Exploring Actor Model Support

Ivan Gavryliuk
SOFTWARE ARCHITECT

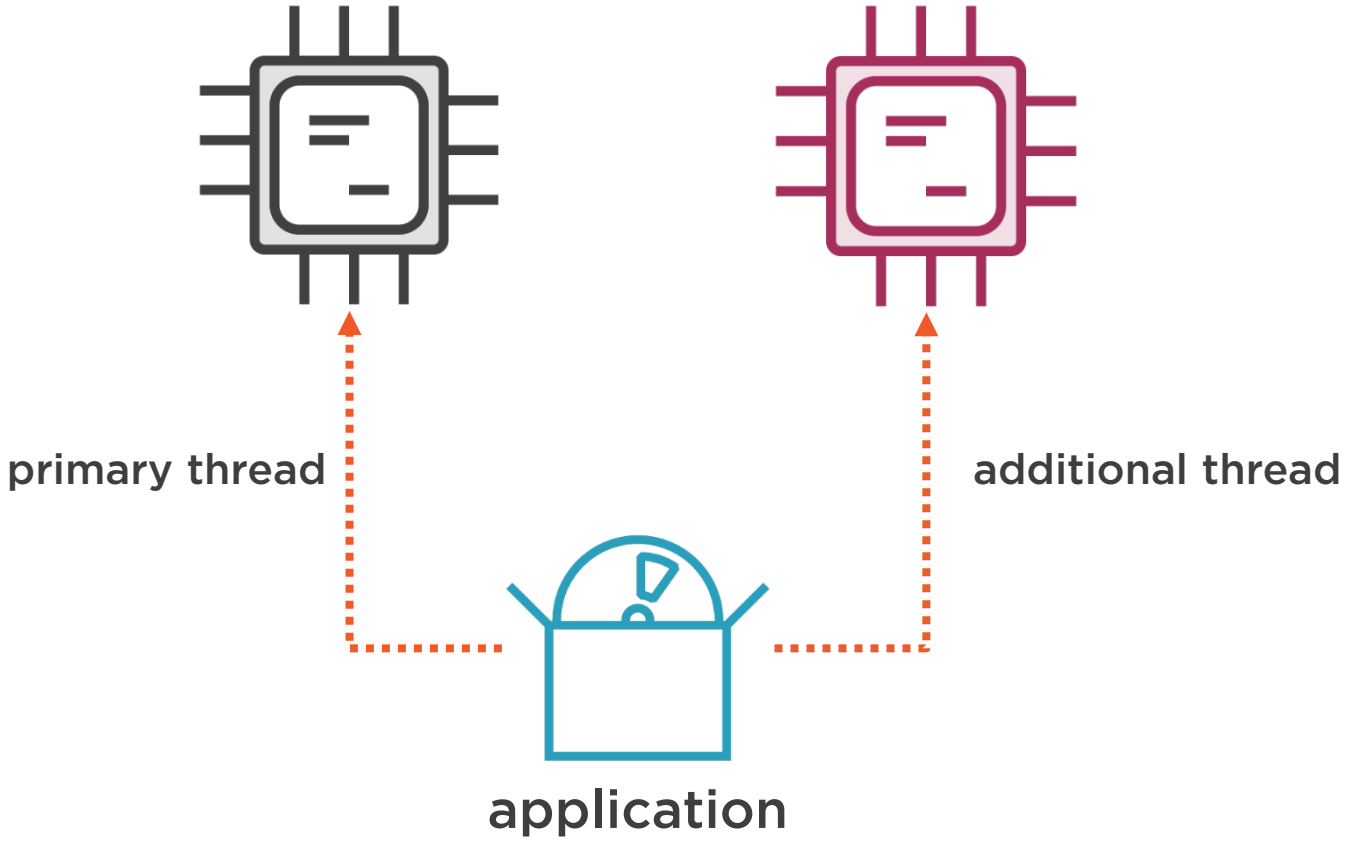
@aloneguid http://isolineltd.com

Overview

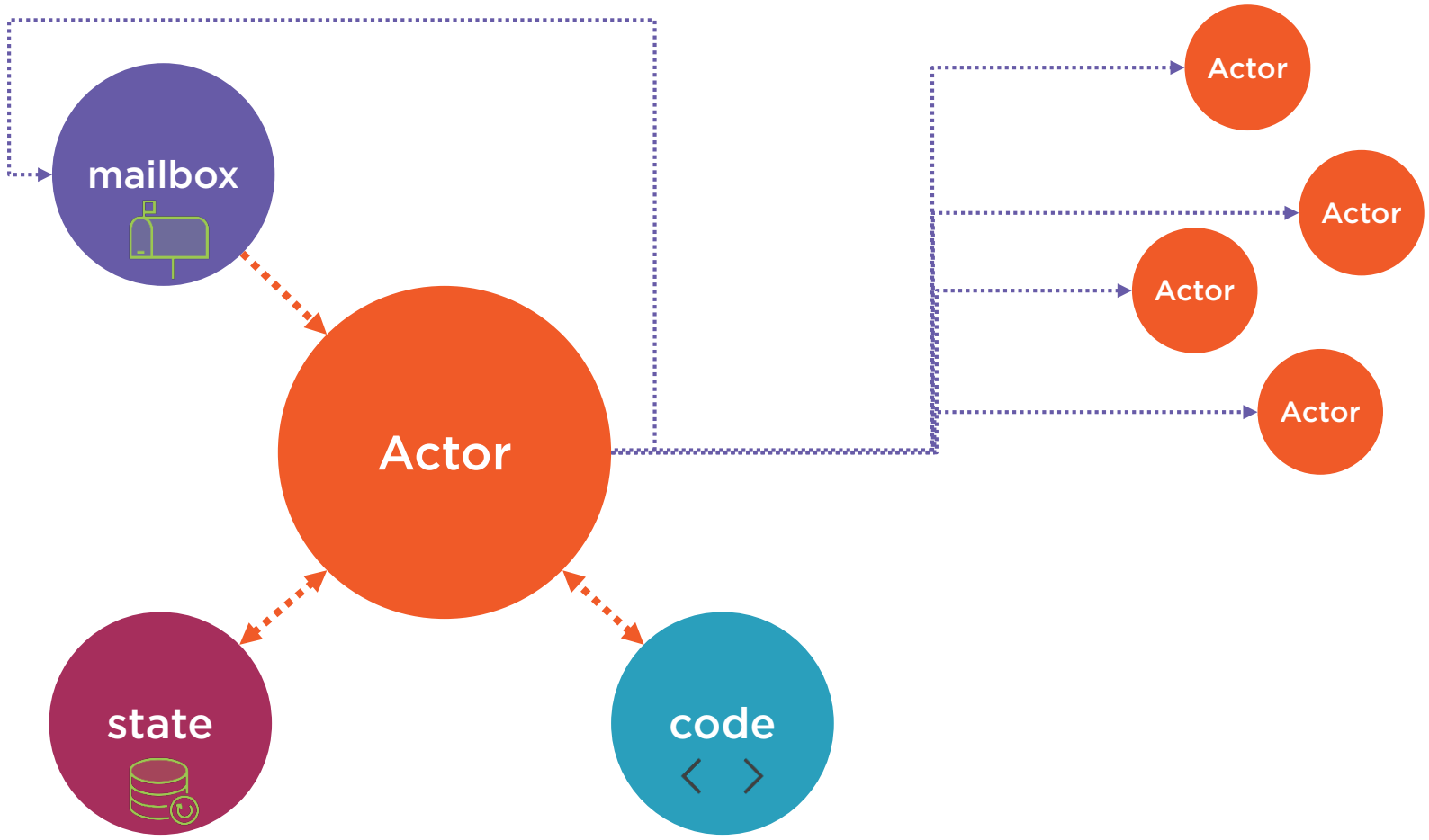
- Why Actor model
- Challenges of parallel programming
- When to use Actor model
- Service Fabric Actors
- Building User Actor

Actor Model Theory

Threading

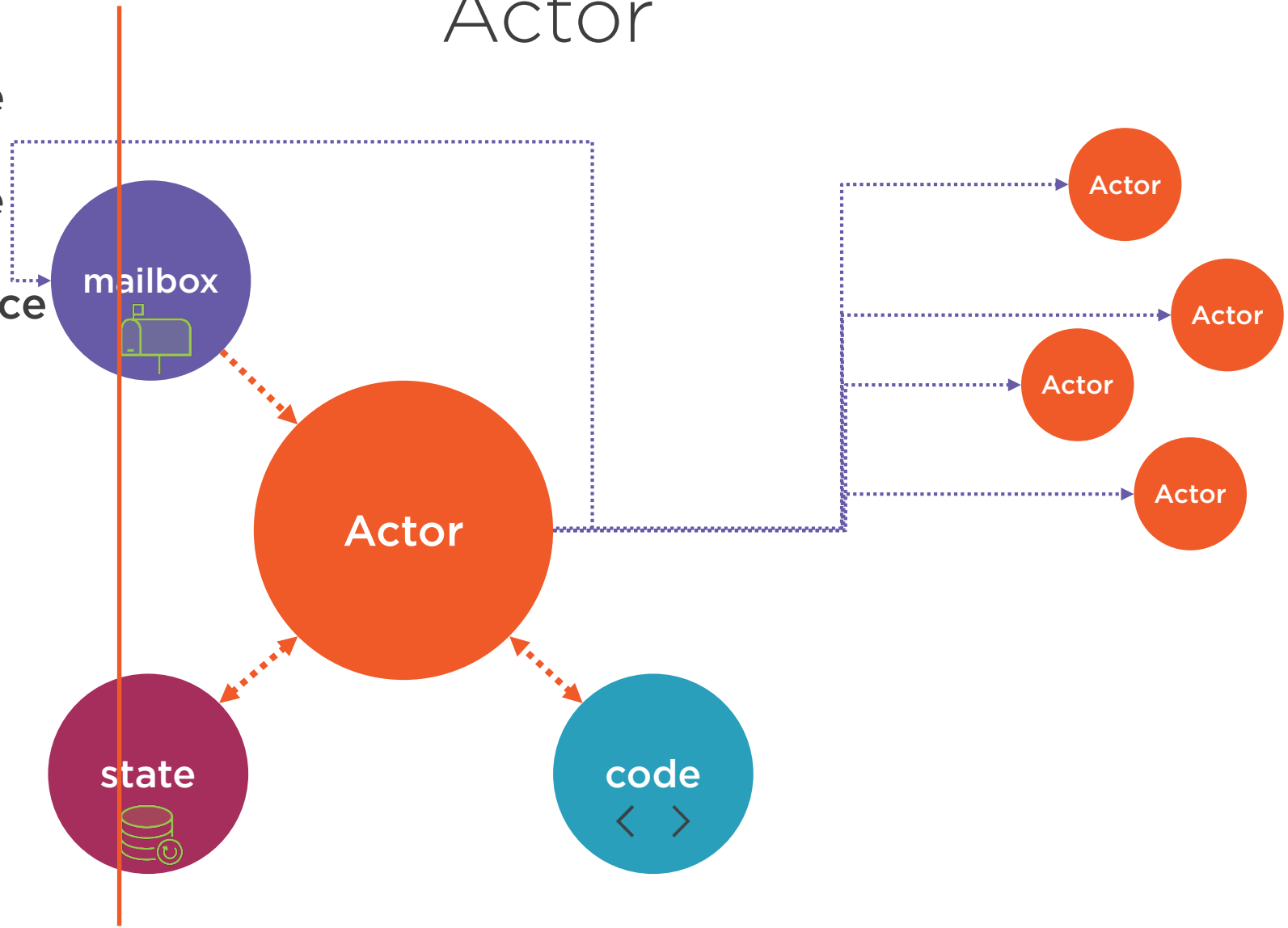


Actor

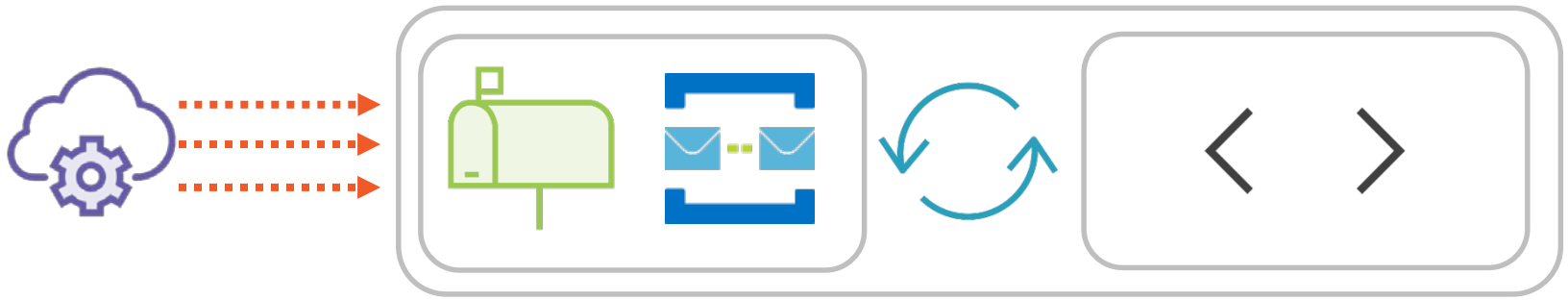


Actor

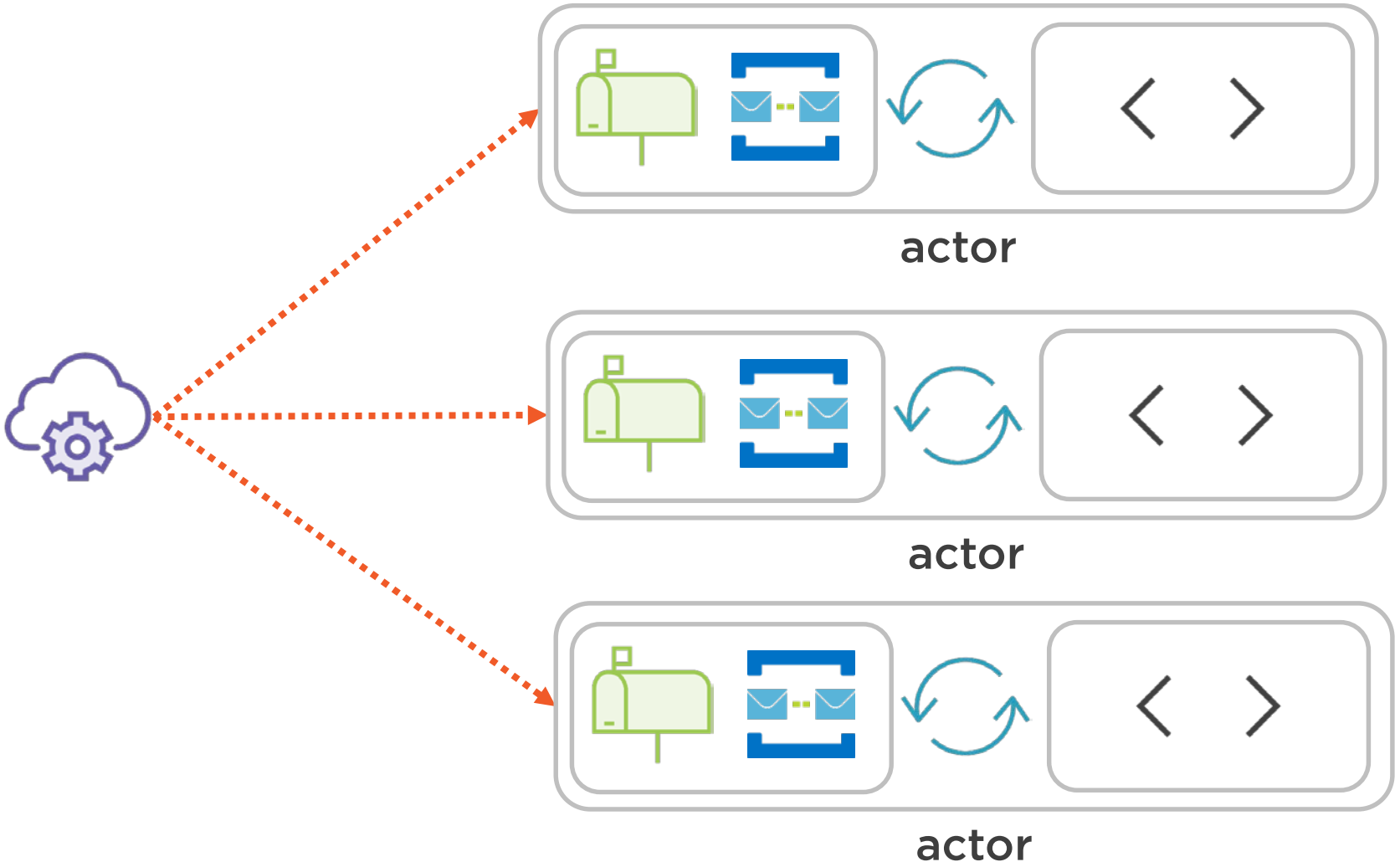
- **Code:** Reliable Services
- **State:** Reliable State
- **Mailbox:** Service Remoting



Multithreading



Multithreading

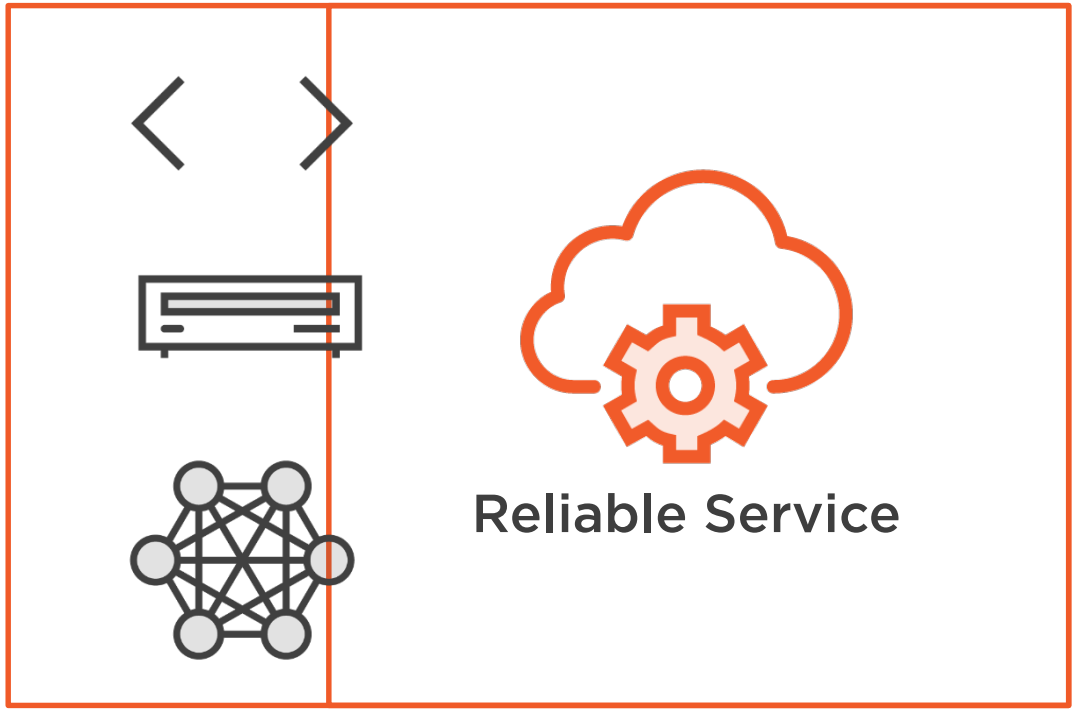


Multiple actors can run parallelly, but one actor processes messages sequentially.

Service Fabric Actors

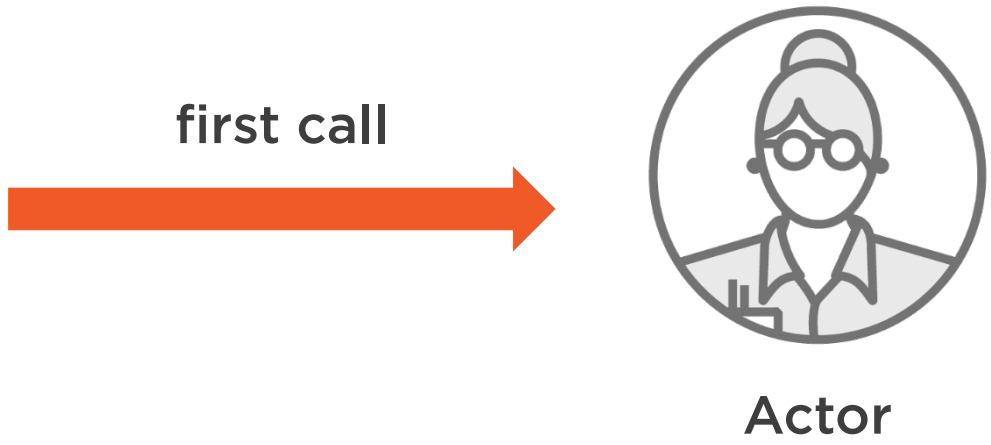
Framework

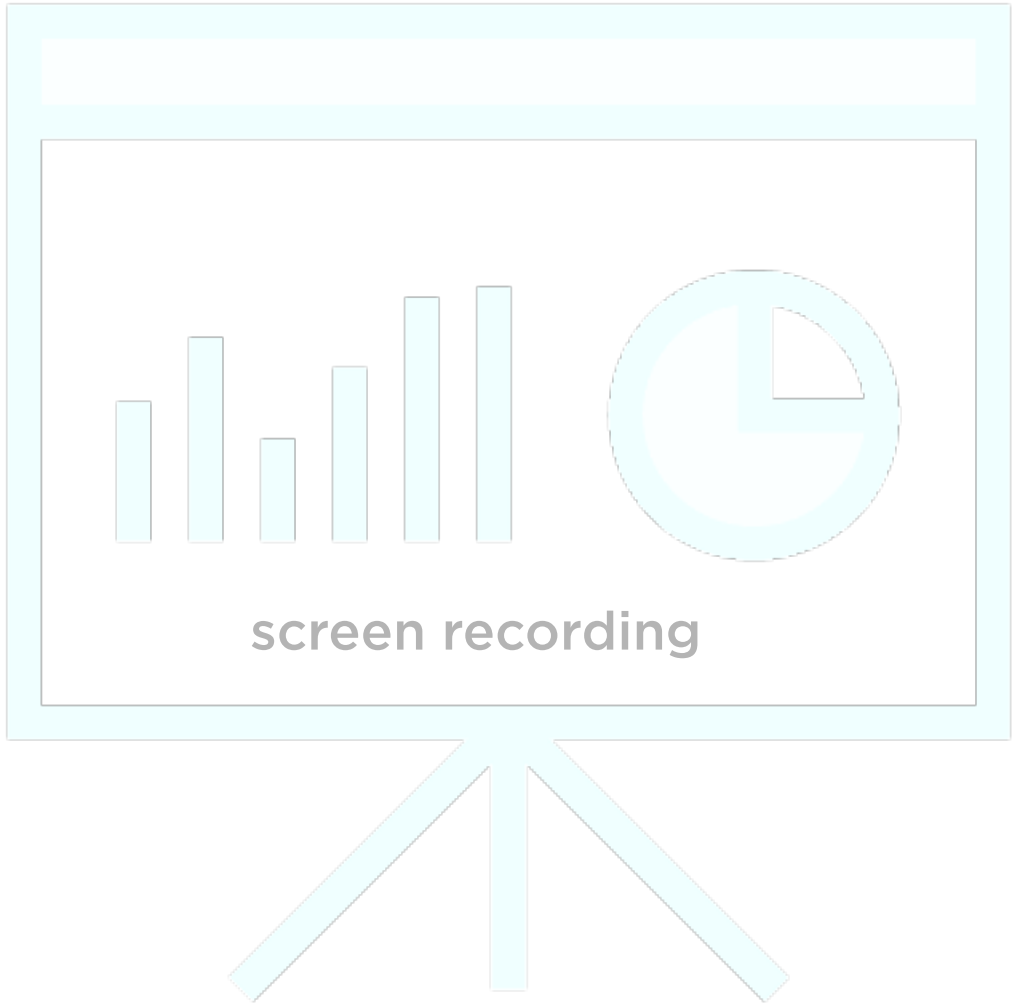
Actor



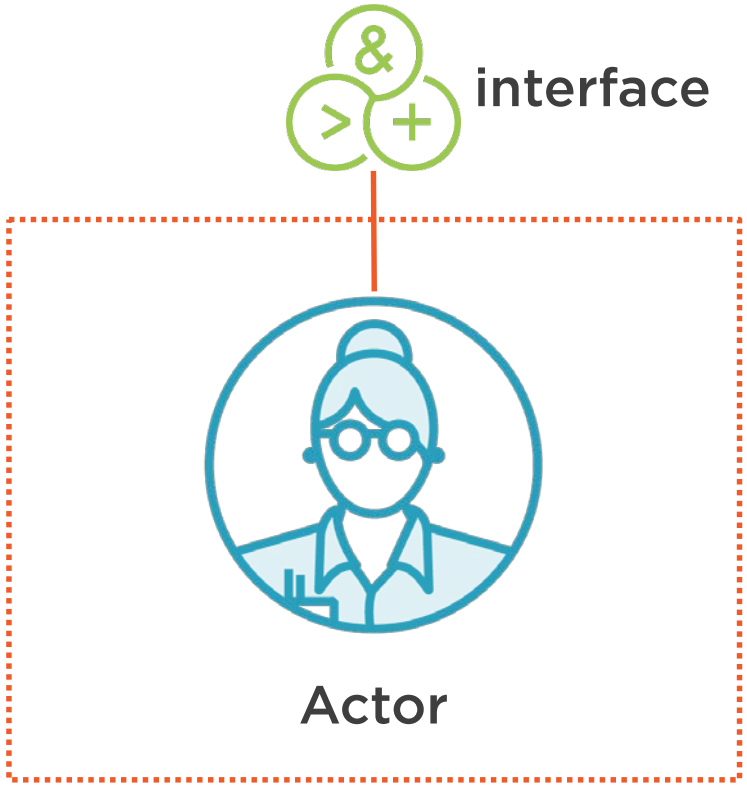
Lifetime not tied to in-memory representation

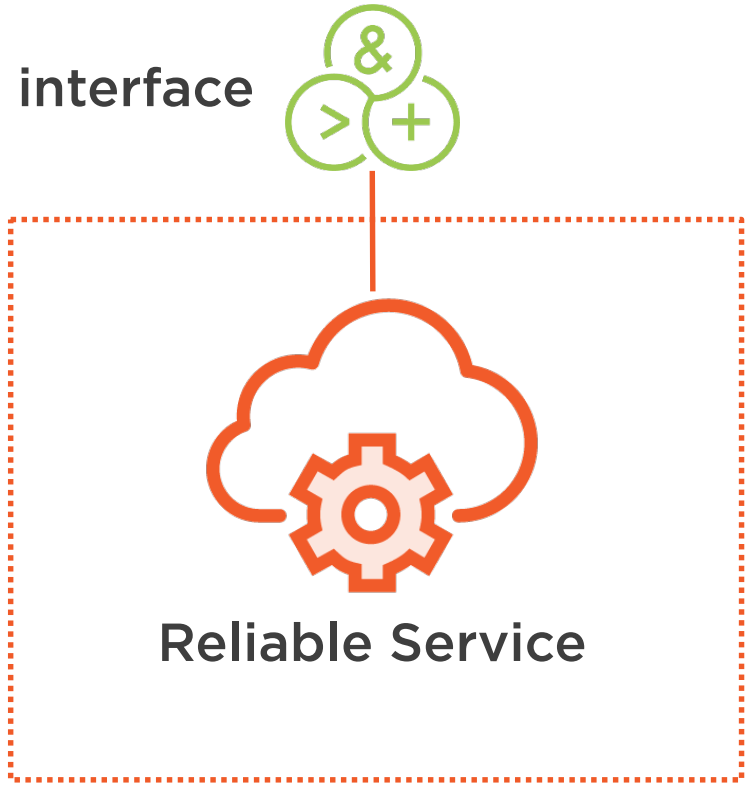
No need to explicitly create or destroy

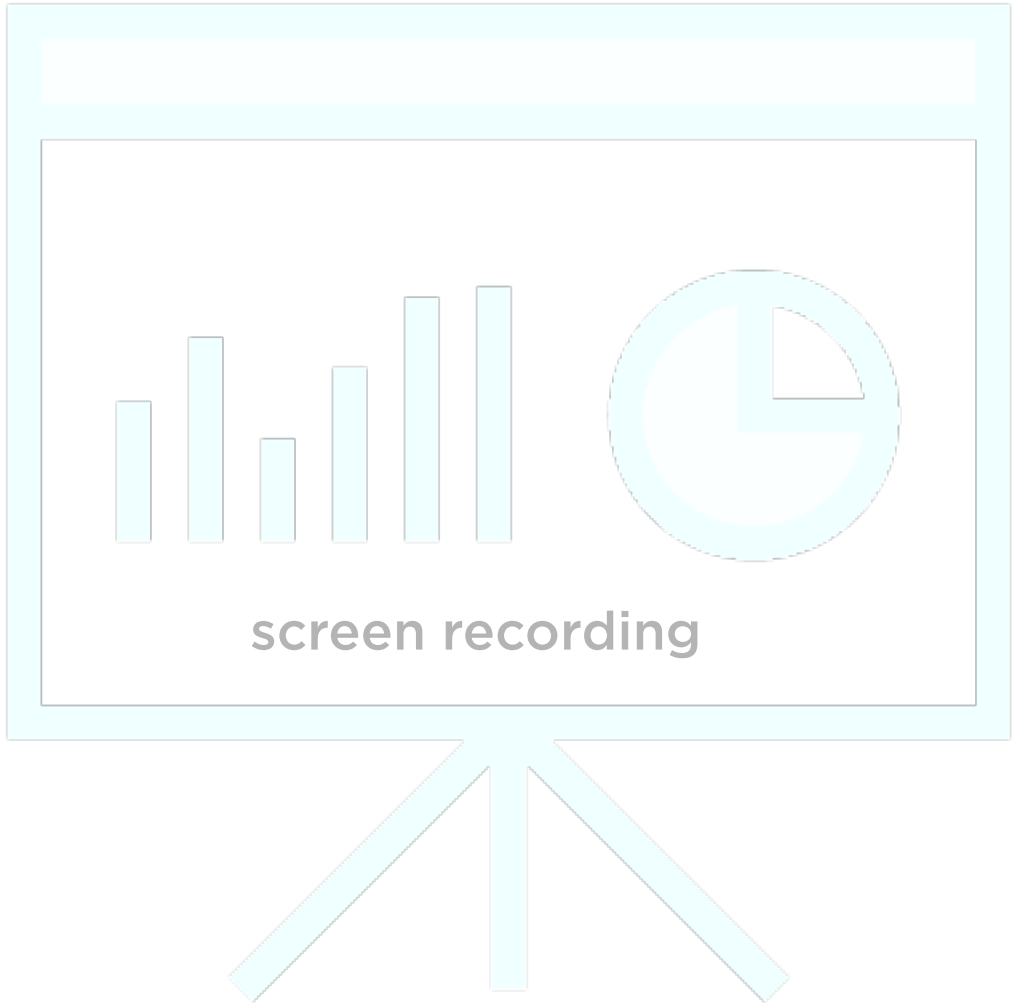



screen recording

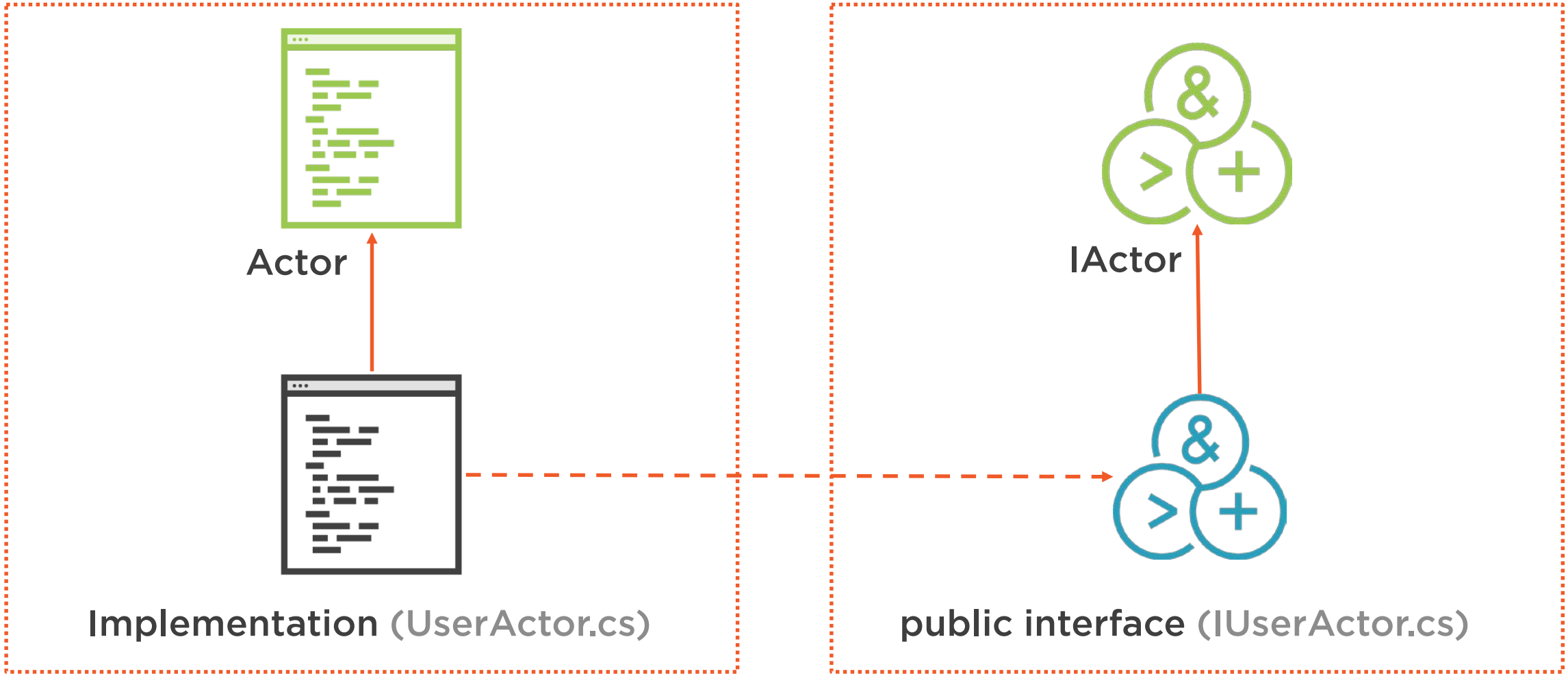
Good Practices

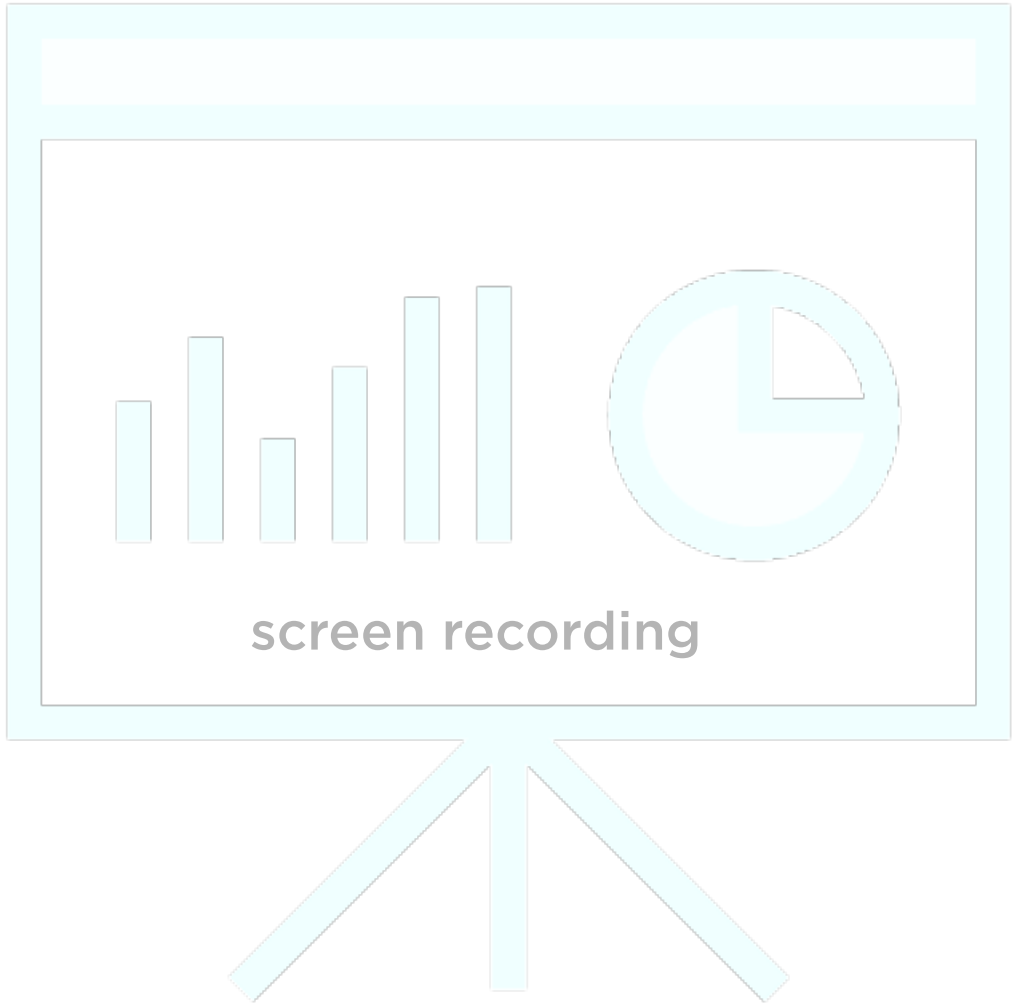





screen recording

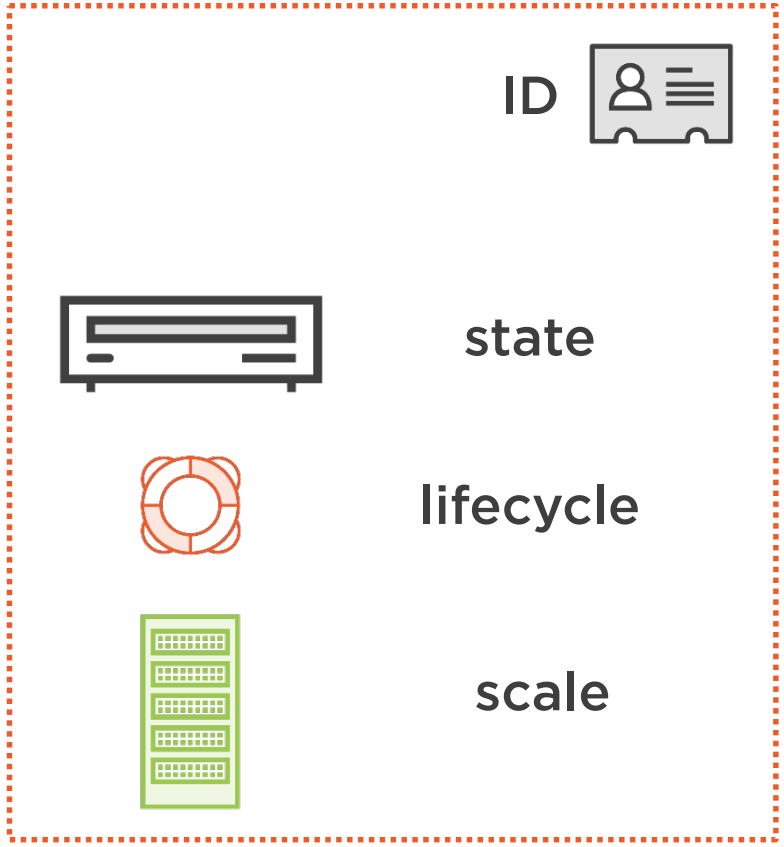
Project Structure




screen recording

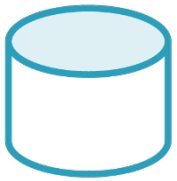
Actor Types

Actor Type

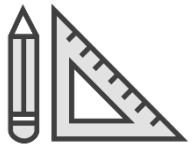


Actor ID

Logging detailed information

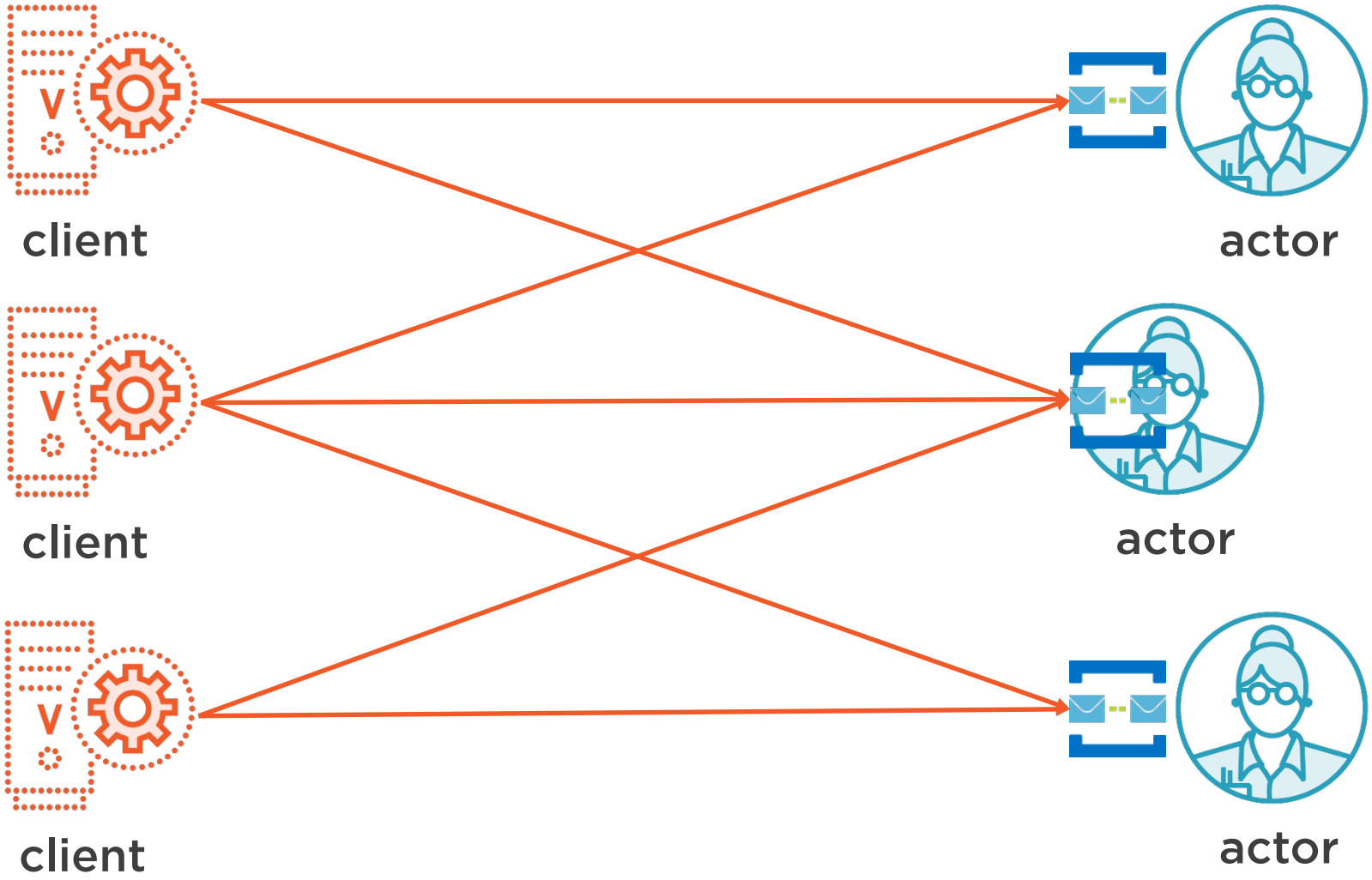


Communication with external data providers



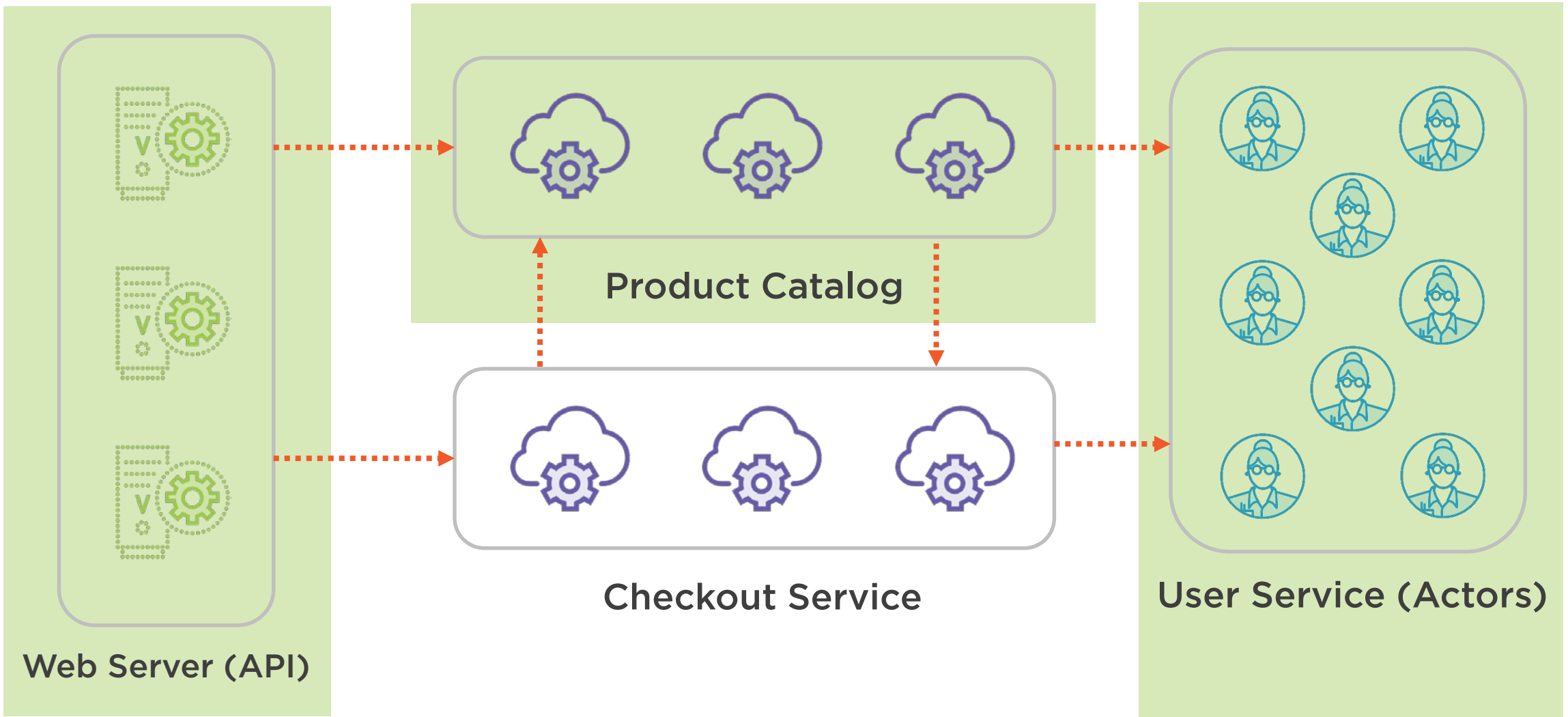
Any other context

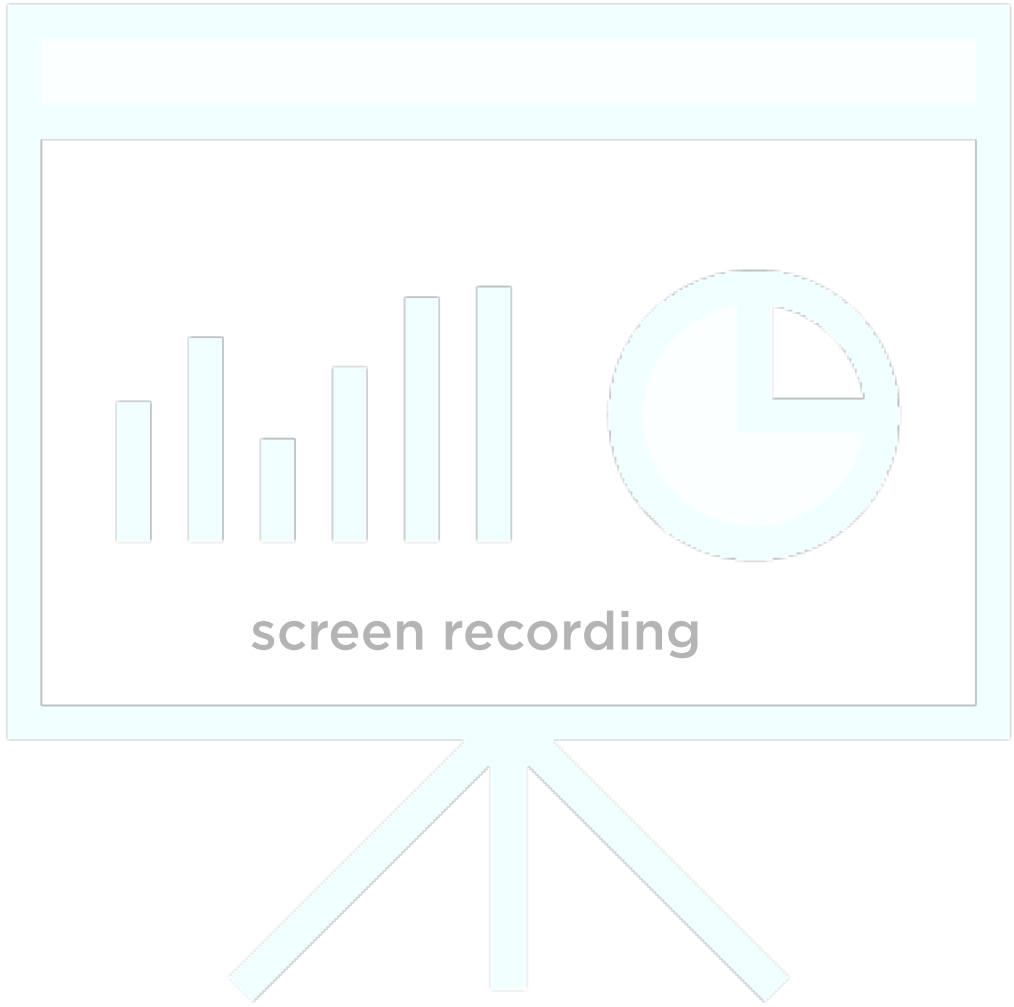
Concurrency



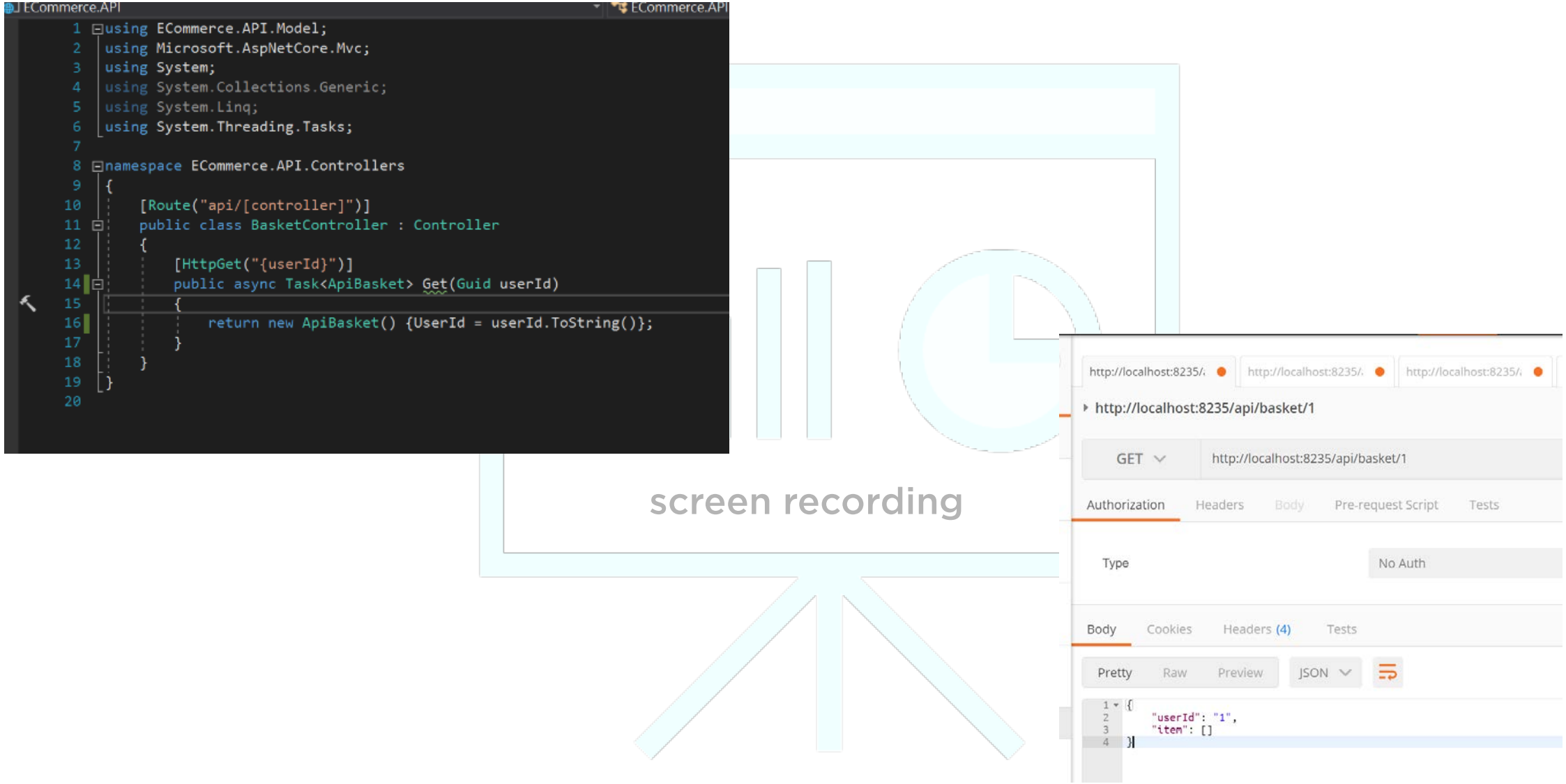
Building User Service

Application




screen recording

Calling to Actors


ECommerce.API
ECommerce.API
using ECommerce.API.Model;
using Microsoft.AspNetCore.Mvc;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
namespace ECommerce.API.Controllers
{
[Route("api/[controller]")]
public class BasketController : Controller
{
[HttpGet("{userId}")]
public async Task<ApiBasket> Get(Guid userId)
{
return new ApiBasket() {UserId = userId.ToString()};
}
}
}
screen recording
http://localhost:8235/
http://localhost:8235/
http://localhost:8235/
http://localhost:8235/api/basket/1
GET
http://localhost:8235/api/basket/1
Authorization
Headers
Body
Pre-request Script
Tests
Type
No Auth
Body
Cookies
Headers (4)
Tests
Pretty
Raw
Preview
JSON
{
"userId": "1",
"item": []
}

Summary

- Actor model theory
- Service Fabric Actors
- Extreme ease of use
- Sample Actor
- Partitioning and scaling
- Actor communication