How to Manage Only Business code

Summary

1. Introduction
2. Graphql Engine
    a. Description
    b. Hasura
    c. demo
3. Workflow Engine
    a. Description
    b. Temporal
    c. demo
4. How this stack is tied together
    a. UML sequence diagram
5. A example workflow
    a. BPMN workflow activity
6. Demo
    a. Monorepo structure
    b. Explain workflow / activity code / client
7. Beyond the POC
8. Real implem

Introduction

Presenters

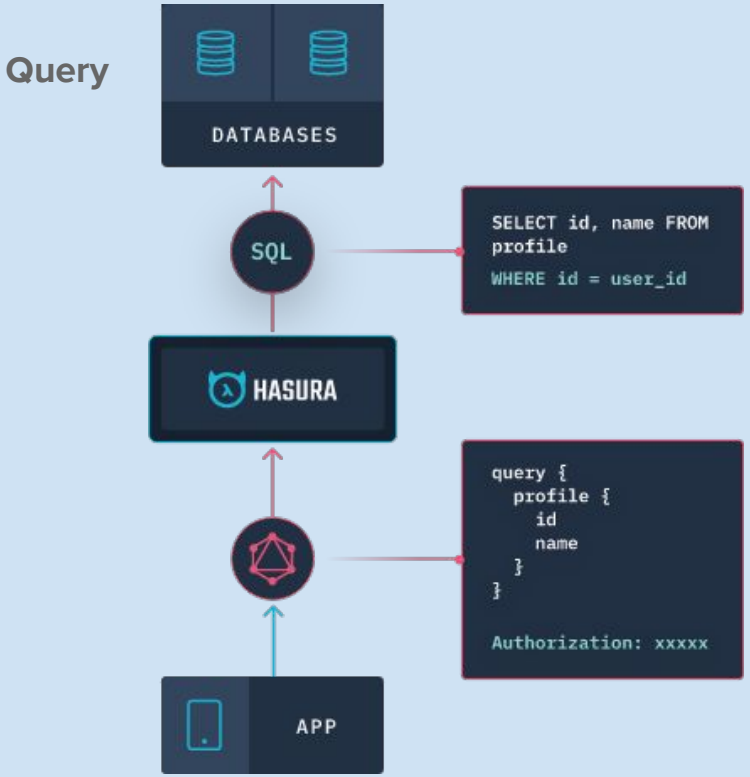
Adrien

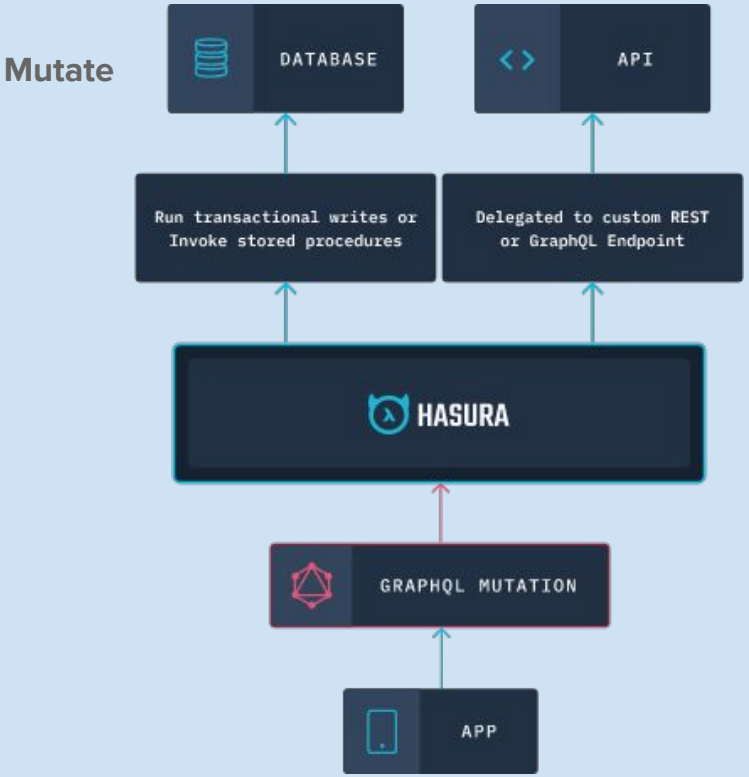
Benoit

Graphql Engine

The Hasura GraphQL Engine makes data instantly accessible over a real-time GraphQL API

Hasura - GraphQL engine



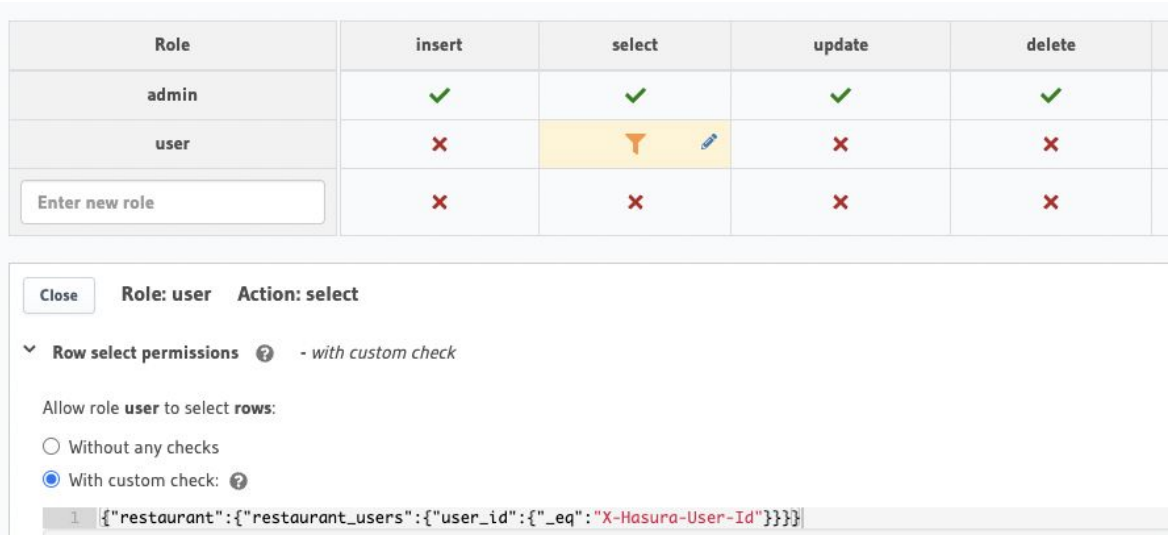


Hasura - GraphQL engine

Hasura - ACL

Hasura use JWT to authenticate users with specific roles.

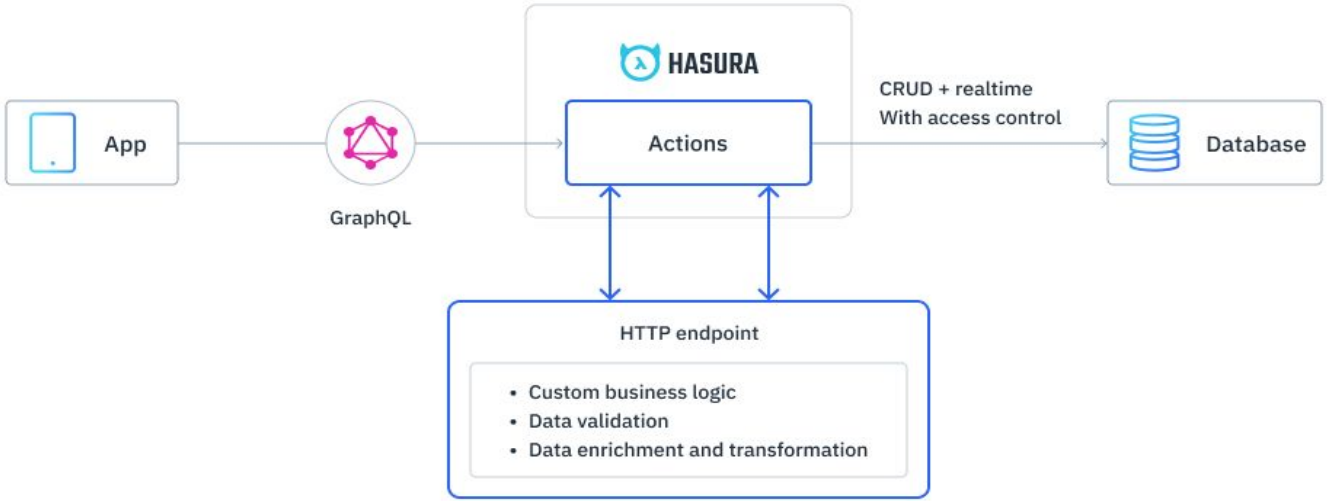
This allows to handle granular permissions on the DB ressources with ease



Restrict restaurant to related users

Hasura - Business logic

1. Event trigger
2. Remote schema
3. Stored procedure / function
4. Actions



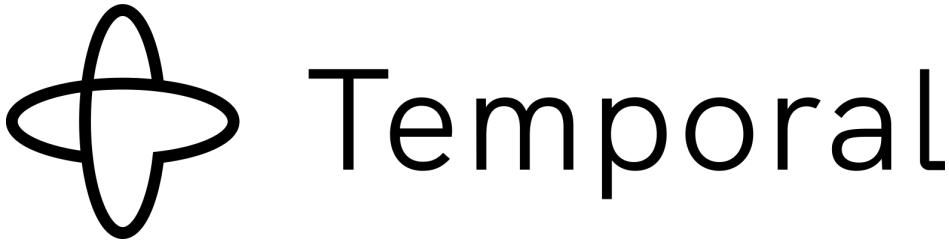
Workflow Engine

A workflow engine is a software application that manages business processes

Temporal

Temporal is a workflow engine that **focus developers on managing business processes**

Temporal **ensures that code executes** reliably, durably, and scalably **while eliminating needless complexity for developers**



Many SDK

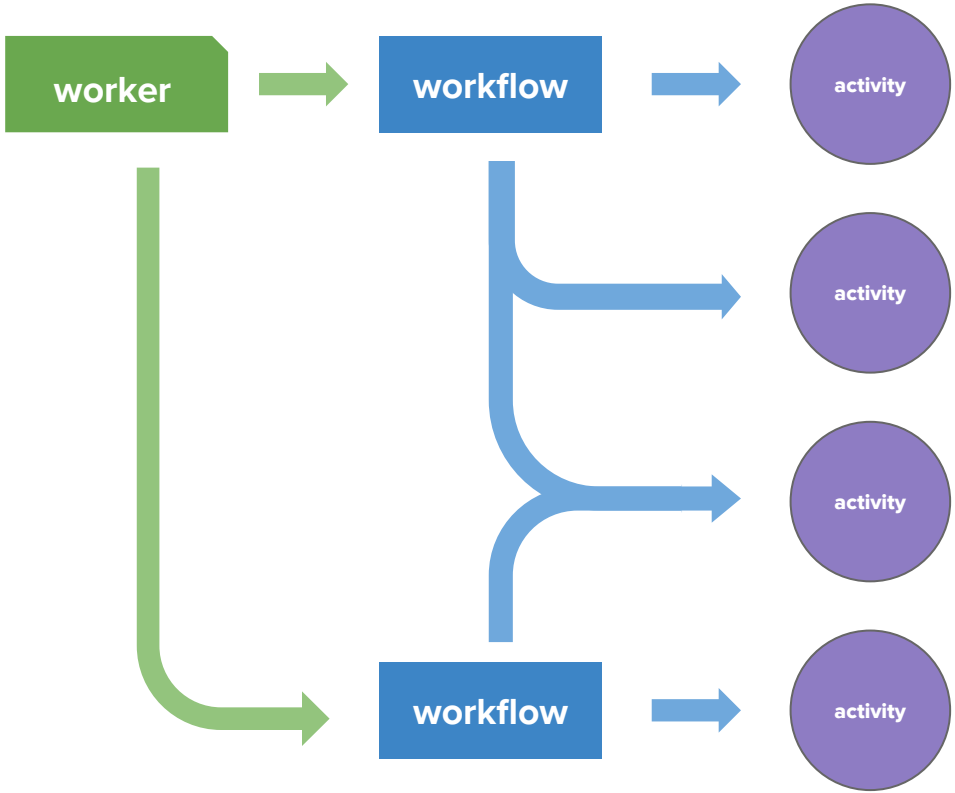
Used by

Temporal - concepts

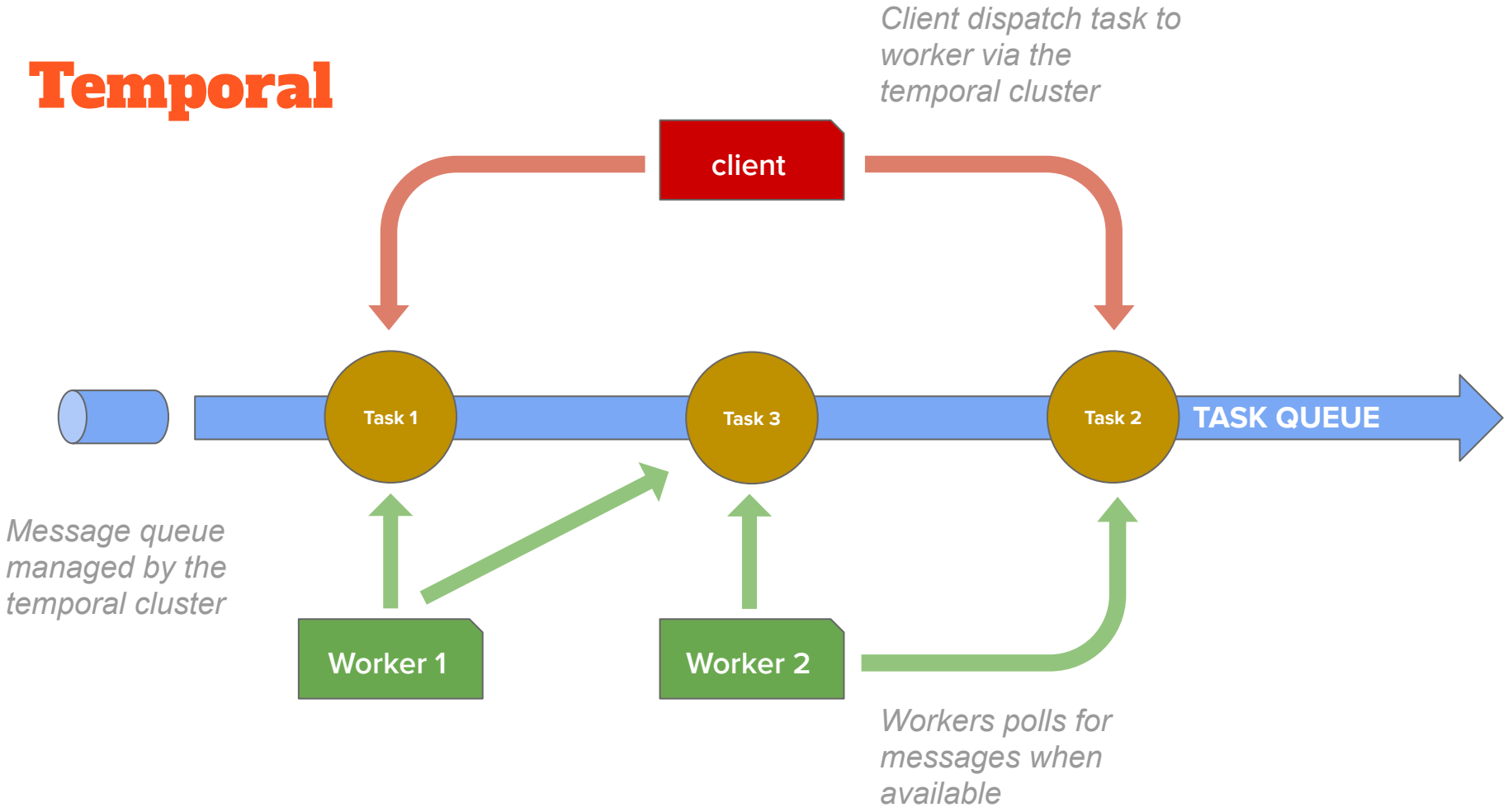
Worker: Listen task queue using temporal sdk, start workflows

Workflow: A business process that **orchestrates the execution of Activities**

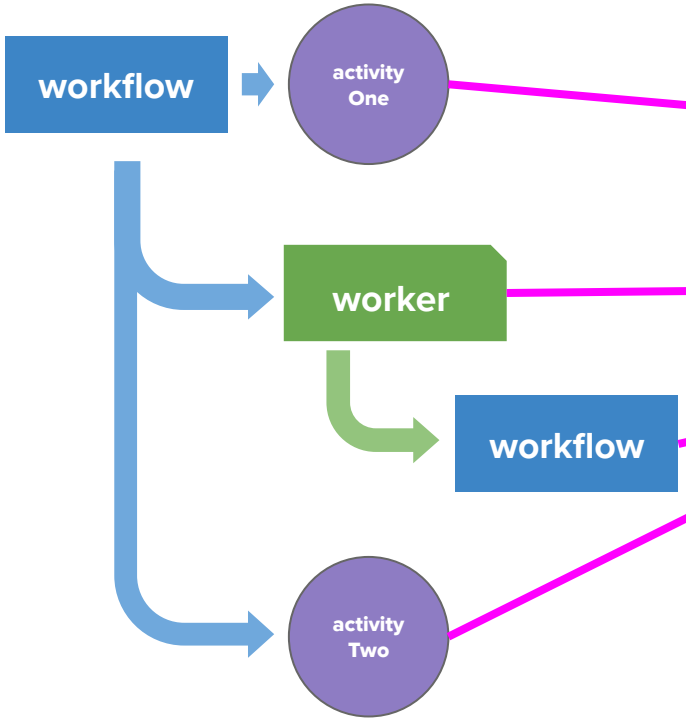
Activity: **An Activity is a normal function** or object method **that executes a single,** well-defined **action** *such as calling another service or sending an email message.*



Temporal

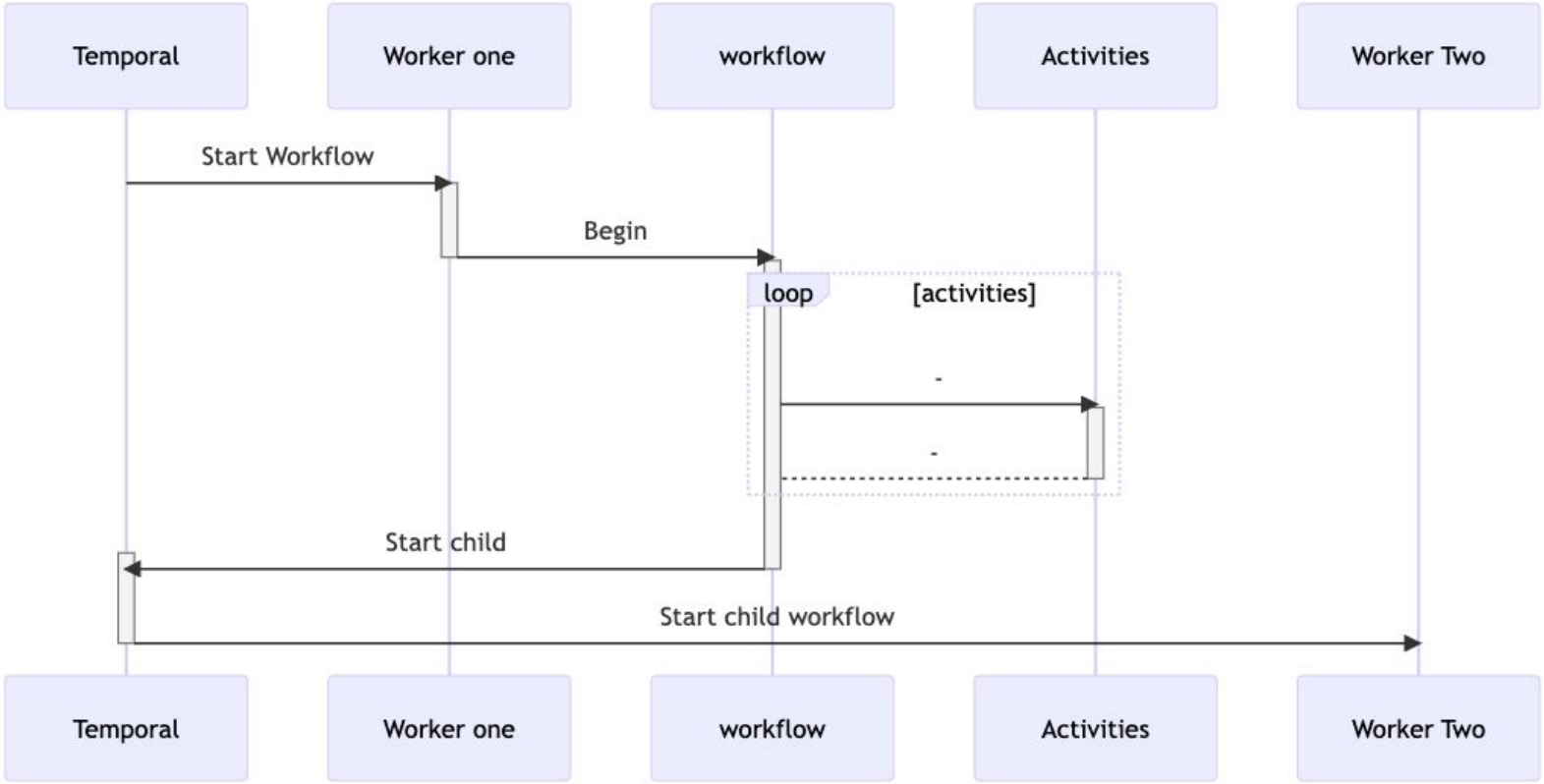

Client dispatch task to worker via the temporal cluster
client
Task 1
Task 3
Task 2
TASK QUEUE
Message queue managed by the temporal cluster
Worker 1
Worker 2
Workers polls for messages when available

Example




```ts
import { executeChild } from '@temporalio/workflow'

// a workflow
export const workflow = async (params) => {

  // the first activity with the initial workflow params
  const resultOne = await activityOne(params)

  // Execute a child workflow and wait for the result
  const resultChild = await executeChild('child-one', {
    args: [resultOne],
    taskQueue: 'task-queue-one',
  })

  // Run the second activity with previous results
  const resultTwo = await activityTwo(
    resultOne,
    resultChild,
  )

  // return the result of the second activity
  return resultTwo
}
```

A workflow sequence



A Workflow is a Business Process

So business expert need to understand it and take care of its conception

Ubiquitous Language With

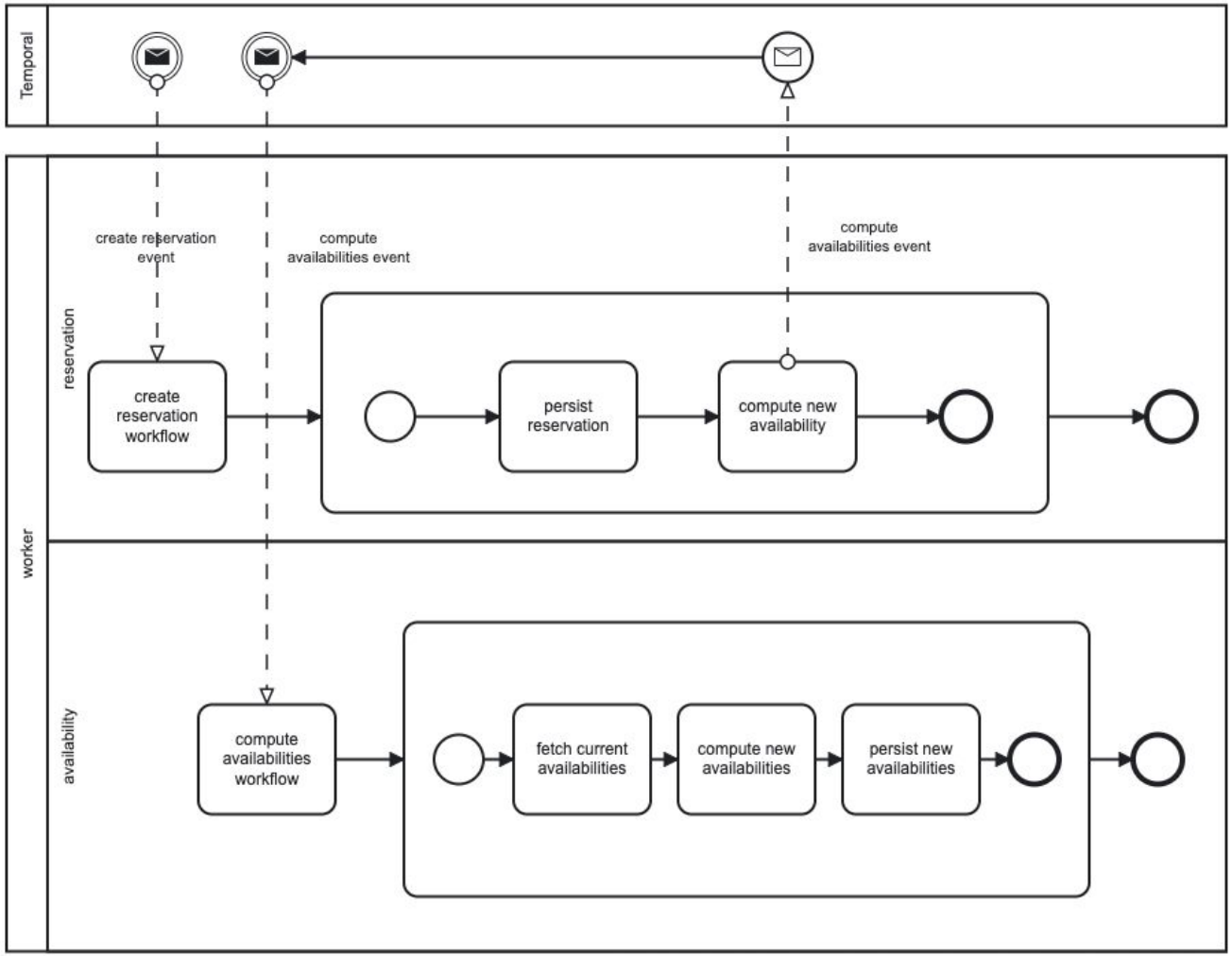
BPMN

Business Process Model and Notation

An example workflow with BPMN

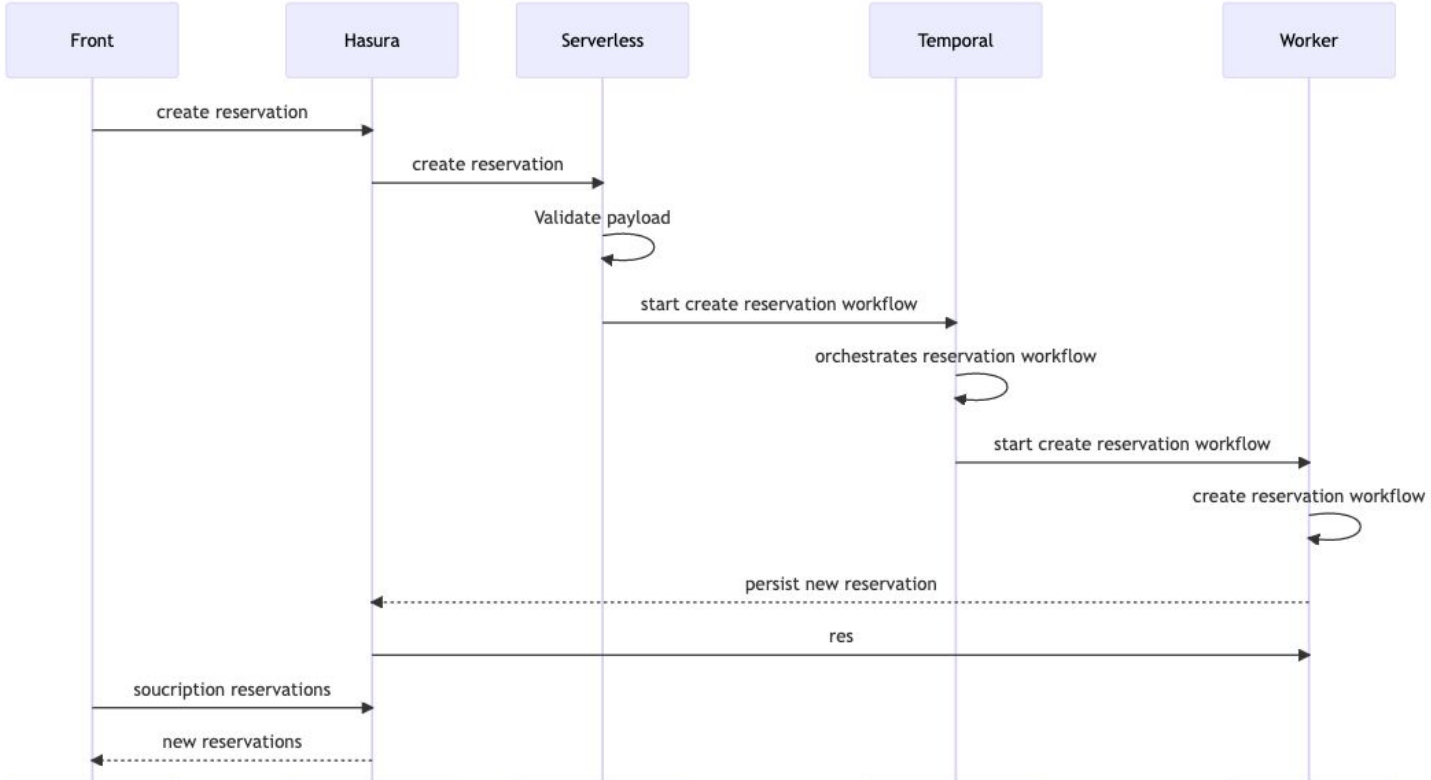
Already use at The Fork, the mobile team loves it

You will love it too



Tied together

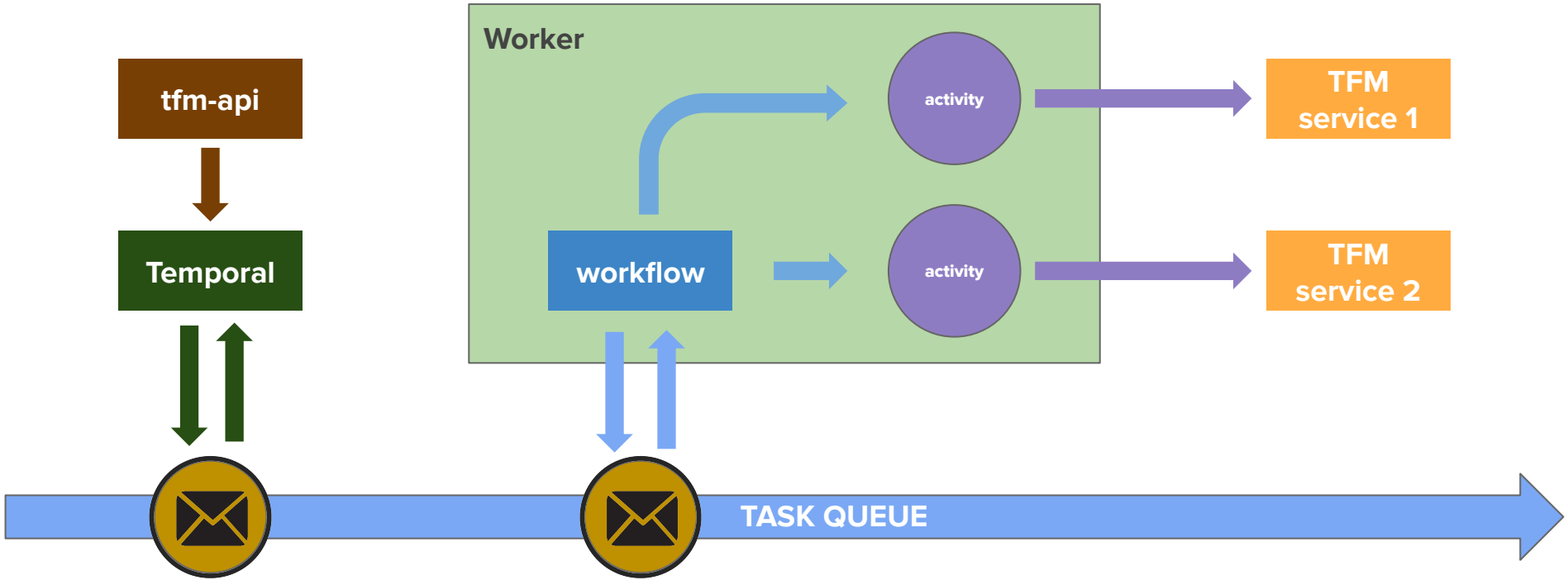
A complete sequence



Demo

Beyond the POC

Using Temporal at TheFork



Simplified TODO

1. Get access to a Temporal cluster
2. Create a monorepo containing workers, workflows and activities
3. Add Temporal client SDK to tfm-api
4. Work on the BPMN for the workflows involving Product people
5. Implements the workflows
6. Profits!

Conclusion

Don’t overthink Hasura (it’s good stuff for poc but not relevant)

Think about Temporal a lot !

Maintain only business code (the important one)

Thank you for your attention

Adrien Louis-Rossignol
Benoit Deglane

Questions ?