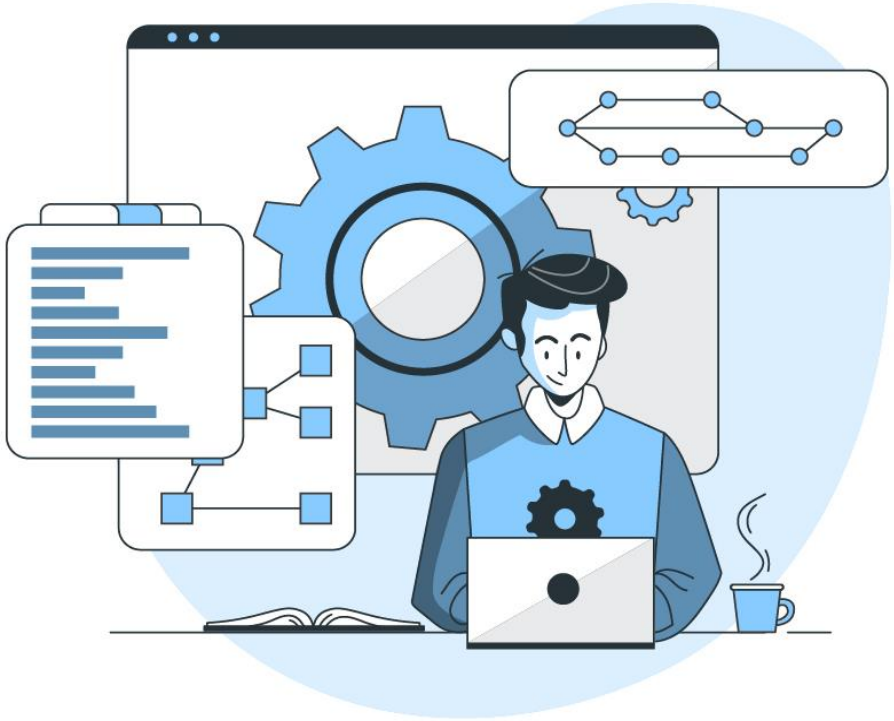


Ayudantía 08: Workers

IIC2173 – 2025.2

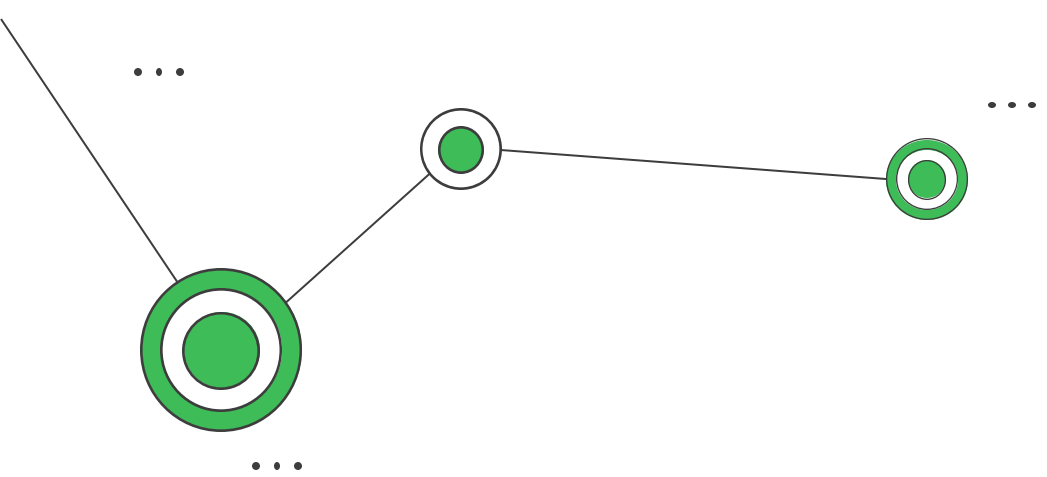
01

Workers

Empecemos con un minimarek



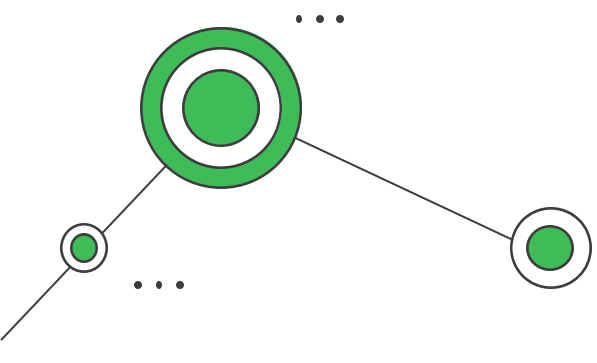

bientôt . merci
10%



¿Qué hace que pueda llegar a esto?

Inicialmente se tiene una fila
Para escalar la atención
Se agregan más filas

Workers



Modelos esenciales de software para este contexto



Monolito



Servicios
SoA (Arquitectura orientada a Servicios)



Microservicios

Modelo monolítico de software

Ganamos

- Una gran unidad de software, con todos los servicios que requerimos
- Rápido de levantar, inicialmente fácil de modificar y testear
- Ideal para PoC [Proof Of Concept] o aplicaciones de baja demanda, salvo aplicaciones específicas

Perdemos

- Baja modularidad
- Mala escalabilidad
- Código atestado
- Lock-in en framework
- Problemas de flexibilidad
- Cae el monolito, cae toda la app

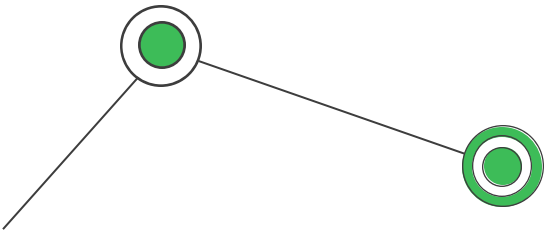
Terminamos pagando a futuro en varios frentes, y en RNFs

Ejemplo Real

Netflix

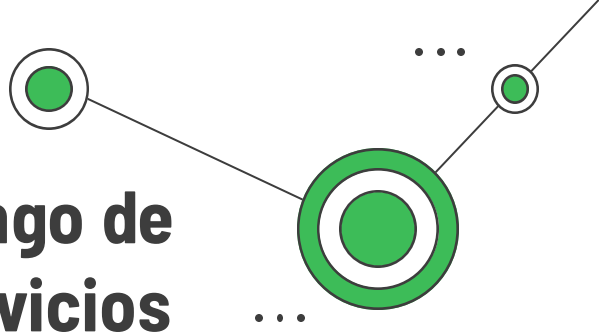
- En 2009, Netflix tuvo problemas de crecimiento. Su infraestructura no podía seguir el ritmo de la demanda de sus servicios de streaming de video, que crecía a toda velocidad. En esta situación, la empresa decidió migrar su infraestructura de TI de sus centros de datos privados a una nube pública y reemplazar la arquitectura monolítica por otra de microservicios. El único problema era que el término "microservicios" no existía en aquella época, y que la estructura tampoco era muy conocida.
- Netflix se convirtió en una de las primeras empresas destacadas en migrar de un monolito a una arquitectura de microservicios basada en la nube.
- Comparación entre la arquitectura monolítica y la arquitectura de microservicios

¿Cómo voy escalando? A grandes rasgos



Vengo de monolito

- Deshilachar el monolito
  - Ir extrayendo servicio tras servicio
    - Capacidades
    - Dominios
    - Equipos
- En microservicios lo piensan bien
  - https://microservices.io/patterns/decomposition/decompose-by-business-capability.html



Vengo de servicios

- Añadir más servicios en el esquema
  - Vía RPC (llamadas remotas), buses de eventos
  - Uso de APIs para definir

Workers

- Tareas “Subcontratadas”: Jobs
  - Preferentemente atómicas
  - Minimalista
- Funcionan en base a trabajos
  - Siguen instrucciones
  - Van de forma paralela a un proceso principal
- Devuelven resultados
  - Ticket / job_id
  - Callback
  - Datastore/mart/lake/whatever



¿Por qué usar Workers?

- Descentralizar tareas
  - Aligerar la carga del servicio principal
  - Desbloqueas capacidad de procesamiento
    - Node/CPU
- Puedes ordenar tareas
- Fácil de pasar a serverless
- Menos complejo que un microservicio
  - Podrías hacer un upgrade a futuro...



Cons de los Workers

- Requiere esfuerzos de coordinación
- Manejo de fragmentación
- Costos up-front extras
  - Diseñar entorno
  - Levantar extras
  - Configuración
- No siempre puedes usar cloud
  - Aún existen las cargas on-premises



Estilos involucrados

RPC (Remote Procedure Call), invocaciones

- Workers independientes como servicio
- Serverless

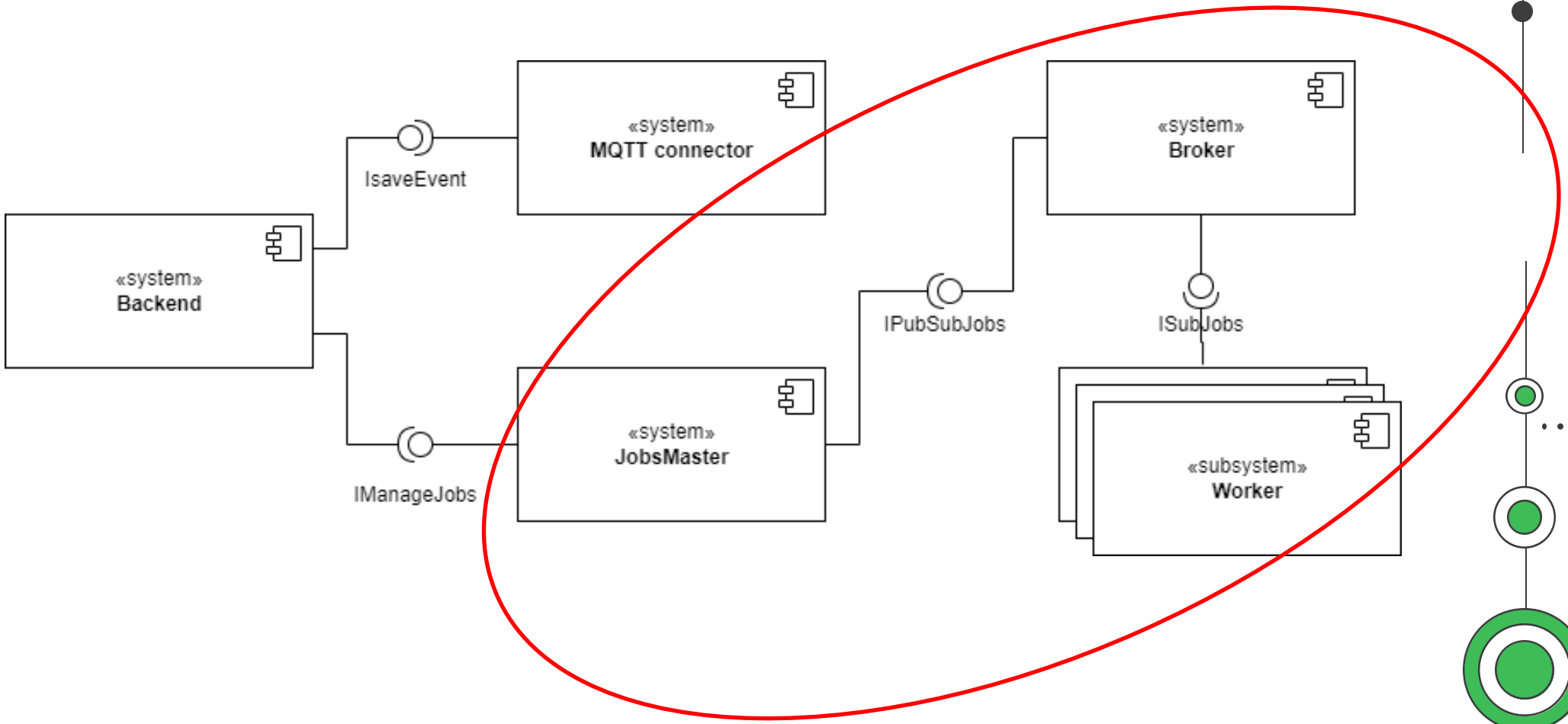
PubSub

- Notificación mediante broker
- Cubre escalamiento y ciertos tipos de trabajos
  - Encolado de trabajos
  - Distribución de trabajos

Flujo de datos

- Por lotes: el trabajo se puede segmentar
- Pipe and filter: pueden encadenarse workers

En el proyecto



Herramientas: Backend

- Código: Lo que deseen
  - Agnóstico
    - Depende del job
- Populares
  - Python
  - NodeJS
  - C#
  - Ruby
- Específicos
  - Golang
  - Java
  - Erlang



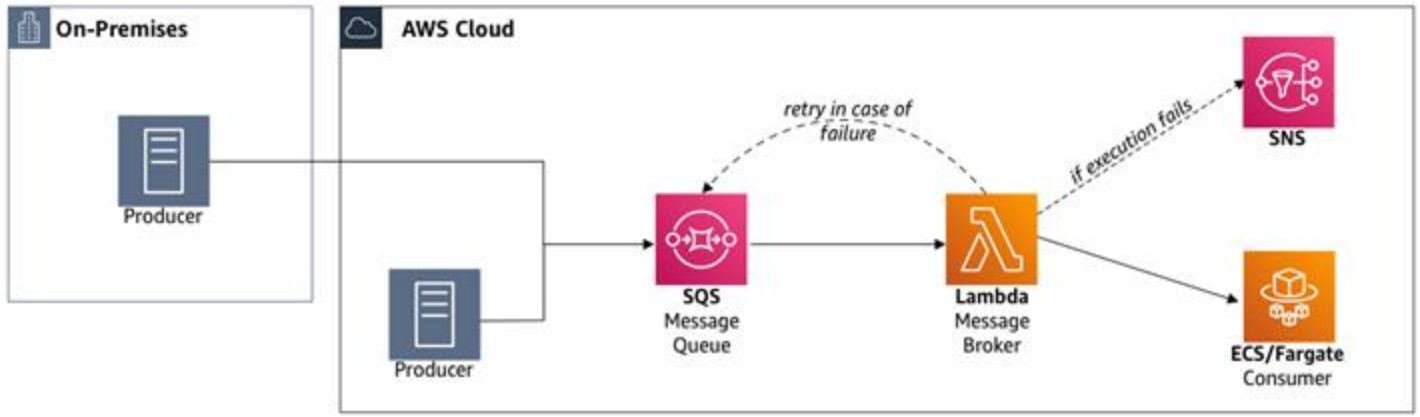
Herramientas: Emisión y coordinación

- Manager
  - Celery
  - Bull
  - Sidekiq
    - Ojo que se paga...
- Broker
  - Apache Kafka
  - RabbitMQ
  - Redis
  - ActiveMQ
- Queues
  - Redis
  - RabbitMQ
  - SQS
- Visibilidad
  - Flower – Solo para Celery





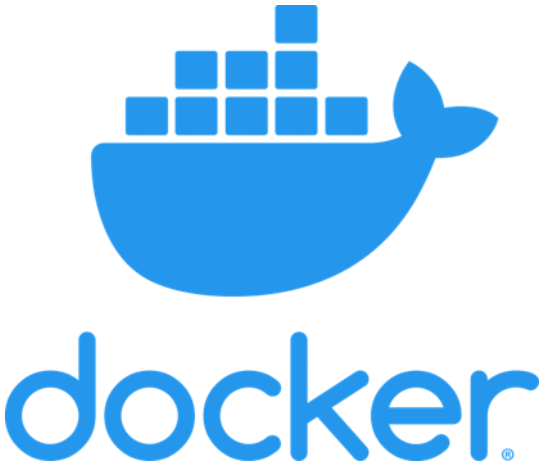




Herramientas: Orquestación de entorno

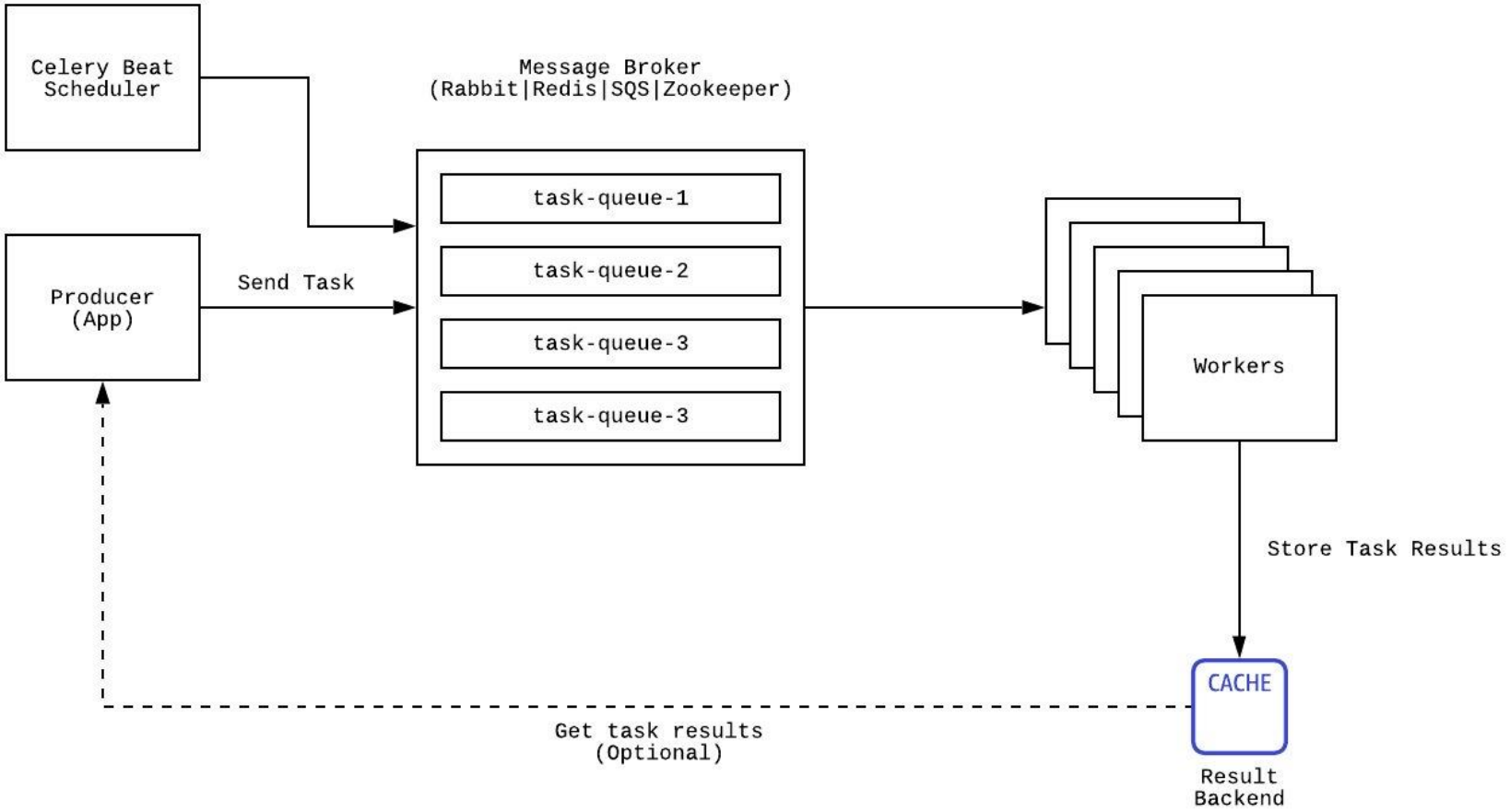
- On-premises
  - Depende del proveedor al que lleguen
- Docker
  - Compose
  - Kubernetes
  - Y todo el universo Docker...
- Serverless
  - AWS
  - GCP
- Otras soluciones custom...
  - AWS batch



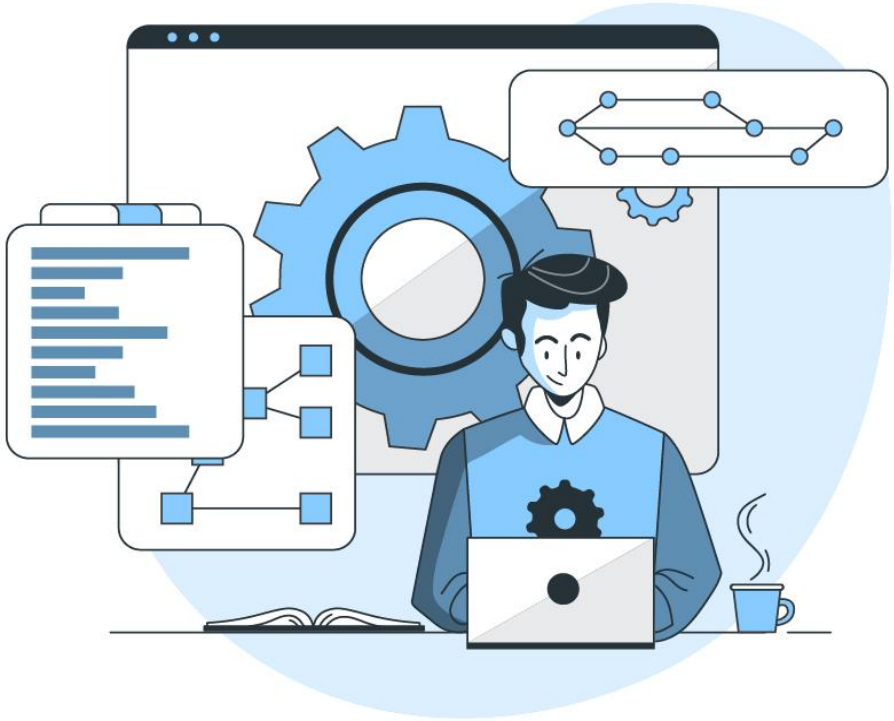




Flujos de ejecución de workers – celery



Demo



Ayudantía 08: Workers

IIC2173 – 2025.2