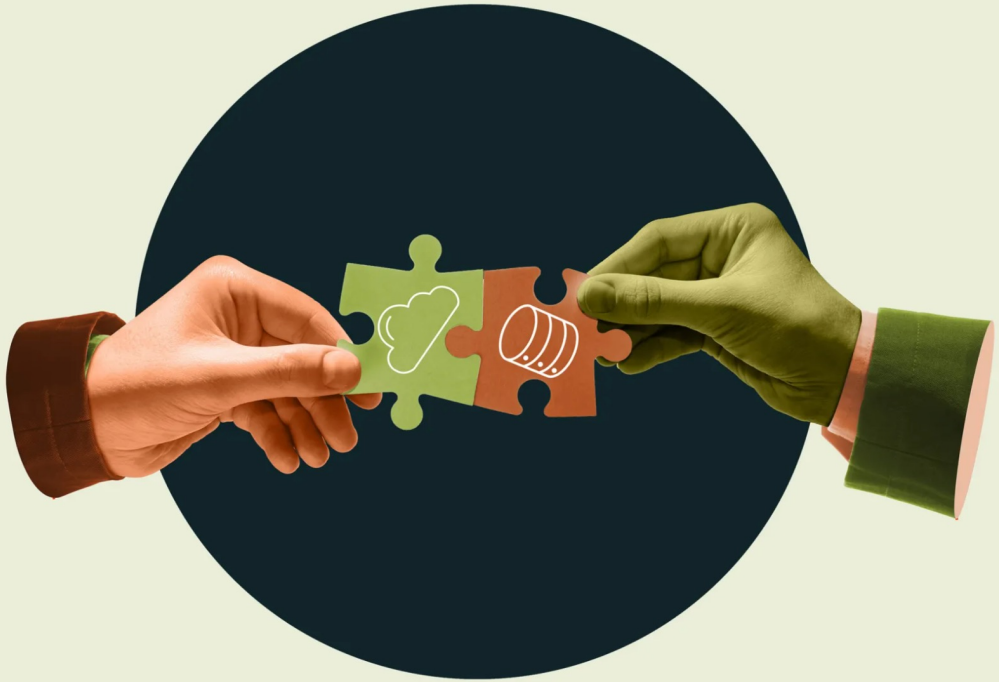
Databases and Serverless are Made for Each Other

Qian Li
Peter Kraft
Co-Founders @ dbos.dev

DBOS

© 2024 dbos.dev



Practice Nov 20 2024

Transactions and Serverless are Made for Each Other

Serverless cloud platforms should be used to deploy stateful applications.

Qian Li and Peter Kraft

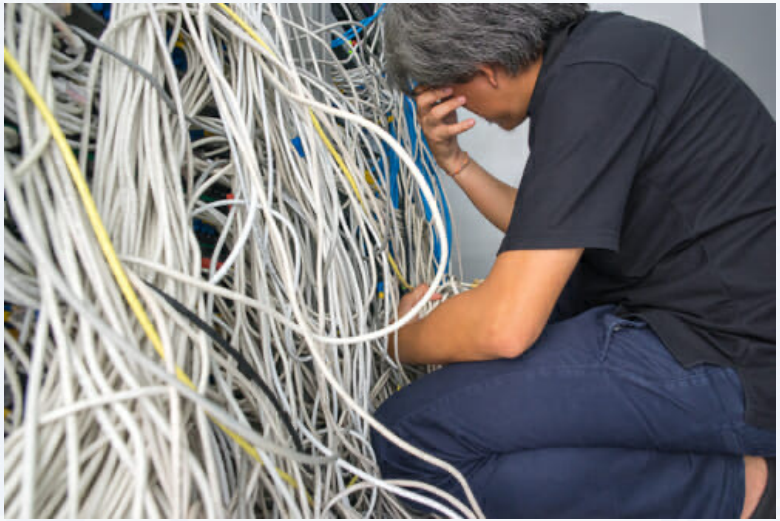
Architecture and Hardware

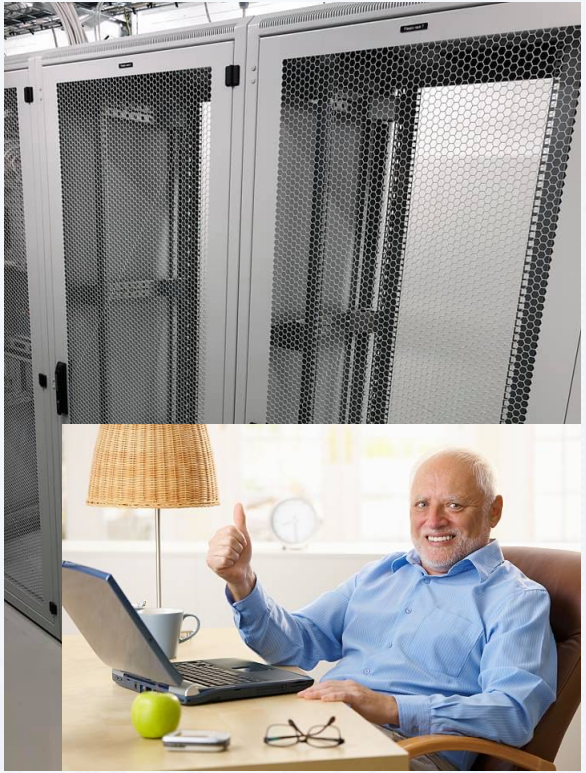
Feedback

© 2024 dbos.dev

Serverless Simplifies Infra Management

- Autoscaling
- Pay-as-you-go





© 2024 dbos.dev

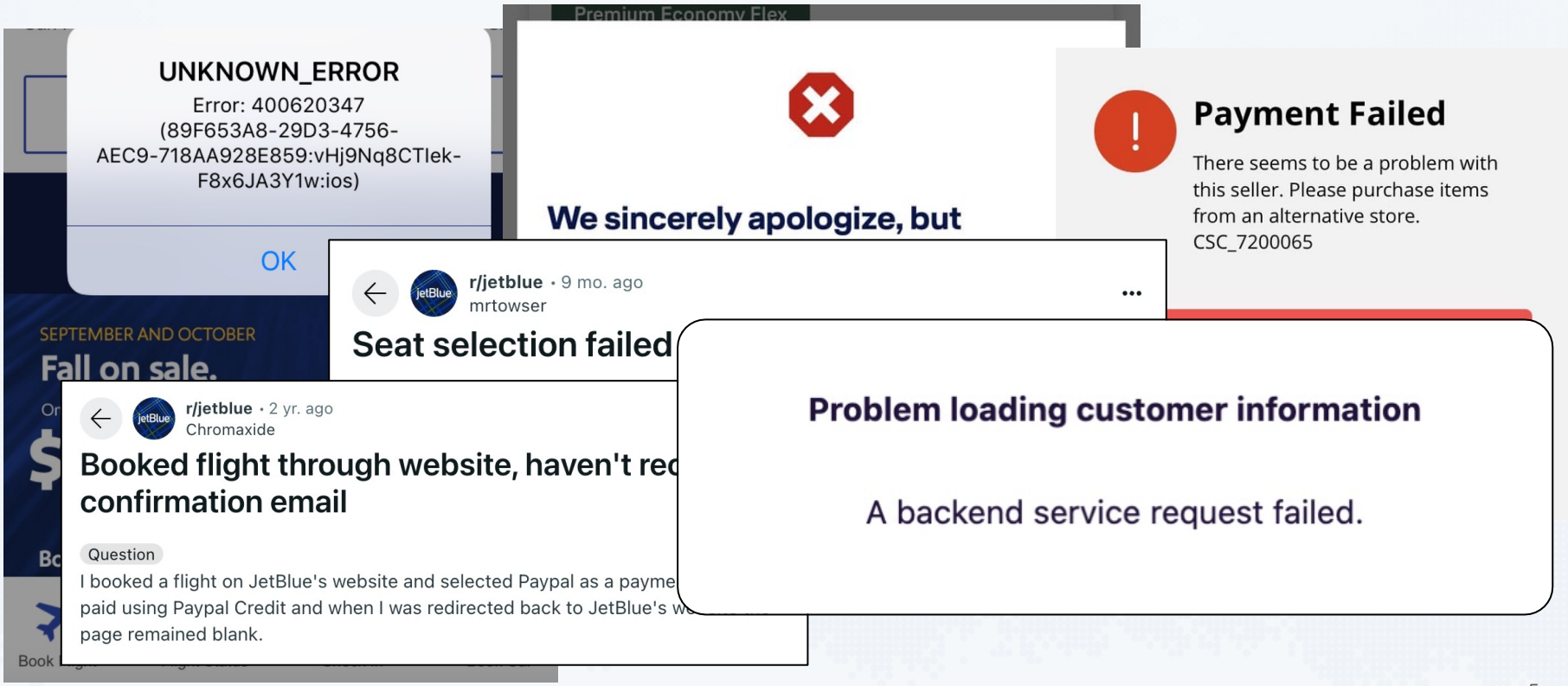
But Serverless Is Usually *Stateless*

Mismatch for stateful apps: long-running with real-world interactions

- Payment processing
- Cron jobs
- Data pipelines
- Event processing (e.g., Kafka consumer)
- AI agents 🤖
- ...many more!

© 2024 dbos.dev

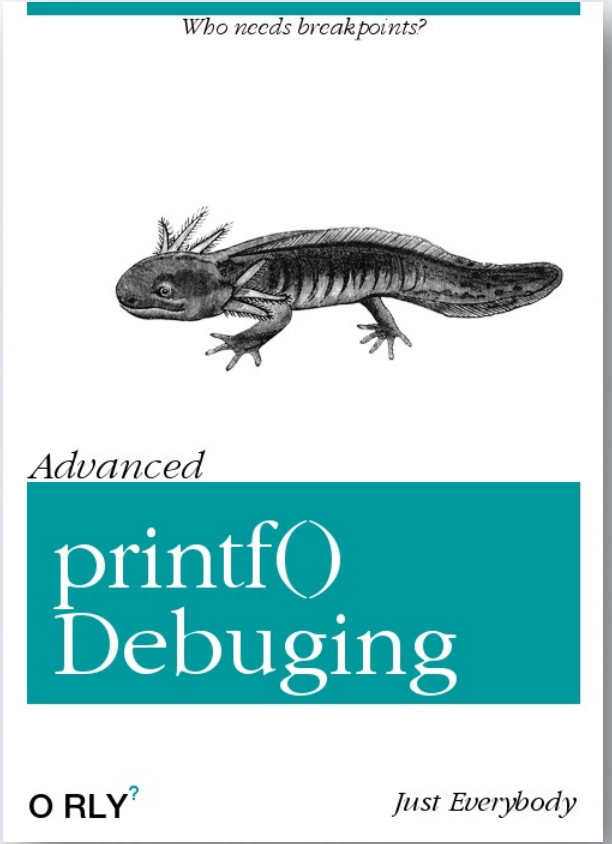
Building Stateful Apps is Hard



© 2024 dbos.dev

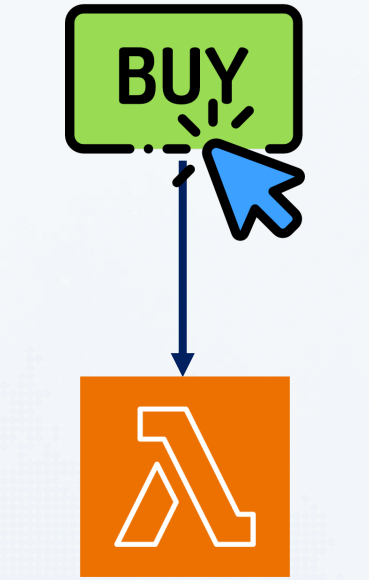
Serverless Makes It Harder

- Timeouts
- Fault tolerance (e.g., retries)
- No interactivity
- Observability



© 2024 dbos.dev

Example: Serverless E-Commerce

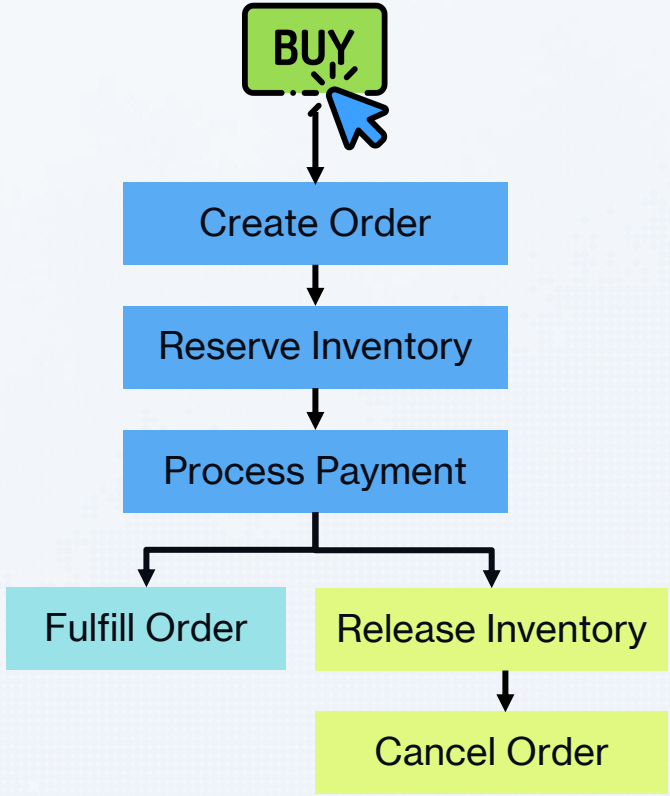


© 2024 dbos.dev

I Want to Just Write Code

widget_store.py

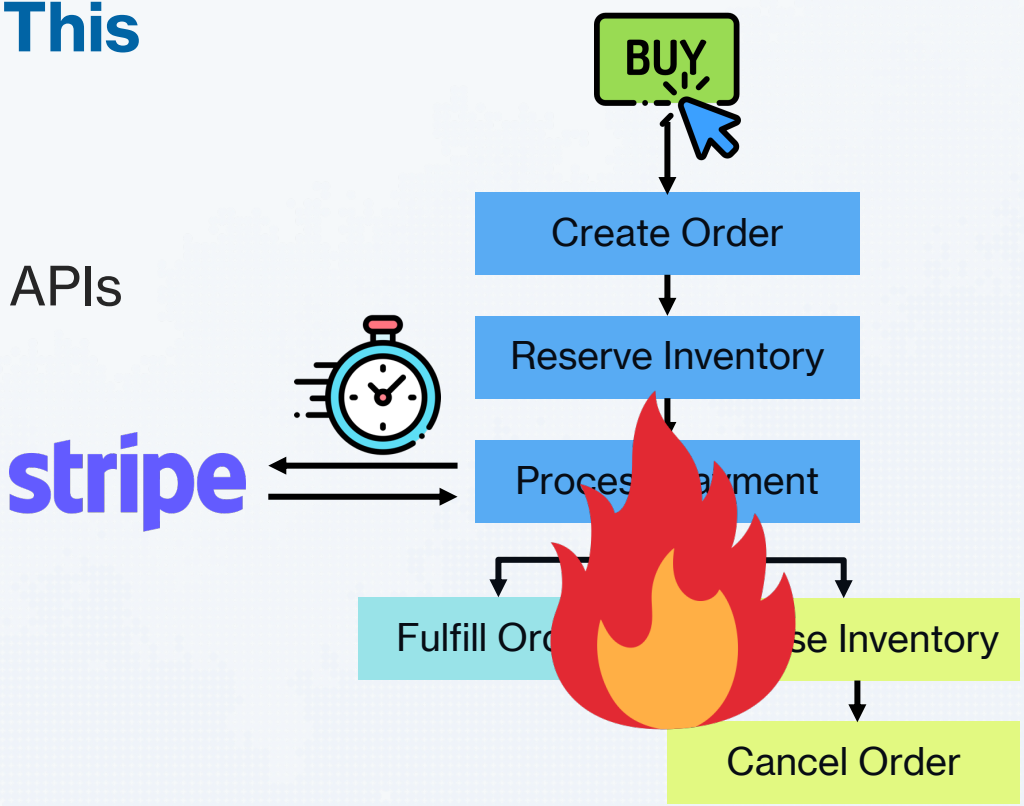
```python
def checkout(items):
    order = create_order()
    reserve_inventory(order, items)
    payment_status = process_payment(order, items)
    if payment_status == 'paid':
        fulfill_order(order)
    else:
        undo_reserve_inventory(order, items)
        cancel_order(order)
```



© 2024 dbos.dev

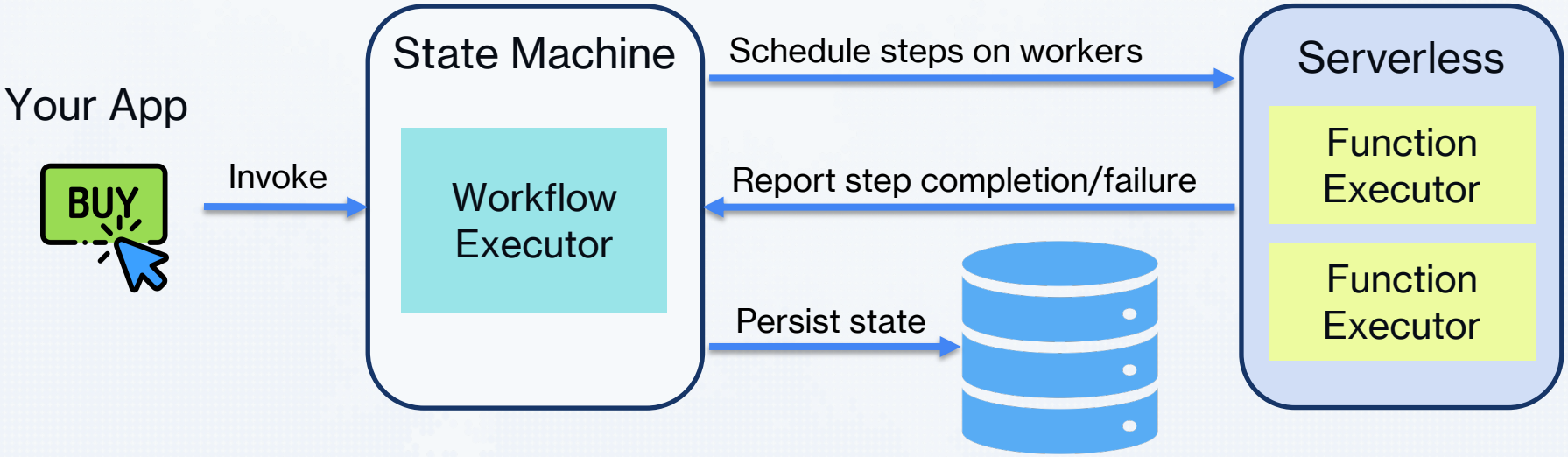
But Serverless Can't Do This

- Timeouts
- Can't invoke async external APIs
- Fault tolerance



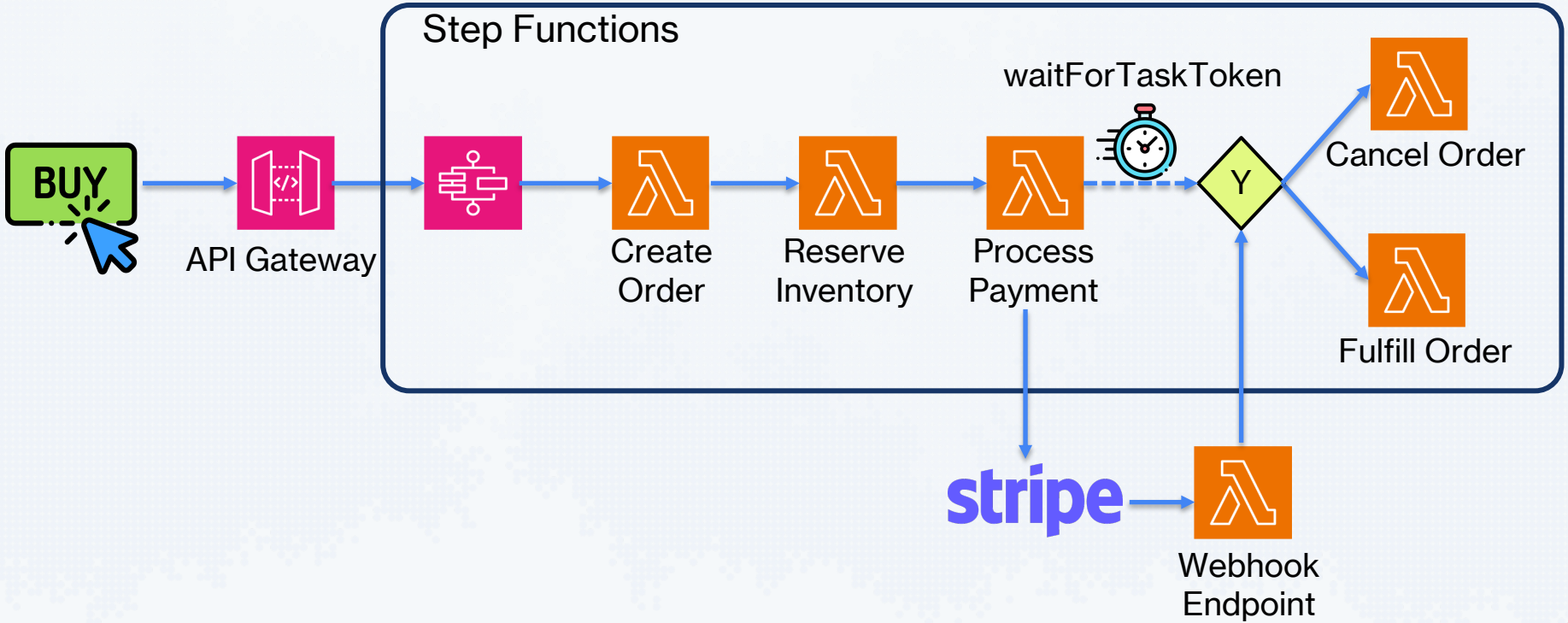
© 2024 dbos.dev

Solution: External Durable Execution Engine



© 2024 dbos.dev

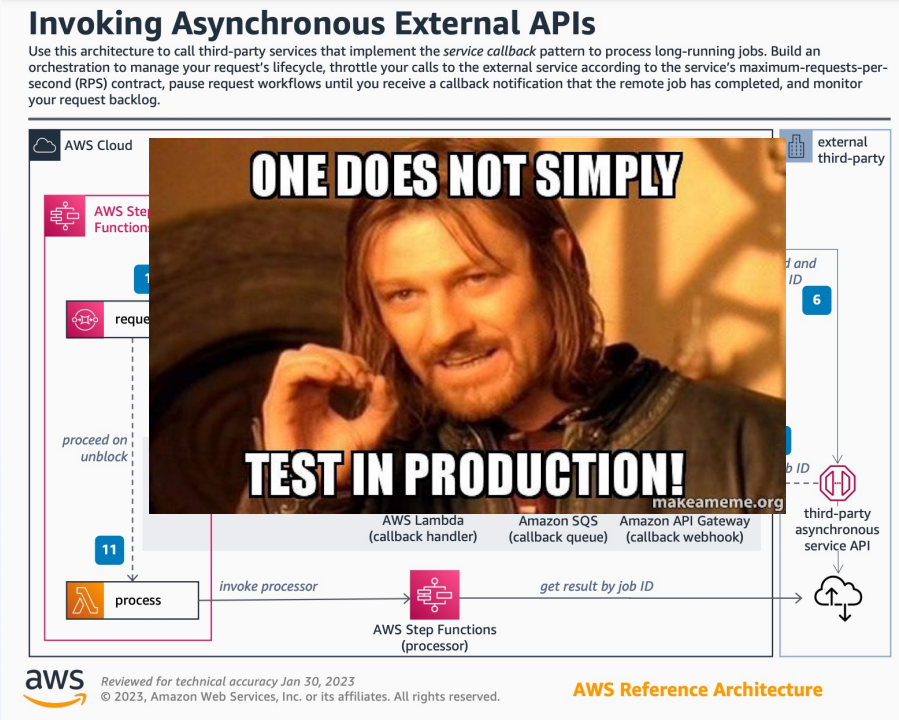
I Have to Make Things Complicated



© 2024 dbos.dev

It Works! But Heavyweight

- Complicated Dev **and** Ops
- Hard to maintain/test
- Bad performance
- Expensive $$$



[1] https://d1.awsstatic.com/architecture-diagrams/ArchitectureDiagrams/invoking-asynchronous-external-apis-ra.pdf

© 2024 dbos.dev

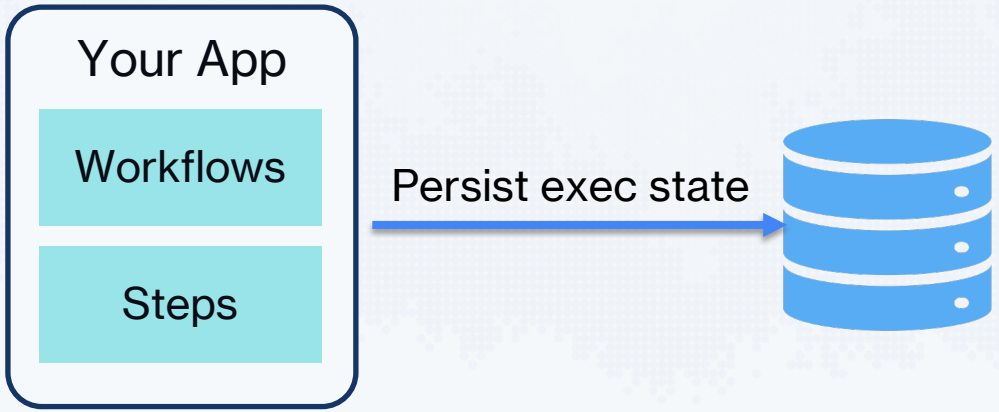
How to make serverless work for stateful apps?

© 2024 dbos.dev

... Databases!

© 2024 dbos.dev

A New Serverless Architecture

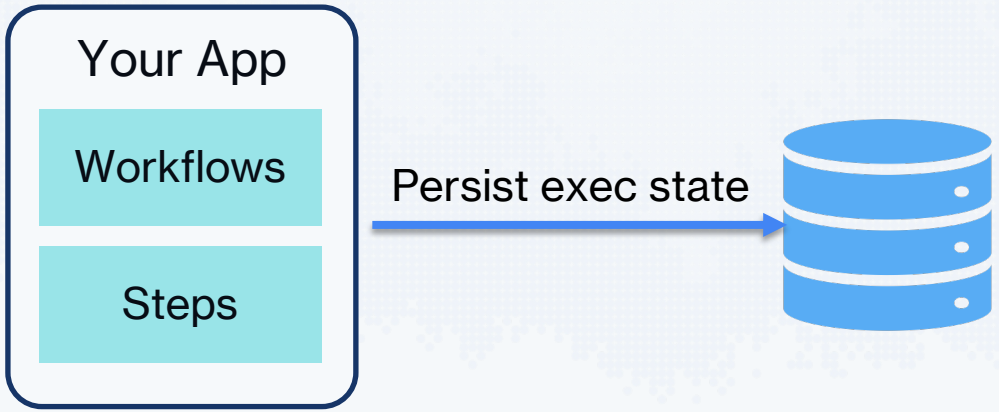


© 2024 dbos.dev

Key Insight

- To make a stateful app serverless, persist its execution state in a database
- Then you can safely execute it in ephemeral environments

- How to implement it?



© 2024 dbos.dev

Implementing Durable Execution as a Library

widget_store.py

```python
def create_order():
  ...


def checkout(items):
   order = create_order()
   reserve_inventory(order, items)
   payment_status = process_payment(order, items)
   if payment_status == 'paid':
       fulfill_order(order)
   else:
       undo_reserve_inventory(order, items)
       cancel_order(order)
```

- Write normal functions

© 2024 dbos.dev

Implementing Durable Execution as a Library

widget_store.py

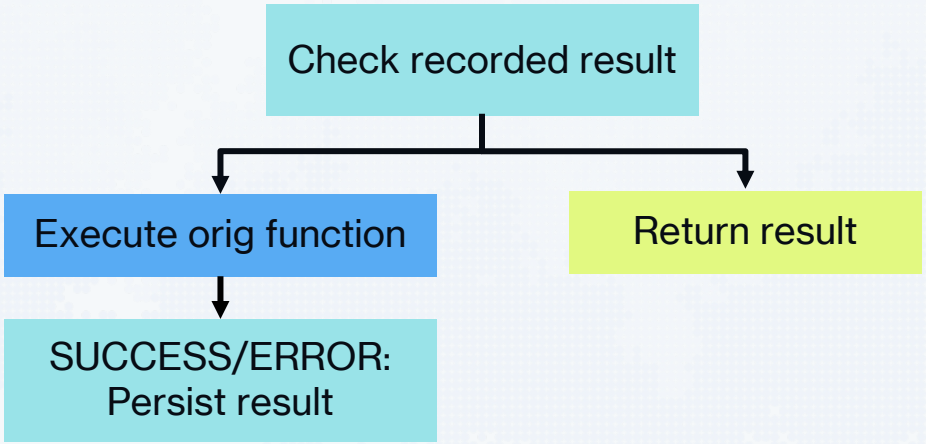
```python
@DBOS.step()
def create_order():
  ...

@DBOS.workflow()
def checkout(items):
    order = create_order()
    reserve_inventory(order, items)
    payment_status = process_payment(order, items)
    if payment_status == 'paid':
        fulfill_order(order)
    else:
        undo_reserve_inventory(order, items)
        cancel_order(order)
```

- Write normal functions
- Annotate code with decorators
  - `@DBOS.workflow()`
  - `@DBOS.step()`
  - `@DBOS.transaction()`

© 2024 dbos.dev

@step

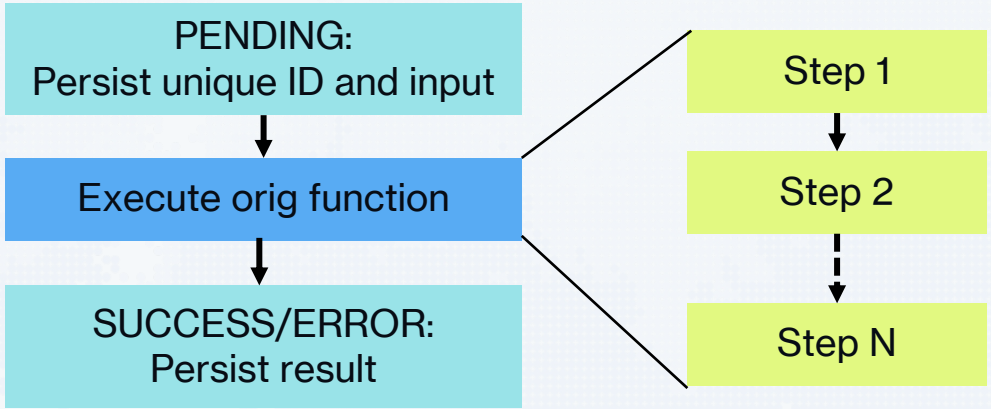
- Wrap functions with durable execution logic
- Never re-execute a step after completion



© 2024 dbos.dev

@workflow

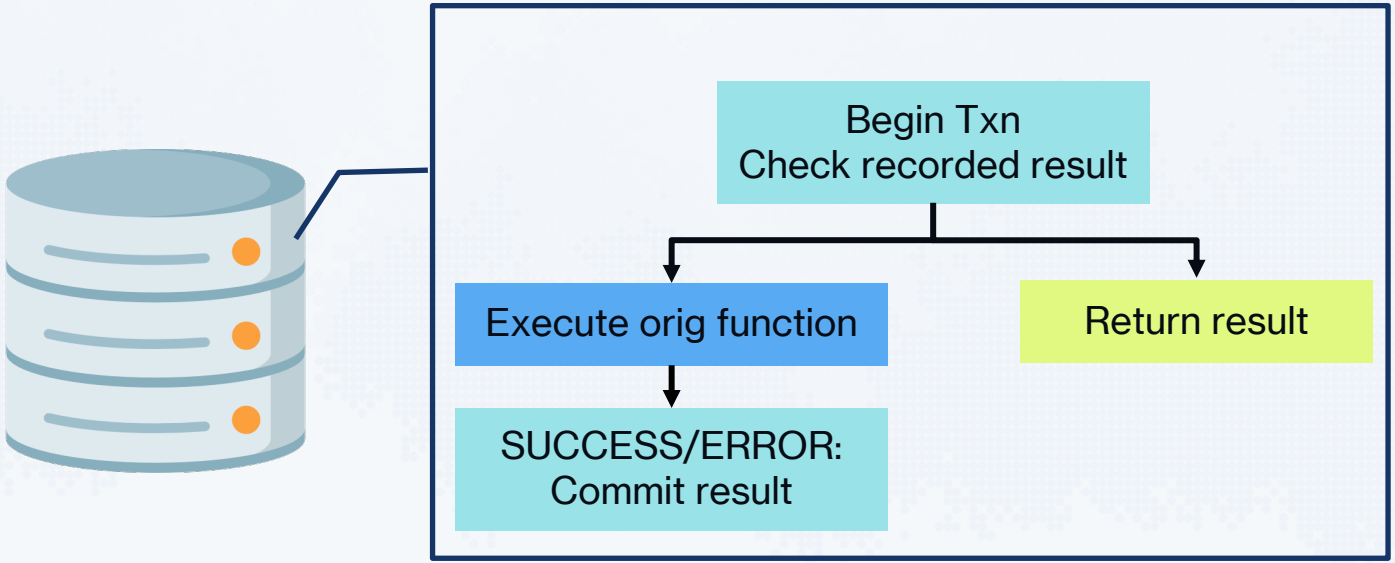
- Must be deterministic (e.g., the order of steps)
- Restart on recovery, run to completion



© 2024 dbos.dev

`@transaction`

- A special case to `@step`, guaranteeing **exactly once**
- Record the output in the same DB transaction



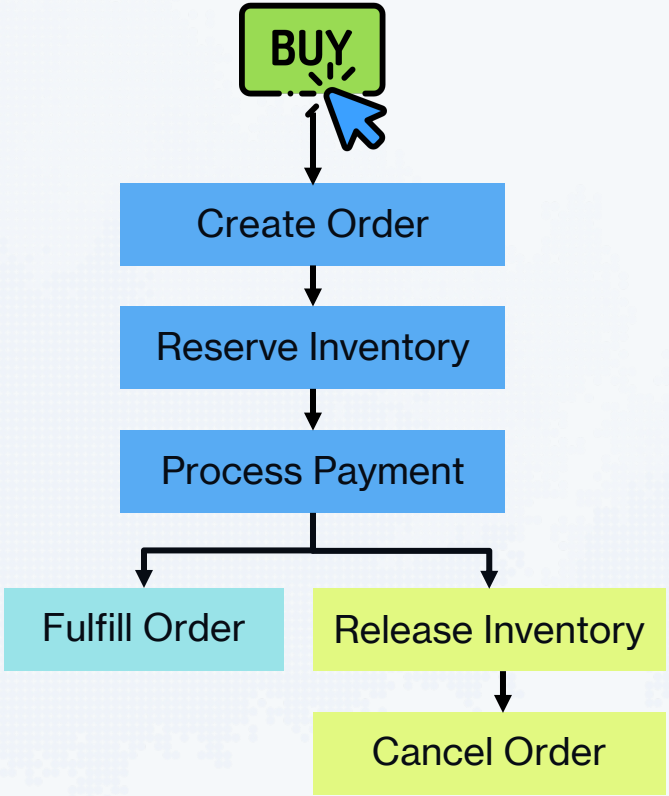
© 2024 dbos.dev

Other Primitives Built On Top of the Database

- Queues
- Send/Recv (notifications)
- Cron jobs
- Durable sleep

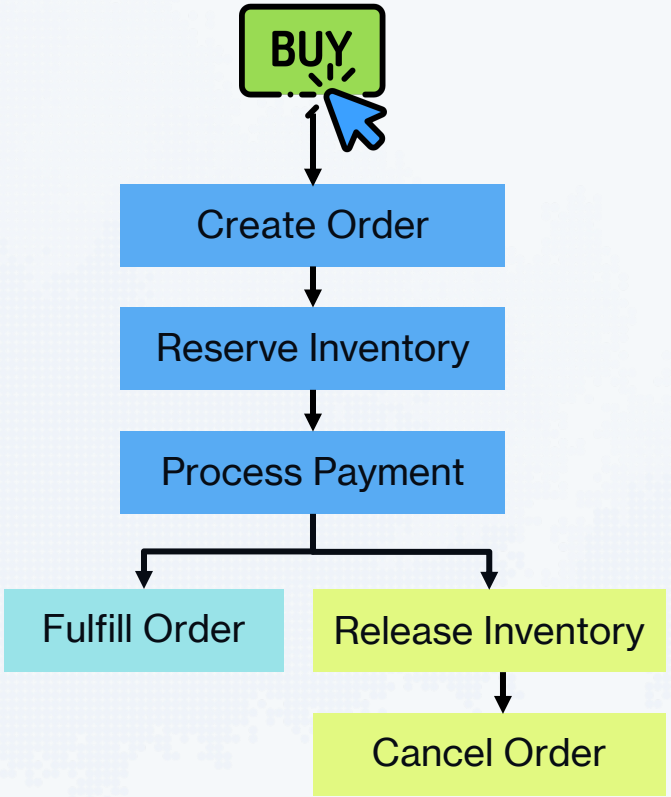
© 2024 dbos.dev

How DBOS Works:



© 2024 dbos.dev

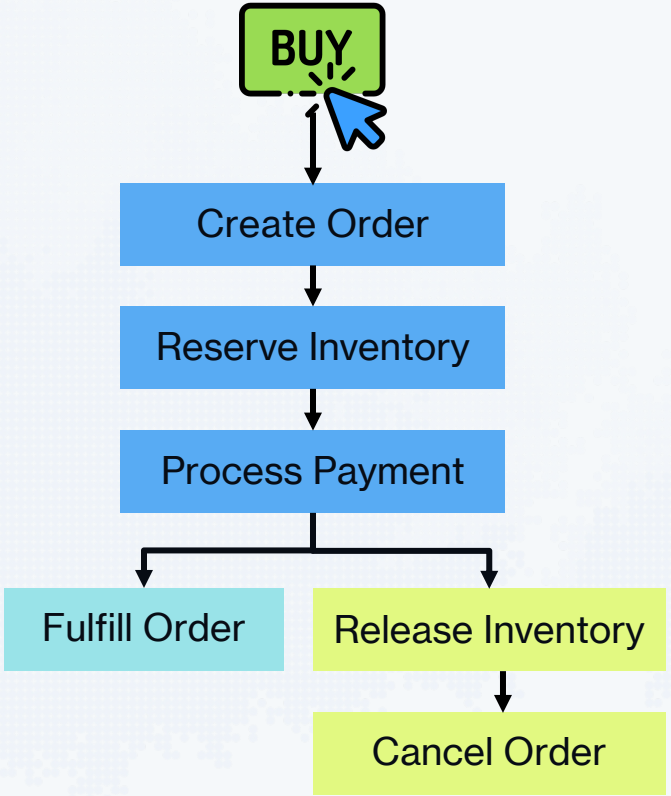
Record the Workflow Start

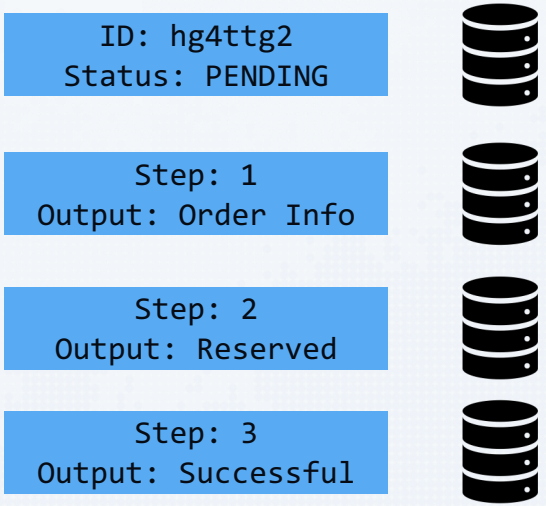



```
ID: hg4ttg2
Status: PENDING
```

© 2024 dbos.dev

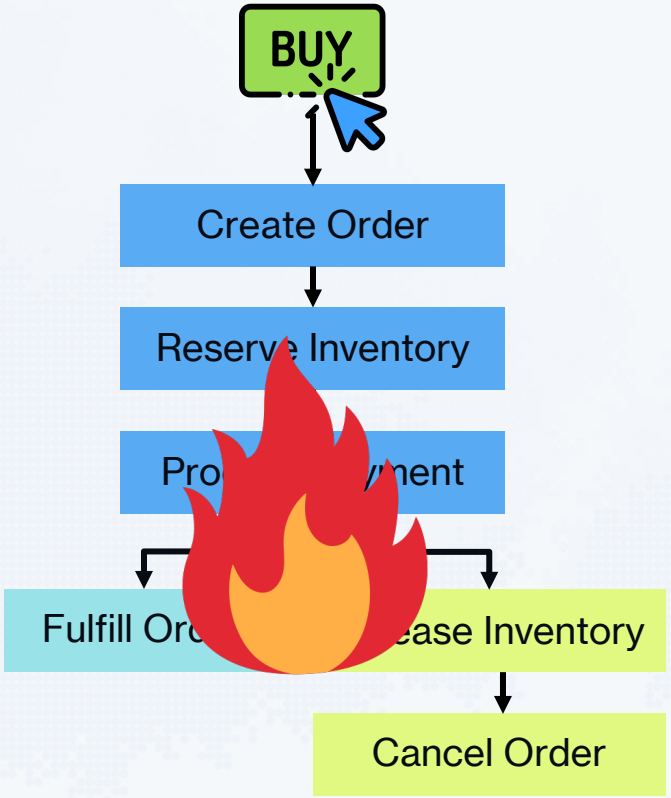
Record Each Step's Output

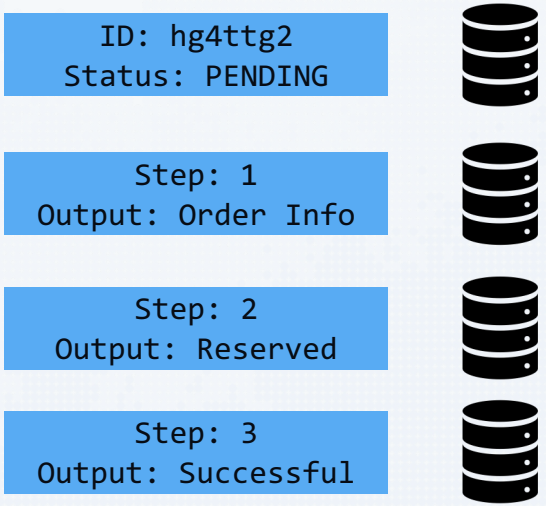




© 2024 dbos.dev

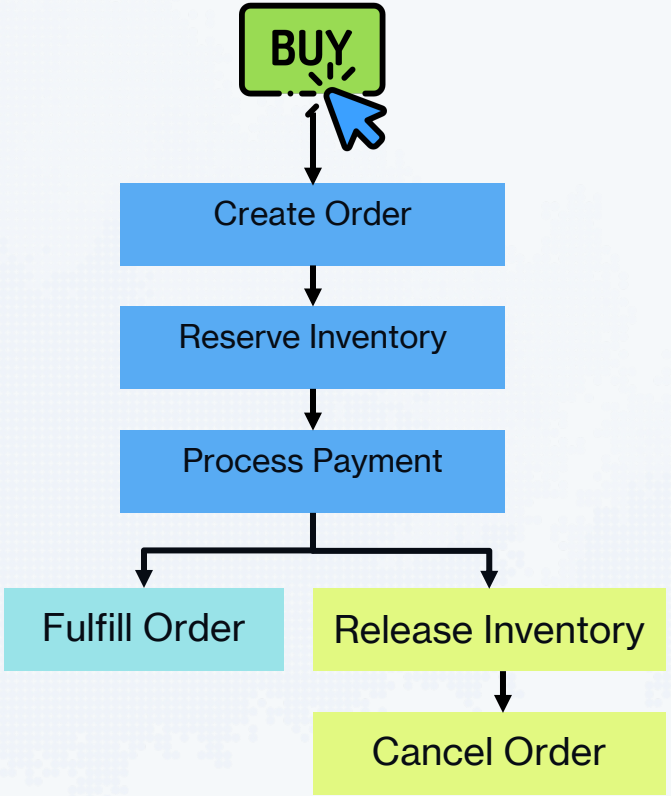
What If It Crashes?

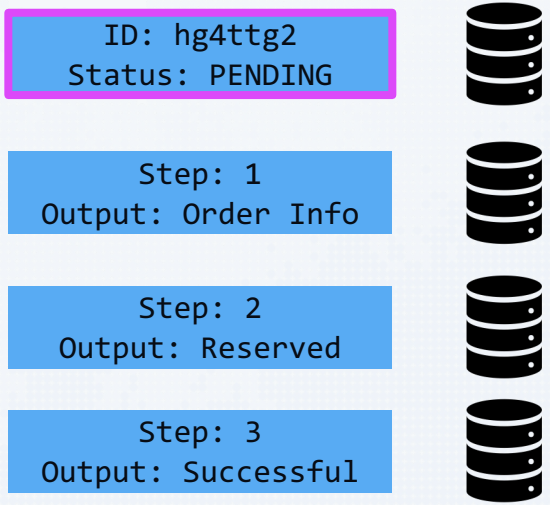




© 2024 dbos.dev

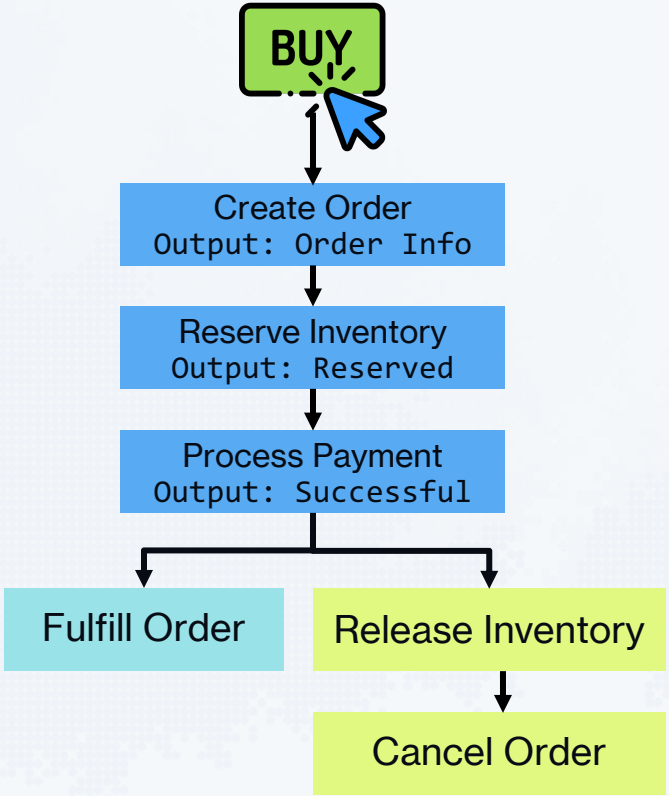
Restart and Check Recorded Outputs

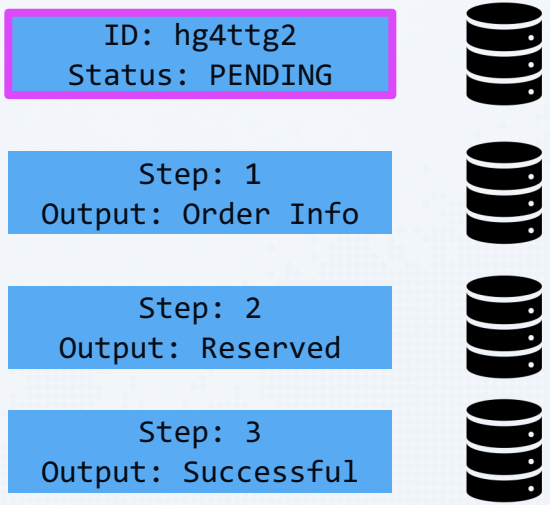




© 2024 dbos.dev

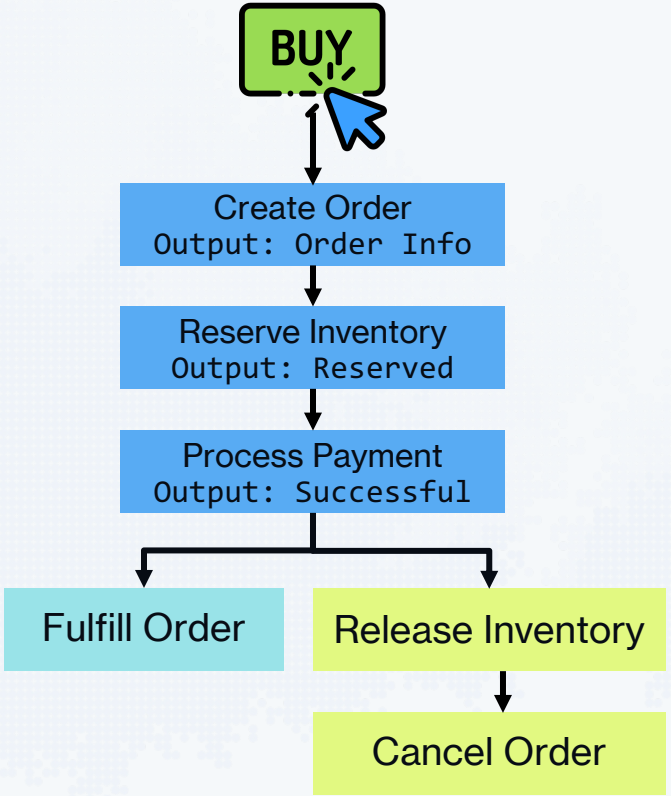
Restart and Check Recorded Outputs

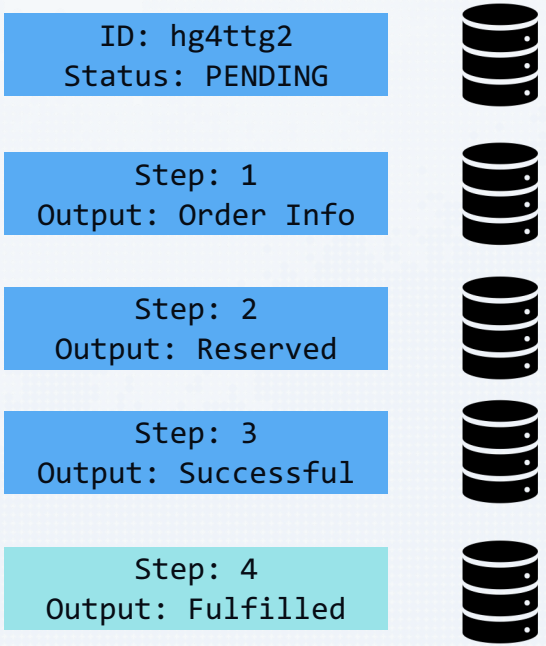




© 2024 dbos.dev

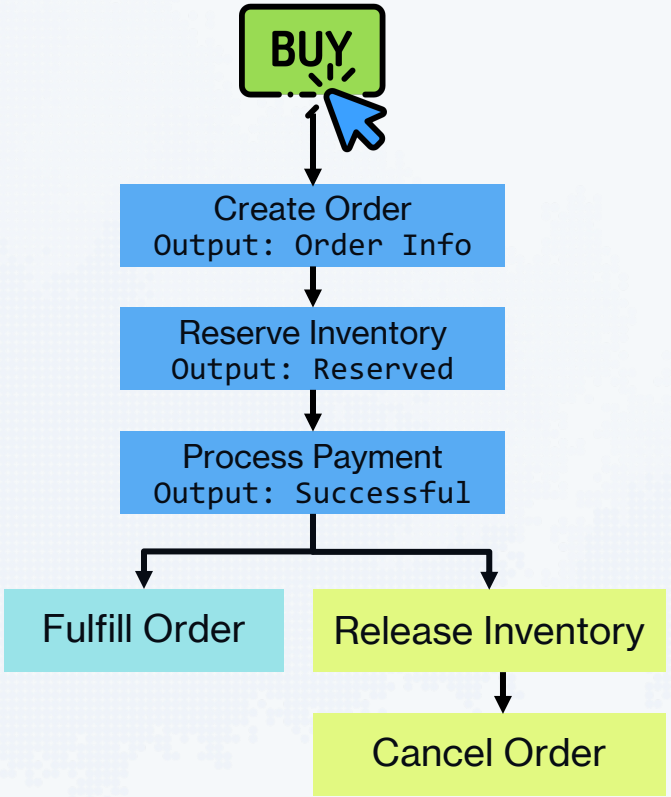
Resume From Where It Left Off

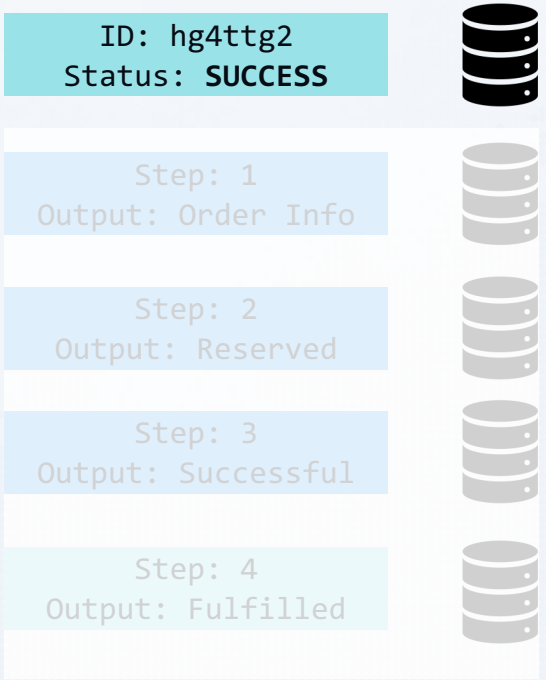




© 2024 dbos.dev

Run to Completion





© 2024 dbos.dev

Widget Store Demo

- Live: https://demo-widget-store.cloud.dbos.dev/
- Tutorial: https://docs.dbos.dev/python/examples/widget-store

DBOS

Shop internals

Inventory

Restock

	Product	Available
	Premium Quality Widget	98

Orders

Order ID	Status	Progress until dispatch
913	CANCELLED	
912	CANCELLED	
911	DISPATCHED	

Widget Store

Premium Quality Widget — Only **98** left!

Enhance your productivity with our top-rated widgets!

Buy Now for $99.99

Server Tools

Crash the application at any time to simulate a server failure or service interruption. After crashing, the application will momentarily become inaccessible but then resume from exactly where it left off before the crash. Try it and see!

Crash the Application

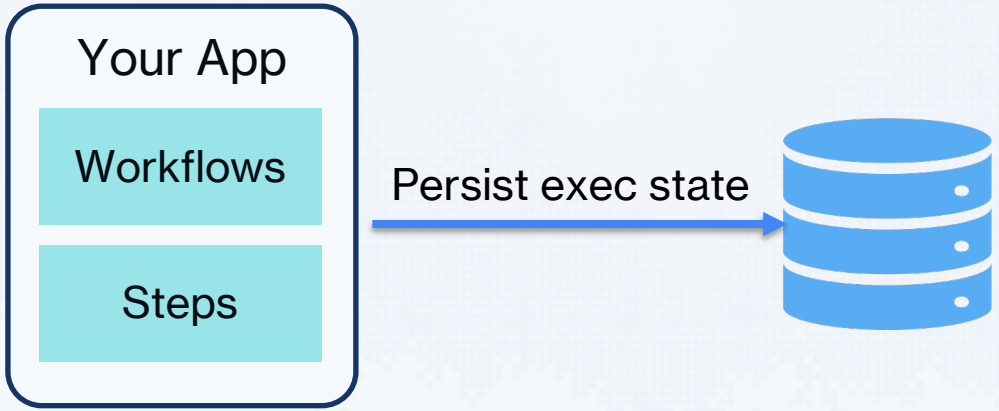
© 2024 dbos.dev

Takeaways

- Persist stateful functions' execution state in a database

  ➔ Simplify serverless architecture

- Databases 💙 Serverless



© 2024 dbos.dev

Try it out & Talk to us!

https://dbos.dev

Open Source:

https://github.com/dbos-inc/dbos-transact-py

Qian Li

Peter Kraft

© 2024 dbos.dev