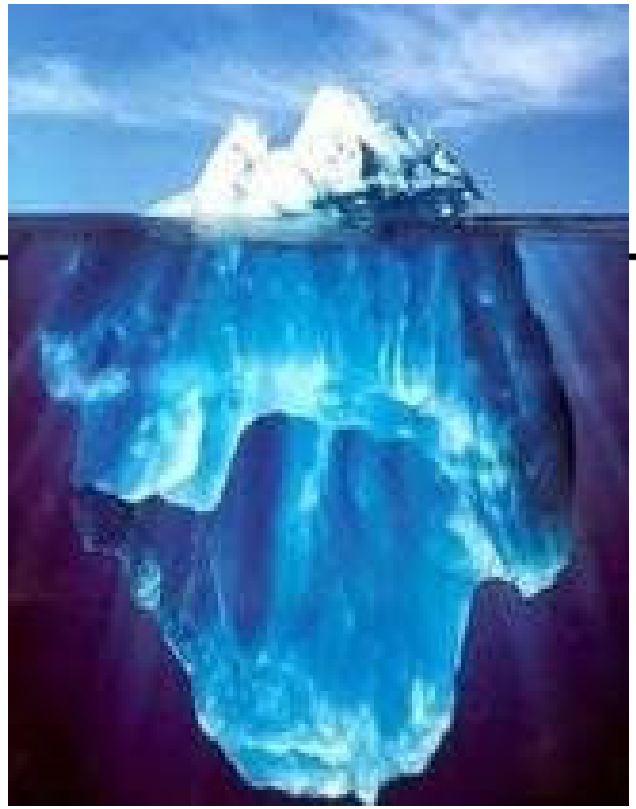
Software as a Service (SaaS)

SaaS

- What is SaaS

Software as a Service

Software as Services (SaaS):the tip of the iceberg (End-user perspective)



Software as a Service

Software as Services (SaaS):the tip of the iceberg (End-user perspective)

Applications offered by service providersand residing in the cloud

- Pay per use basis (or sometimes “free”)

Software as a Service

Software as Services (SaaS):the tip of the iceberg (End-user perspective)

Applications offered by service providersand residing in the cloud

- Offer:
  - Non programmatic interface and / or non programmatic interface (most SaaS offer both)
  - Programmatic interfaces are usually Web services based
    - RESTful Web services

Software as a Service

Software as Services (SaaS):the tip of the iceberg (End-user perspective) – Some examples:

- Zoom
- Dropbox
- Google docs
- And so on ..

Case Studies

Platform-as-a-Service:
AWS Elastic Beanstalk Case Study

Presented by: Behshid Shayesteh

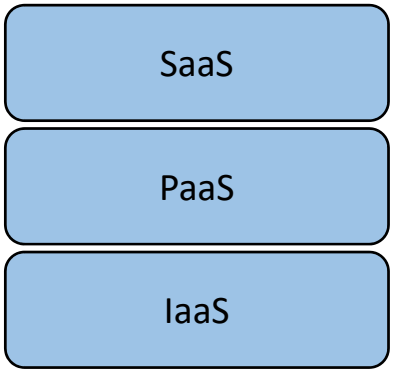
Outline

- Platform-as-a-Service
- AWS Elastic Beanstalk
  - Overview
  - Key Features
  - How it works
  - Getting started
  - Deployment example

Platform-as-a-Service (PaaS)

- Platform used for the development and management of the applications offered as SaaS to end-users (and other applications)

- Examples:
  - Google App Engine
  - Microsoft Azure App Service
  - Heroku
  - **Amazon AWS Elastic Beanstalk**



AWS Elastic Beanstalk Overview

- PaaS offered by AWS
  - Introduced in 2011 to simplify application deployment on AWS
  - A service for deploying and scaling web applications and services

- Automatically handles deployment details
  - Provisioning
  - Load balancing
  - Auto-scaling
  - Application health monitoring

AWS Elastic Beanstalk Overview

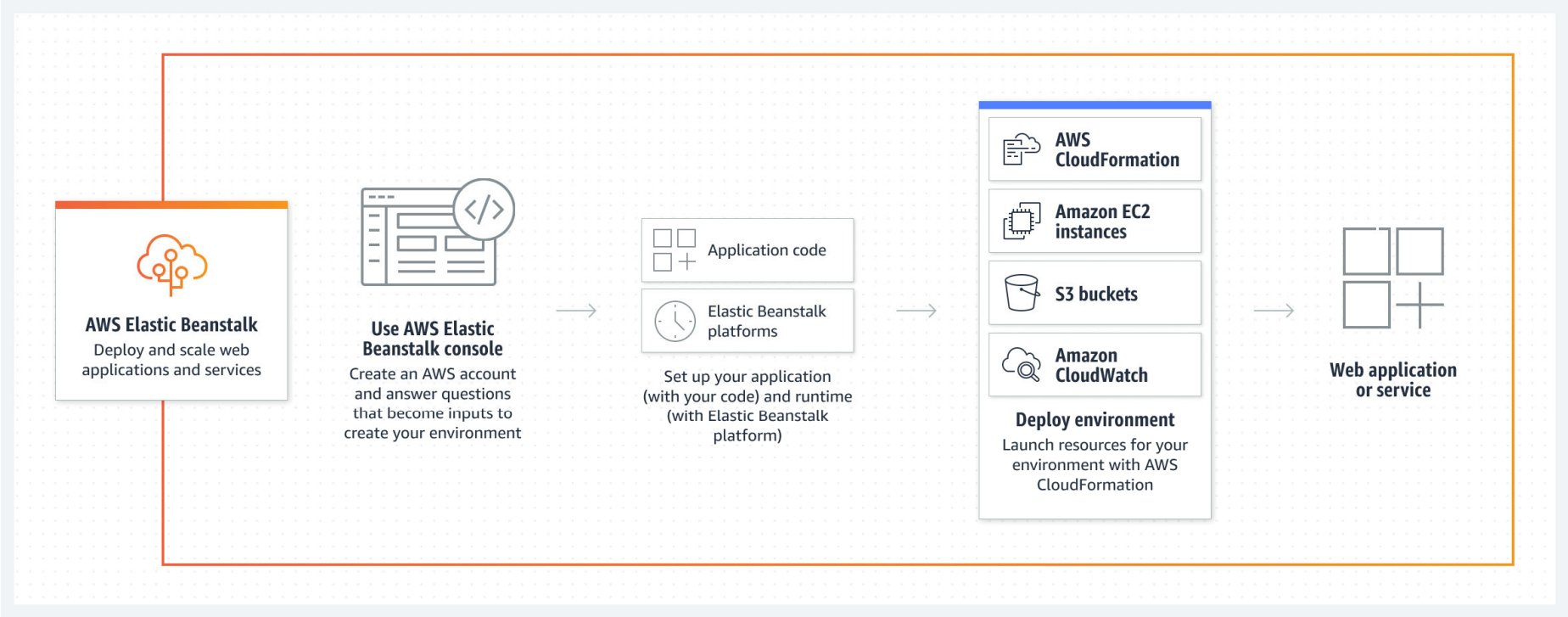


Figure from https://aws.amazon.com/elasticbeanstalk/

AWS Elastic Beanstalk Features

- Some of the key features
  - Easy to use
  - Wide selection of development options
    - Java, .NET, Node.js, PHP, Ruby, Python, Go, and Docker
  - Integrated with core AWS services
    - Amazon Elastic Compute Cloud (EC2), Amazon Elastic Container Service (ECS), AWS Auto Scaling, and Elastic Load Balancing (ELB)

AWS Elastic Beanstalk Features (contd.)

- Some of the key features
  - Variety of application deployment options
    - AWS management consule, Elastic Beanstalk CommandLine Interface (CLI), Visual Studio, Eclipse
  - Multiple deployment policies
    - All at once, rolling, rolling with an additional batch, immutable, and blue/green
  - No additional charge for AWS Elastic Beanstalk
    - Pay for the resources

How AWS Elastic Beanstalk Works

- Developers upload their code to AWS Elastic Beanstalk
  - Using AWS console, CLI, IDE
- AWS Elastic Beanstalk sets the environment and deploys the application into AWS infrastructure
  - Developers can configure the resources if desired
- Management and update
  - Developers can monitor their application and update

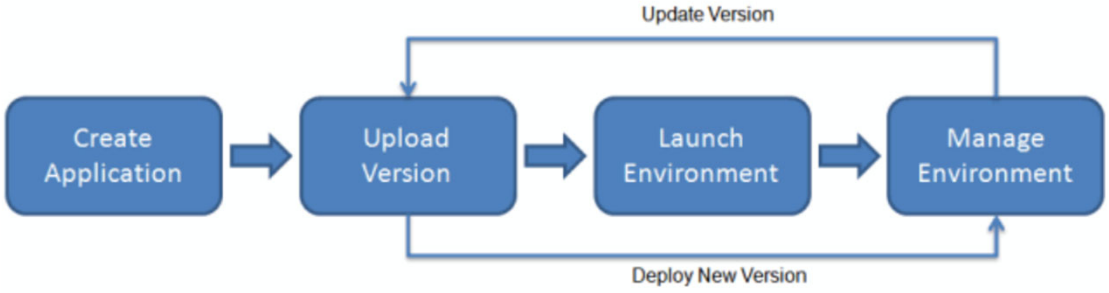


Figure from https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/Welcome.html

Getting Started with AWS Elastic Beanstalk

- Develop your application locally
  - Example: a Flask web application developed in python
- Create an AWS account
- Choose AWS Elastic Beanstalk service and web server environment
  - Upload your source code bundle

  Upload a list of requirements, e.g., Flask
- Use AWS Elastic Beanstalk to provision resources and deploy application automatically
- Access the URL provided by AWS Elastic Beanstalk for your application

Hybrid Deployment Example

- Scenario 1: Use AWS CloudFormation to deploy an Elastic Beanstalk application along with an AWS service integration
  - DynamoDB
  - Amazon RDS
  - Amazon S3.

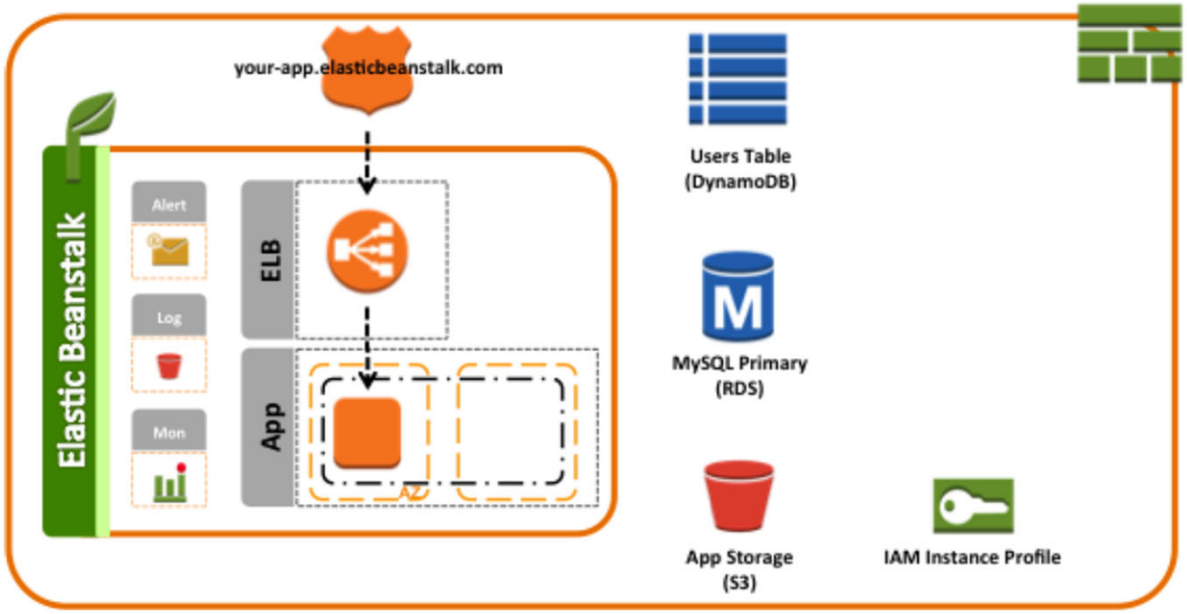


Figure 3: Reference Architecture for Scenario 1

Figure from Overview of Deployment Options on AWS, by Peter Dalbhanjam, March 2015

Software-as-a-Service
A Case study on Zoom

Mahsa Raeiszadeh

Outline

- Introduction On SaaS
- Advantages of SaaS
- Introduction to Zoom
- Example Application
- Scenarios Overview

Introduction

- **Software as a Service (SaaS)** is a cloud computing model that delivers software applications over the internet.
- Characteristics:
  - On-demand
  - Subscription-based
  - Centrally hosted
  - Scalable
  - Accessible via a web browser
- Enables organizations to access powerful software without the need for complex installations and maintenance.

Why SaaS?

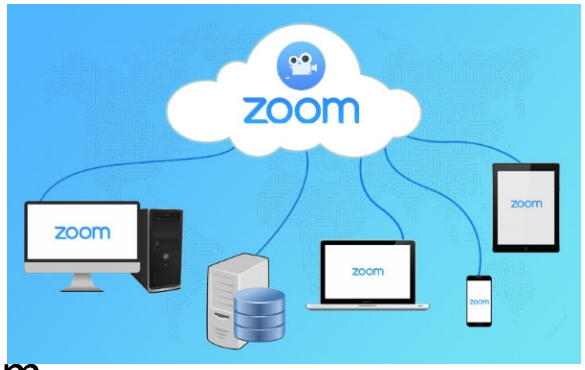
- **Advantages of SaaS for Cloud Application Development**
  - **Cost-Efficiency:** Eliminates the need for upfront hardware and software investments.
  - **Scalability:** Easily scale resources up or down based on business needs.
  - **Accessibility:** Anytime, anywhere access to applications through the internet.
  - **Automatic Updates:** Providers handle maintenance and updates, ensuring the latest features and security patches.

  - Example:
    - Zoom
    - Telegram
    - Gmail

Introduction to Zoom

- **What is Zoom API?**
  - A set of RESTful web service APIs that allow developers to integrate Zoom functionality into their applications.
- **Use Cases:**
  - Embedding video conferencing capabilities.
  - Managing users, meetings, and webinars programmatically.

Key Features of Zoom API



- **Meeting Management:**
  - Schedule, update, and manage meetings seamlessly.
- **User Management:**
  - Add, delete, and manage users within the Zoom ecosystem.
- **Webinar Management:**
  - Host and manage webinars, control participant access.
- **Reporting:**
  - Retrieve detailed reports on meetings, users, and webinars.

Zoom API Integration for HR Management System

Example Application:

- **HR Management System:** Showcase the versatility of Zoom API in enhancing various HR processes, providing a seamless and integrated experience for HR administrators and employees.

Key Scenarios:

- **Employee Onboarding:** Schedule recurring onboarding meetings.
- **Team Collaboration:** List upcoming collaboration meetings.
- **Performance Reviews:** Update meeting details for performance reviews.
- **Recruitment Interviews:** Cancel scheduled job interviews.
- **Virtual Team Building:** Create impromptu team building sessions.
- **Attendance Tracking:** Retrieve meeting participants for attendance tracking

Simulating HR Management System Actions with Flask

- We begin our exploration with an interactive Flask application.
- The application simulates an HR Management System with different buttons representing key HTTP methods.

- **Application Features:**
  - **Get Participants Button:**
    - Initiates a request to retrieve a list of meeting participants.
  - **Delete Meeting Button:**
    - Triggers a request to delete a scheduled meeting.
  - **Set Meeting Button:**
    - Sends a request to schedule a new meeting.

Scenario 1: Employee Onboarding - Schedule Recurring Meetings (POST Request)

- **Objective:** Streamline the onboarding process for new employees by integrating virtual orientations and training sessions using Zoom API.
- **Implementation:**
  - HR administrators use the HR Management System to initiate the onboarding process for new employees.
  - The HR system sends a POST request to Zoom API to schedule virtual orientation meetings and training sessions.

```
{
  "action": "Employee Onboarding - Schedule Recurring Meetings",
  "method": "POST",
  "endpoint": "/v2/users/{newEmployeeId}/meetings",
  "data": {
    "topic": "Employee Onboarding Session",
    "type": 3, // Recurring meeting type
    "start_time": "2023-03-01T09:00:00Z",
    "duration": 60,
    "timezone": "America/New_York",
    "recurrence": {
      "type": 2, // Weekly recurrence
      "weekly_days": "Monday"
    },
    "agenda": "Introduction to company culture and policies"
```

Scenario 2: Team Collaboration – List Upcoming Meetings (GET Request)

- **Objective:** Retrieve a list of upcoming team collaboration meetings.

```
{
  "action": "Team Collaboration - List Upcoming Meetings",
  "method": "GET",
  "endpoint": "/v2/users/{teamLeadId}/meetings"
}
```

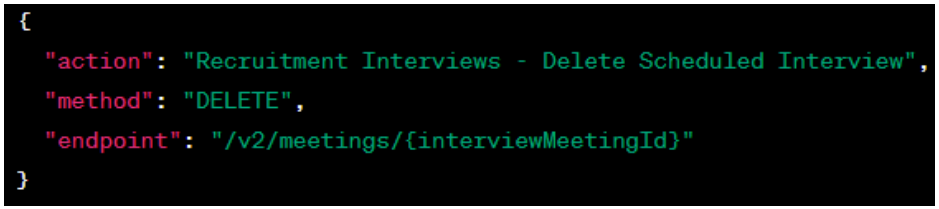
Scenario 3: Performance Reviews - Update Meeting Details (PATCH Request)

- **Objective:** Modify the details of a scheduled performance review meeting.

```
{
  "action": "Performance Reviews - Update Meeting Details",
  "method": "PATCH",
  "endpoint": "/v2/meetings/{performanceReviewMeetingId}",
  "data": {
    "topic": "Updated Performance Review",
    "agenda": "Discuss performance, set goals, and provide feedback"
  }
}
```

Scenario 4: Recruitment Interviews - Delete Scheduled Interview (DELETE Request)

- **Objective:** Cancel a scheduled virtual job interview.



Scenario 5: Virtual Team Building - Create Instant Meeting (POST Request)

- **Objective:** Facilitate impromptu virtual team building sessions.

```
{
  "action": "Virtual Team Building - Create Instant Meeting",
  "method": "POST",
  "endpoint": "/v2/users/{teamLeadId}/meetings",
  "data": {
    "topic": "Virtual Team Building",
    "type": 1, // Instant meeting type
    "agenda": "Fun activities and team bonding"
  }
}
```

Scenario 6: Attendance Tracking - Get Meeting Participants (GET Request)

- **Objective:** Retrieve a list of participants for a specific meeting to track attendance.


```
{
  "action": "Attendance Tracking - Get Meeting Participants",
  "method": "GET",
  "endpoint": "/v2/meetings/{meetingId}/participants"
}
```

•A

The End