The Mess We're (Still) In

🌈🧘‍♀️ Unbounded Parallelism, True Names, & Keeping CALM 🎱✨

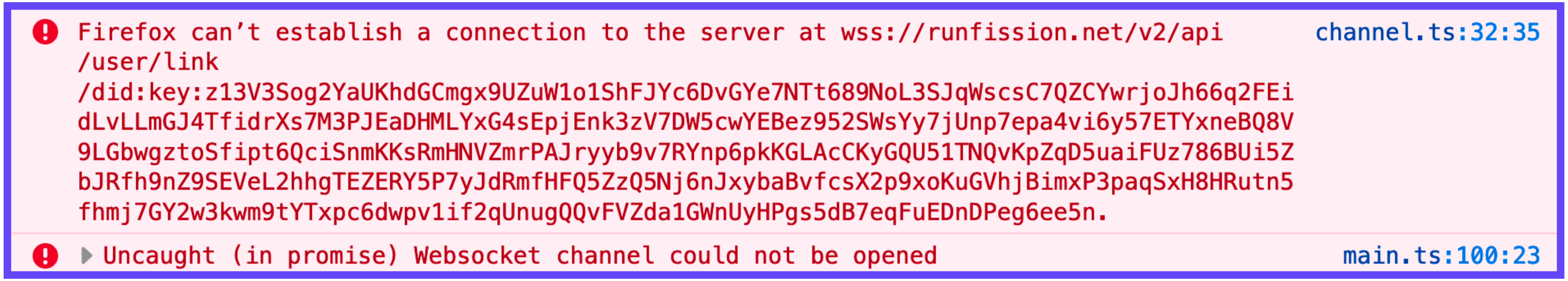
The Mess We're (Still) In

The Mess We're (Still) In

> A distributed system is one in which the failure of a computer you didn't even know existed can ***render your own computer unusable***
>
> — Leslie Lamport

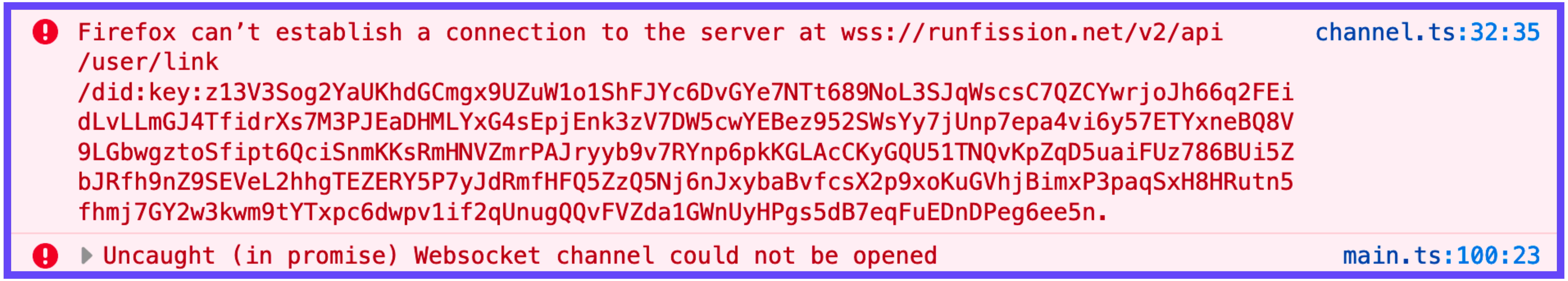
The Mess We're (Still) In

The Mess We're (Still) In

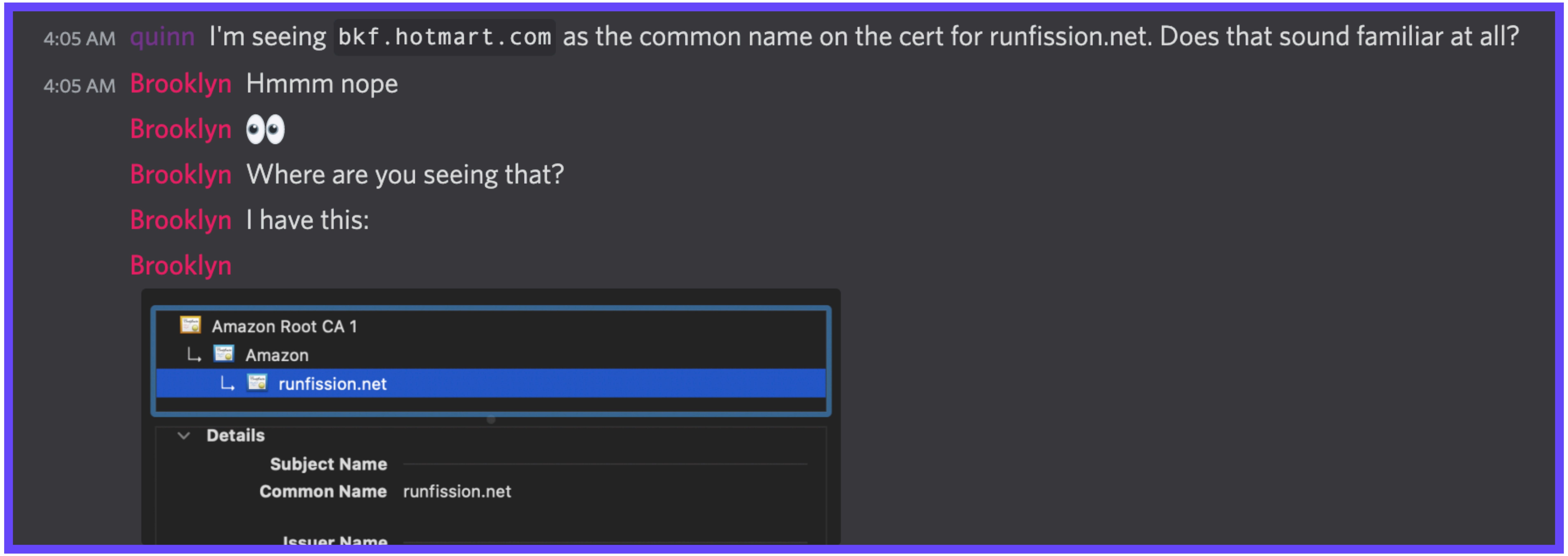


```
Firefox can’t establish a connection to the server at wss://runfission.net/v2/api      channel.ts:32:35
/user/link
/did:key:z13V3Sog2YaUKhdGCmgx9UZuW1o1ShFJYc6DvGYe7NTt689NoL3SJqWscsC7QZCYwrjoJh66q2FEi
dLvLLmGJ4TfidrXs7M3PJEaDHMLYxG4sEpjEnk3zV7DW5cwYEBez952SWsYy7jUnp7epa4vi6y57ETYxneBQ8V
9LGbwgztoSfipt6QciSnmKKsRmHNVZmrPAJryyb9v7RYnp6pkKGLAcCKyGQU51TNQvKpZqD5uaiFUz786BUi5Z
bJRfh9nZ9SEVeL2hhgTEZERY5P7yJdRmfHFQ5ZzQ5Nj6nJxybaBvfcsX2p9xoKuGVhjBimxP3paqSxH8HRutn5
fhmj7GY2w3kwm9tYTxpc6dwpv1if2qUnugQQvFVZda1GWnUyHPgs5dB7eqFuEDnDPeg6ee5n.

▶ Uncaught (in promise) Websocket channel could not be opened                          main.ts:100:23
```

The Mess We're (Still) In



```
Firefox can’t establish a connection to the server at wss://runfission.net/v2/api      channel.ts:32:35
/user/link
/did:key:z13V3Sog2YaUKhdGCmgx9UZuW1o1ShFJYc6DvGYe7NTt689NoL3SJqWscsC7QZCYwrjoJh66q2FEi
dLvLLmGJ4TfidrXs7M3PJEaDHMLYxG4sEpjEnk3zV7DW5cwYEBez952SWsYy7jUnp7epa4vi6y57ETYxneBQ8V
9LGbwgztoSfipt6QciSnmKKsRmHNVZmrPAJryyb9v7RYnp6pkKGLAcCKyGQU51TNQvKpZqD5uaiFUz786BUi5Z
bJRfh9nZ9SEVeL2hhgTEZERY5P7yJdRmfHFQ5ZzQ5Nj6nJxybaBvfcsX2p9xoKuGVhjBimxP3paqSxH8HRutn5
fhmj7GY2w3kwm9tYTxpc6dwpv1if2qUnugQQvFVZda1GWnUyHPgs5dB7eqFuEDnDPeg6ee5n.
Uncaught (in promise) Websocket channel could not be opened                            main.ts:100:23
```



4:05 AM quinn I'm seeing `bkf.hotmart.com` as the common name on the cert for runfission.net. Does that sound familiar at all?

4:05 AM Brooklyn Hmmm nope

Brooklyn 👀

Brooklyn Where are you seeing that?

Brooklyn I have this:

Brooklyn

Amazon Root CA 1
└ Amazon
  └ runfission.net

Details

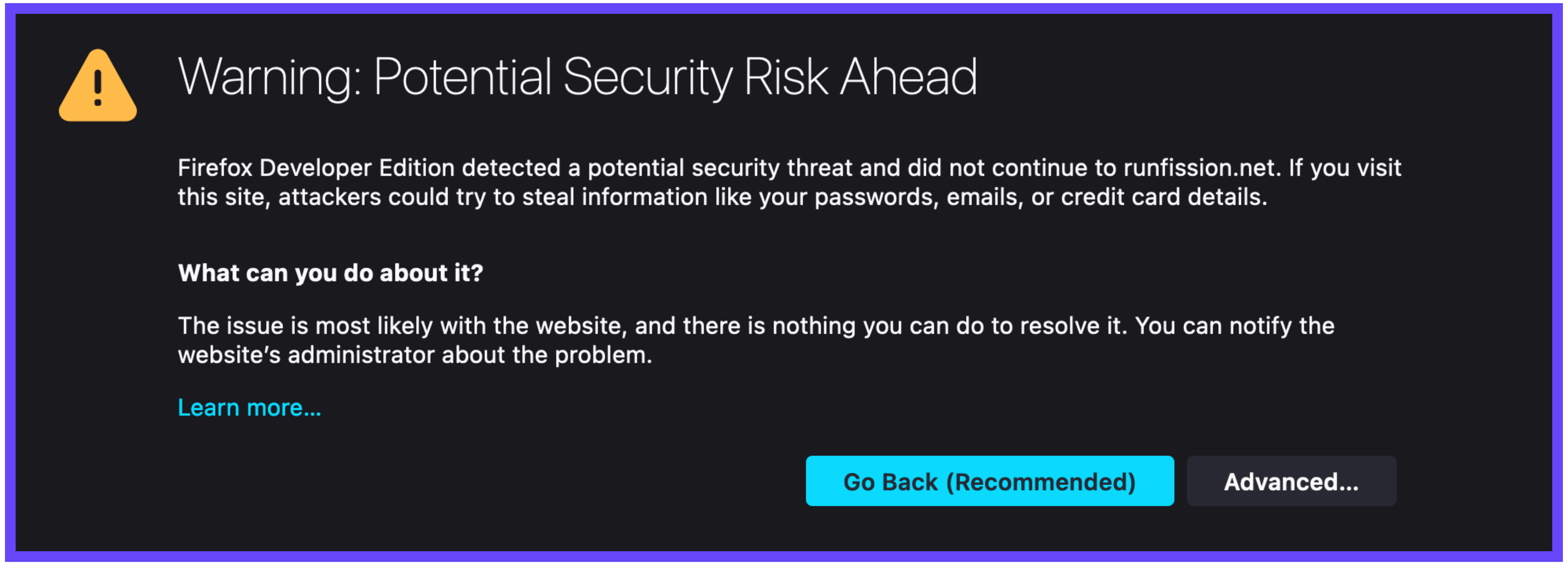
Subject Name

Common Name runfission.net

Issuer Name

The Mess We're (Still) In

Meanwhile...



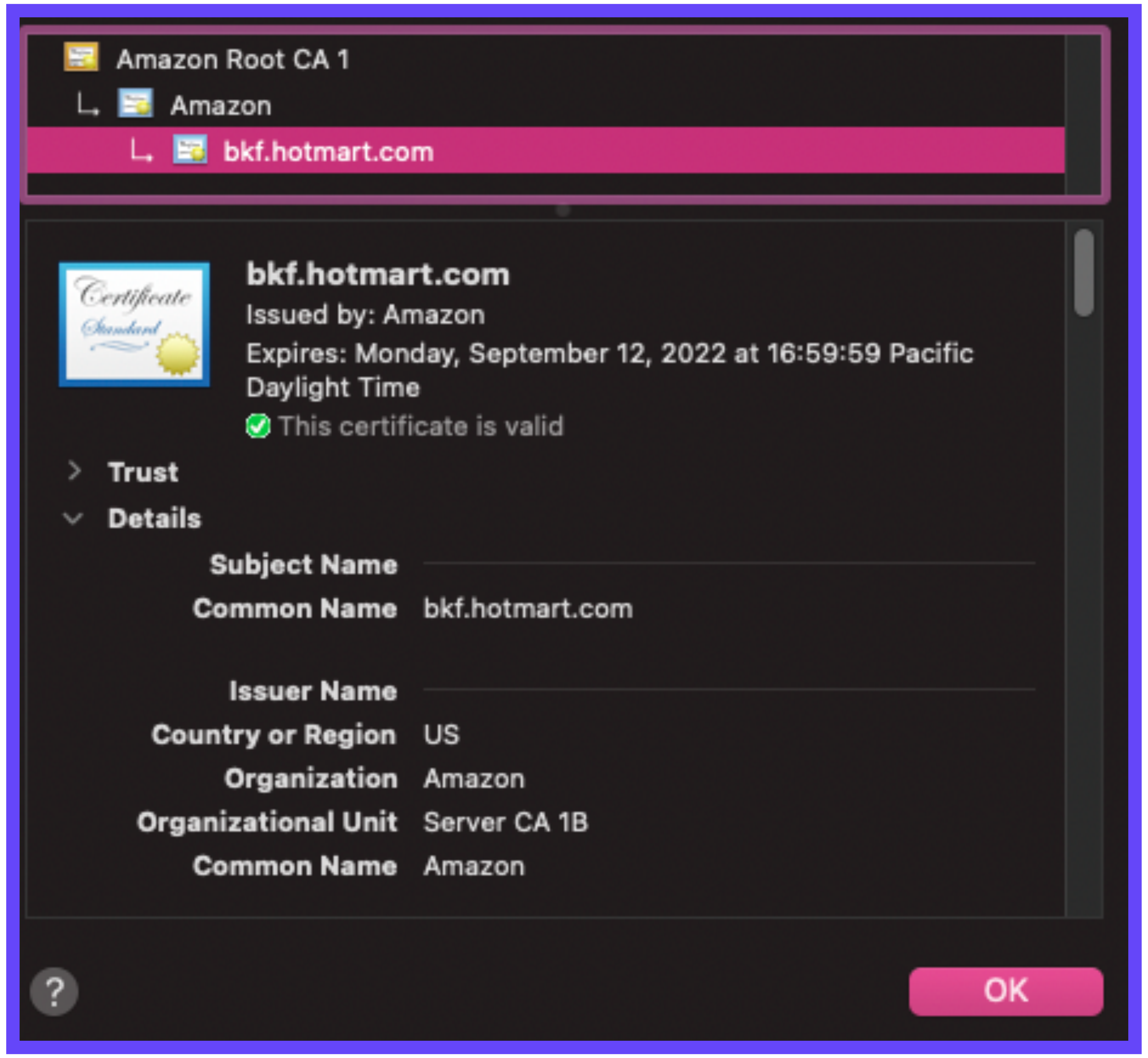
Warning: Potential Security Risk Ahead

Firefox Developer Edition detected a potential security threat and did not continue to runfission.net. If you visit this site, attackers could try to steal information like your passwords, emails, or credit card details.

What can you do about it?

The issue is most likely with the website, and there is nothing you can do to resolve it. You can notify the website's administrator about the problem.

Learn more...



Amazon Root CA 1
└ Amazon
 └ bkf.hotmart.com

bkf.hotmart.com
Issued by: Amazon
Expires: Monday, September 12, 2022 at 16:59:59 Pacific Daylight Time
✓ This certificate is valid

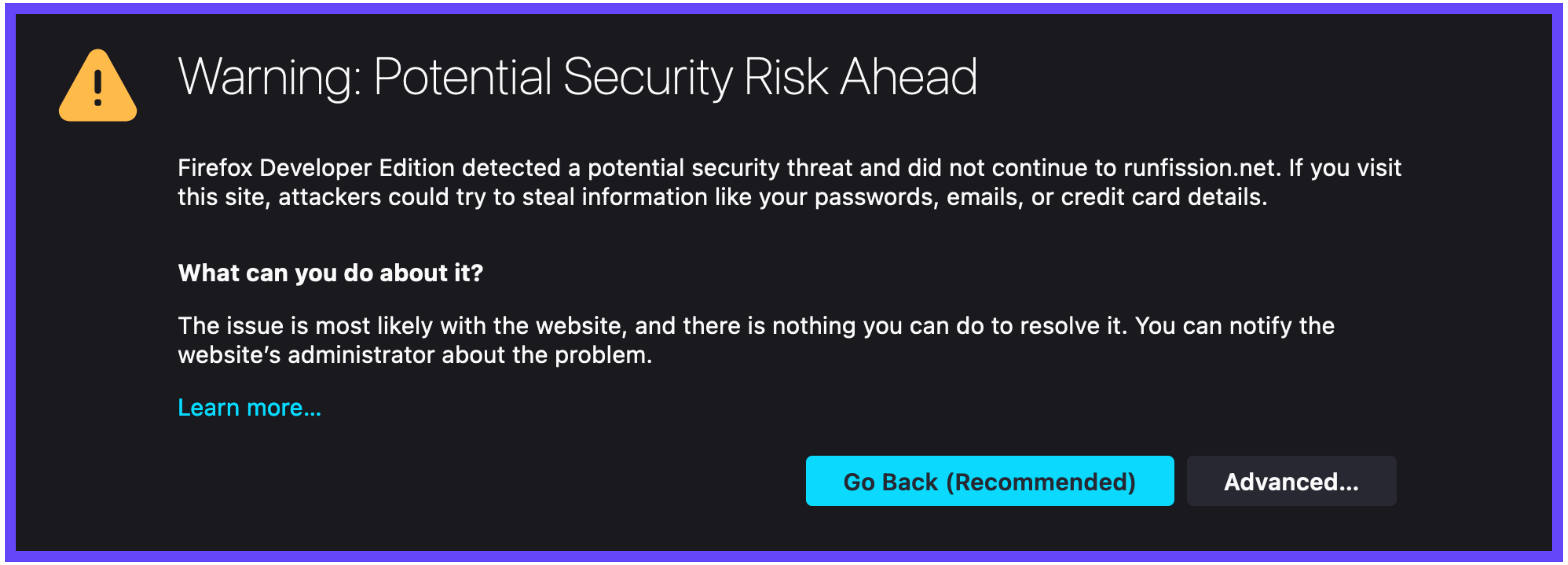
> Trust

⌄ Details

Subject Name	
Common Name	bkf.hotmart.com
Issuer Name	
Country or Region	US
Organization	Amazon
Organizational Unit	Server CA 1B
Common Name	Amazon

The Mess We're (Still) In

Meanwhile…



Warning: Potential Security Risk Ahead

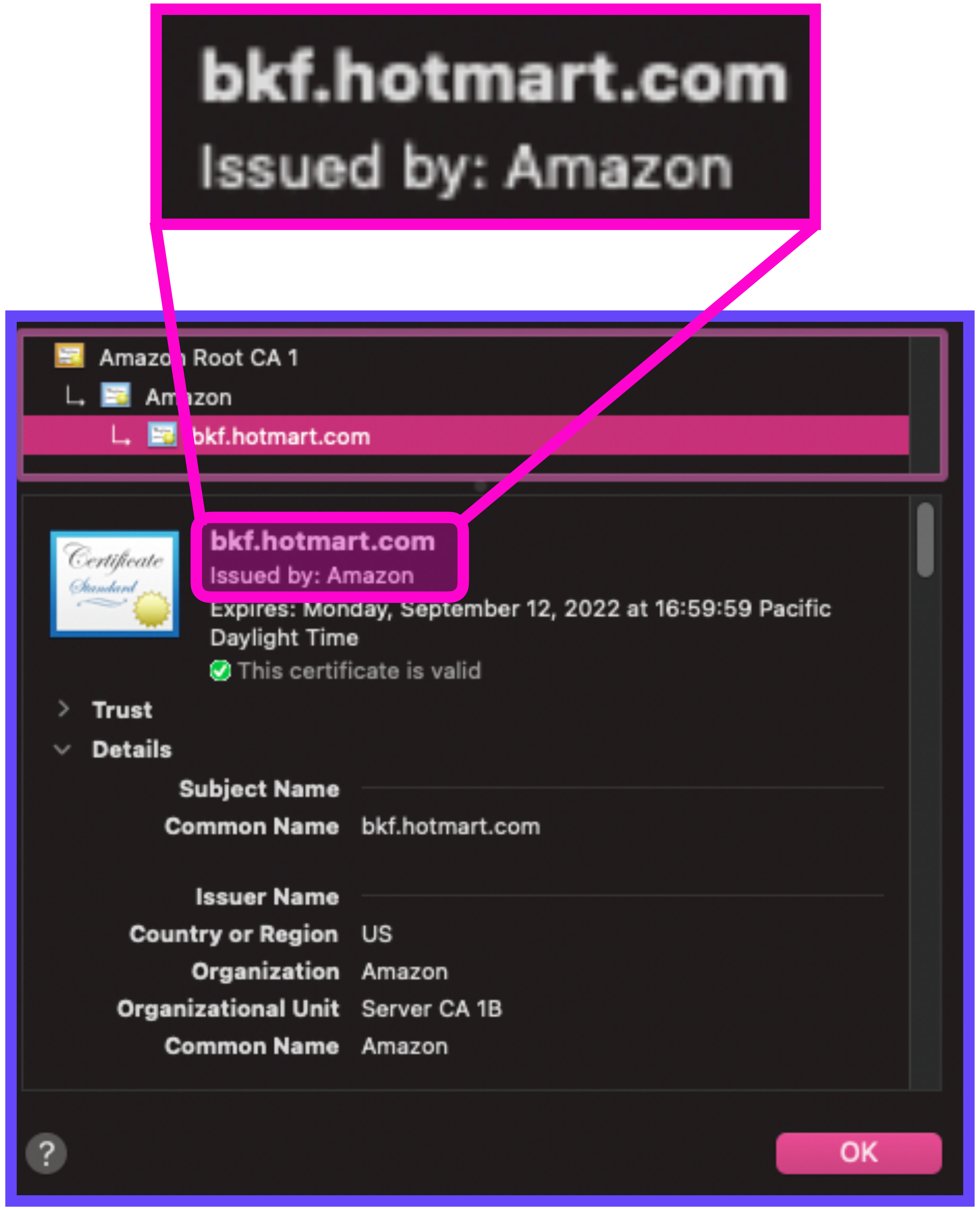
Firefox Developer Edition detected a potential security threat and did not continue to runfission.net. If you visit this site, attackers could try to steal information like your passwords, emails, or credit card details.

What can you do about it?

The issue is most likely with the website, and there is nothing you can do to resolve it. You can notify the website's administrator about the problem.

Learn more…

…who? 🤨



bkf.hotmart.com
Issued by: Amazon

Amazon Root CA 1
└ Amazon
  └ bkf.hotmart.com

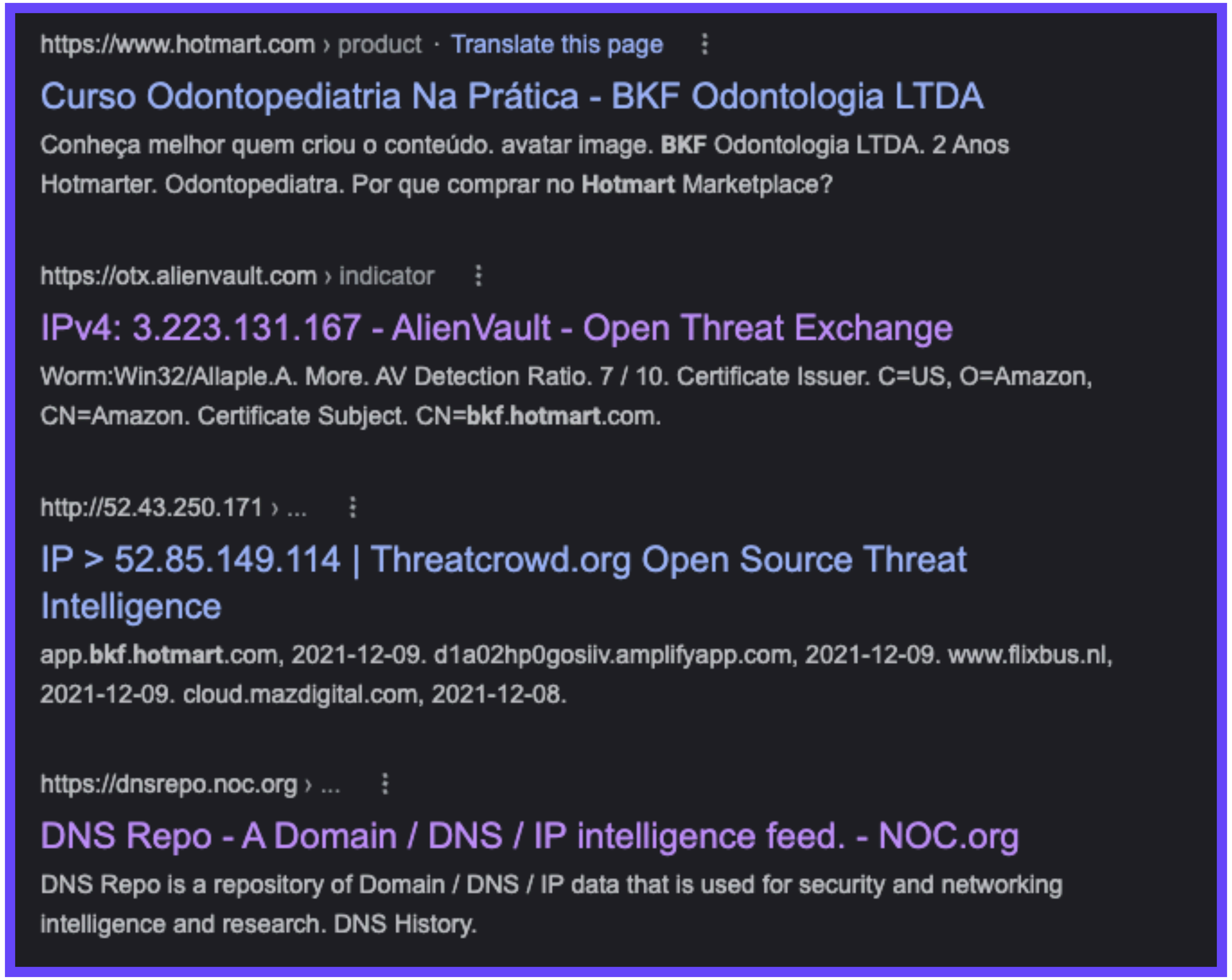
bkf.hotmart.com
Issued by: Amazon

Expires: Monday, September 12, 2022 at 16:59:59 Pacific Daylight Time
✓ This certificate is valid

› Trust
⌄ Details

Subject Name	
Common Name	bkf.hotmart.com
Issuer Name	
Country or Region	US
Organization	Amazon
Organizational Unit	Server CA 1B
Common Name	Amazon

The Mess We're (Still) In



https://www.hotmart.com › product · Translate this page ⋮

Curso Odontopediatria Na Prática - BKF Odontologia LTDA

Conheça melhor quem criou o conteúdo. avatar image. **BKF** Odontologia LTDA. 2 Anos Hotmarter. Odontopediatra. Por que comprar no **Hotmart** Marketplace?

https://otx.alienvault.com › indicator ⋮

IPv4: 3.223.131.167 - AlienVault - Open Threat Exchange

Worm:Win32/Allaple.A. More. AV Detection Ratio. 7 / 10. Certificate Issuer. C=US, O=Amazon, CN=Amazon. Certificate Subject. CN=**bkf.hotmart**.com.

http://52.43.250.171 › ... ⋮

IP > 52.85.149.114 | Threatcrowd.org Open Source Threat Intelligence

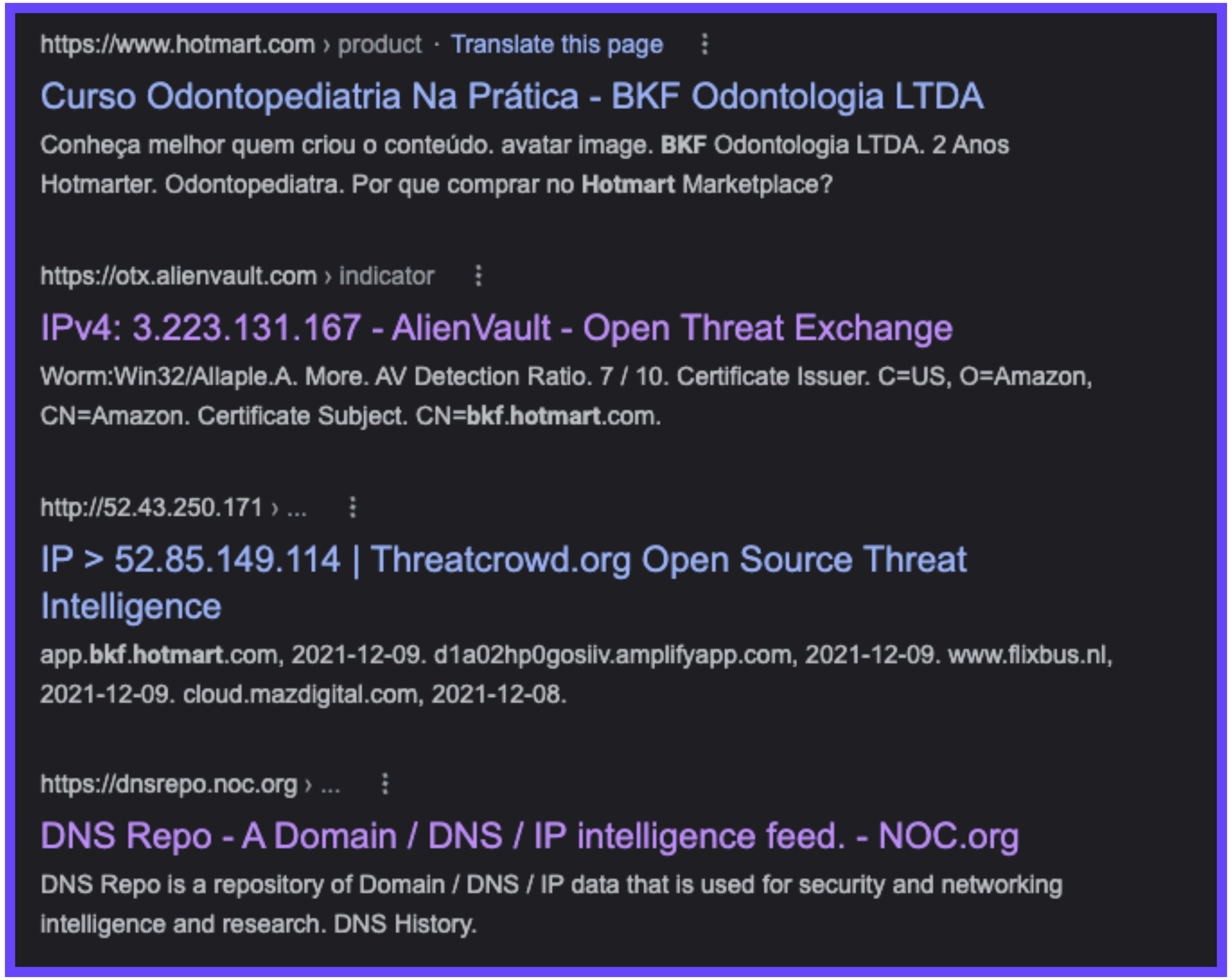
app.**bkf.hotmart**.com, 2021-12-09. d1a02hp0gosiiv.amplifyapp.com, 2021-12-09. www.flixbus.nl, 2021-12-09. cloud.mazdigital.com, 2021-12-08.

https://dnsrepo.noc.org › ... ⋮

DNS Repo - A Domain / DNS / IP intelligence feed. - NOC.org

DNS Repo is a repository of Domain / DNS / IP data that is used for security and networking intelligence and research. DNS History.

The Mess We're (Still) In



https://www.hotmart.com › product · Translate this page ⋮

Curso Odontopediatria Na Prática - BKF Odontologia LTDA

Conheça melhor quem criou o conteúdo. avatar image. **BKF** Odontologia LTDA. 2 Anos Hotmarter. Odontopediatra. Por que comprar no **Hotmart** Marketplace?

https://otx.alienvault.com › indicator ⋮

IPv4: 3.223.131.167 - AlienVault - Open Threat Exchange

Worm:Win32/Allaple.A. More. AV Detection Ratio. 7 / 10. Certificate Issuer. C=US, O=Amazon, CN=Amazon. Certificate Subject. CN=**bkf.hotmart**.com.

http://52.43.250.171 › ... ⋮

IP > 52.85.149.114 | Threatcrowd.org Open Source Threat Intelligence

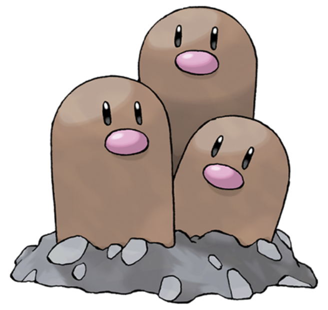
app.**bkf.hotmart**.com, 2021-12-09. d1a02hp0gosiiv.amplifyapp.com, 2021-12-09. www.flixbus.nl, 2021-12-09. cloud.mazdigital.com, 2021-12-08.

https://dnsrepo.noc.org › ... ⋮

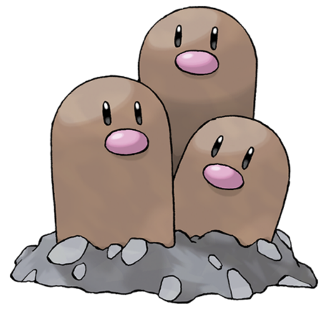
DNS Repo - A Domain / DNS / IP intelligence feed. - NOC.org

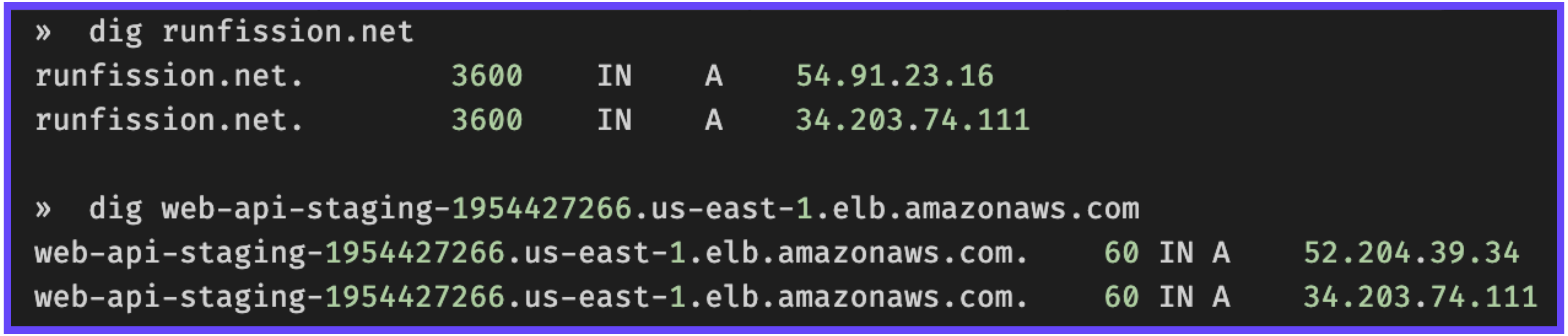
DNS Repo is a repository of Domain / DNS / IP data that is used for security and networking intelligence and research. DNS History.

The Mess We're (Still) In

Keep Calm and Dig In 🕳️⛏️ 

The Mess We're (Still) In

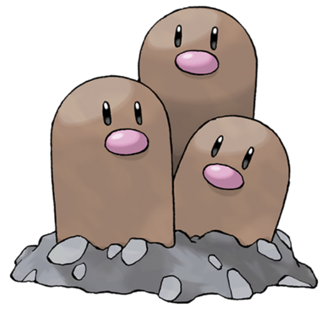
Keep Calm and Dig In 🕳️⛏️ 

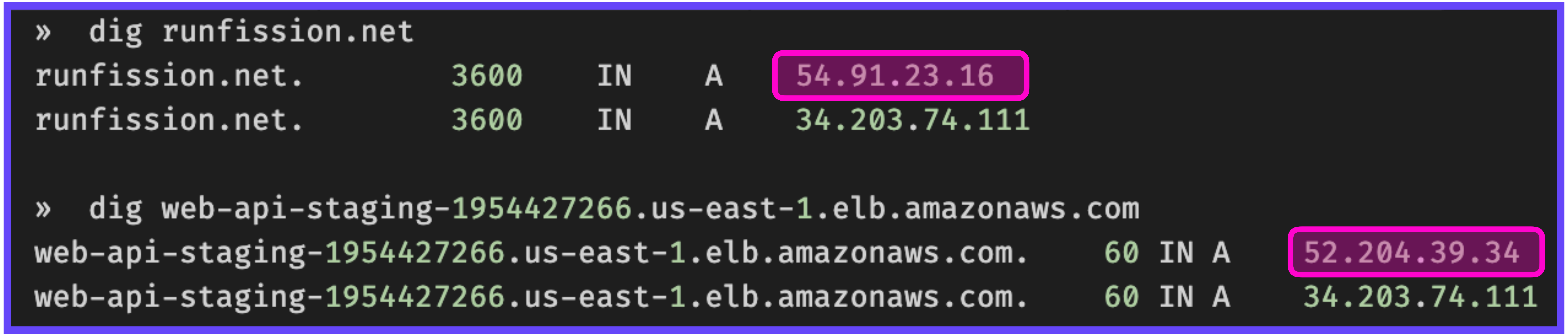


```
»  dig runfission.net
runfission.net.         3600    IN    A    54.91.23.16
runfission.net.         3600    IN    A    34.203.74.111

»  dig web-api-staging-1954427266.us-east-1.elb.amazonaws.com
web-api-staging-1954427266.us-east-1.elb.amazonaws.com.    60 IN A    52.204.39.34
web-api-staging-1954427266.us-east-1.elb.amazonaws.com.    60 IN A    34.203.74.111
```

The Mess We're (Still) In

Keep Calm and Dig In 🕳️⛏️

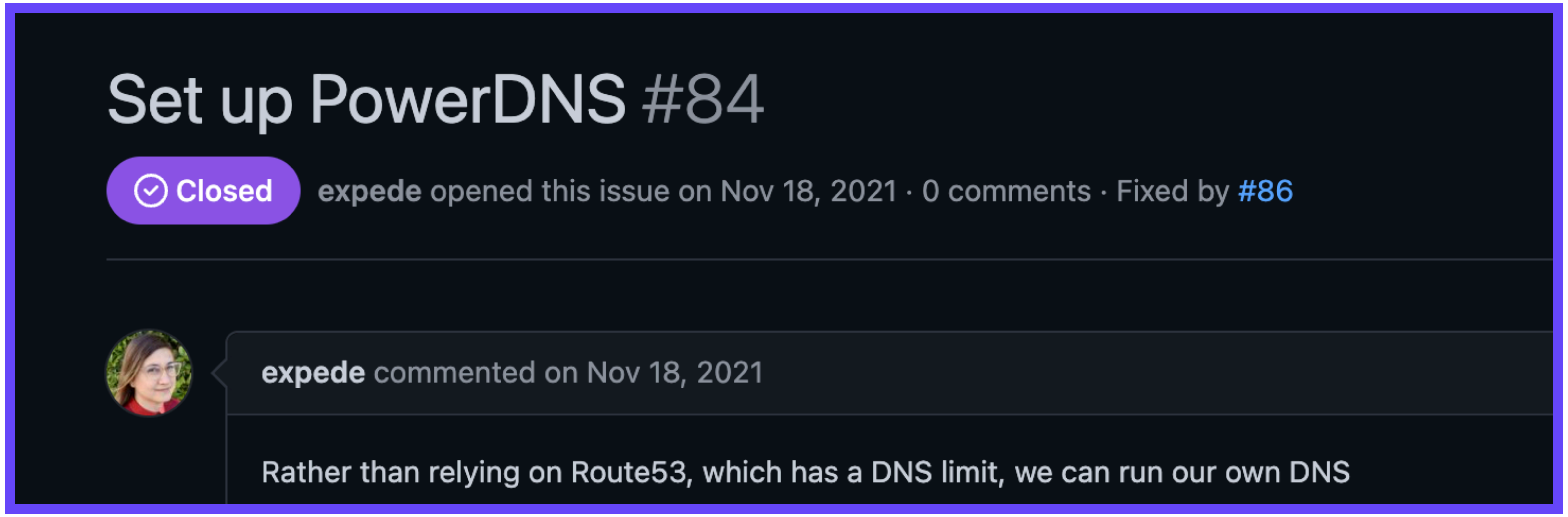




```
»  dig runfission.net
runfission.net.          3600    IN    A    54.91.23.16
runfission.net.          3600    IN    A    34.203.74.111

»  dig web-api-staging-1954427266.us-east-1.elb.amazonaws.com
web-api-staging-1954427266.us-east-1.elb.amazonaws.com.    60 IN A    52.204.39.34
web-api-staging-1954427266.us-east-1.elb.amazonaws.com.    60 IN A    34.203.74.111
```

The Mess We're (Still) In

The Mess We're (Still) In



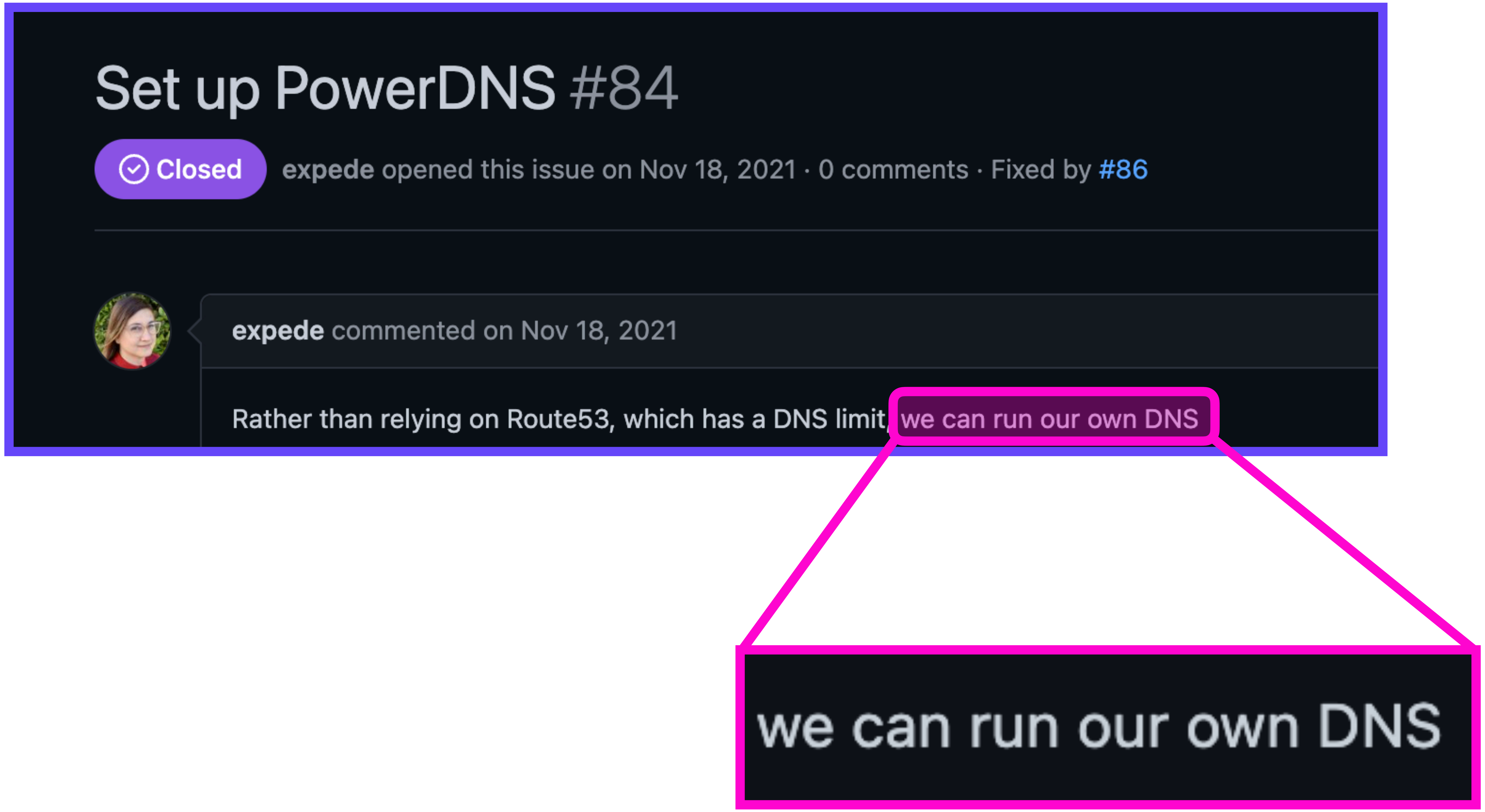
Set up PowerDNS #84

Closed expede opened this issue on Nov 18, 2021 · 0 comments · Fixed by #86

expede commented on Nov 18, 2021

Rather than relying on Route53, which has a DNS limit, we can run our own DNS

The Mess We're (Still) In



Set up PowerDNS #84

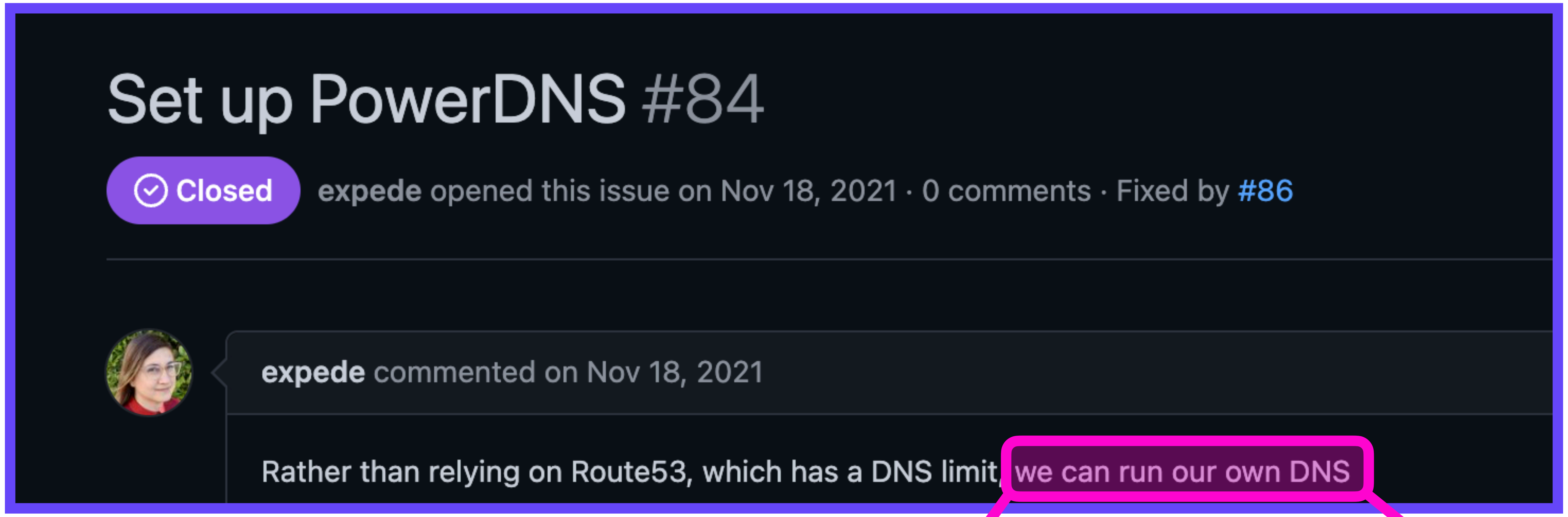
Closed expede opened this issue on Nov 18, 2021 · 0 comments · Fixed by #86

expede commented on Nov 18, 2021

Rather than relying on Route53, which has a DNS limit, we can run our own DNS

we can run our own DNS

The Mess We're (Still) In

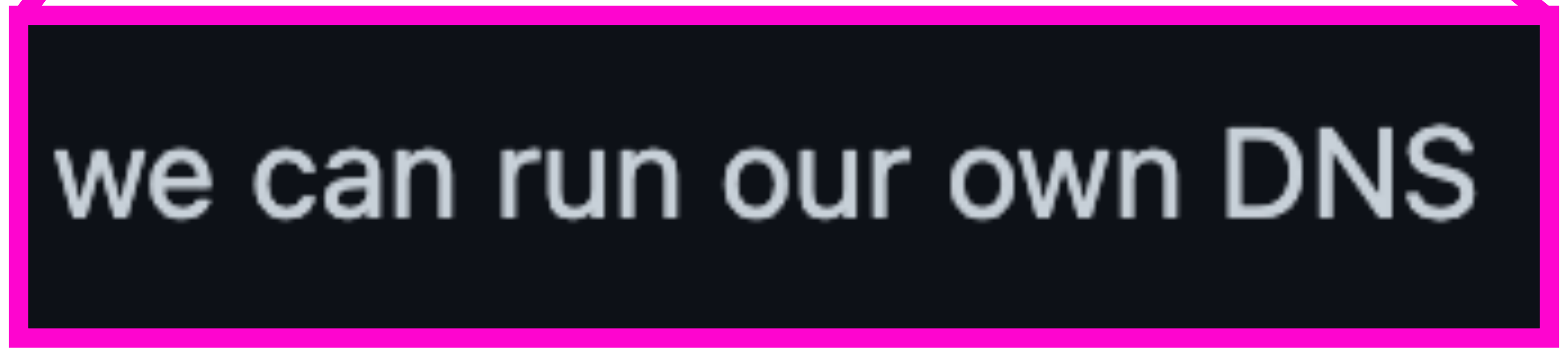


Set up PowerDNS #84

Closed expede opened this issue on Nov 18, 2021 · 0 comments · Fixed by #86

expede commented on Nov 18, 2021

Rather than relying on Route53, which has a DNS limit, we can run our own DNS



we can run our own DNS

Brooklyn Phew! Not hacked 😅

It's not DNS

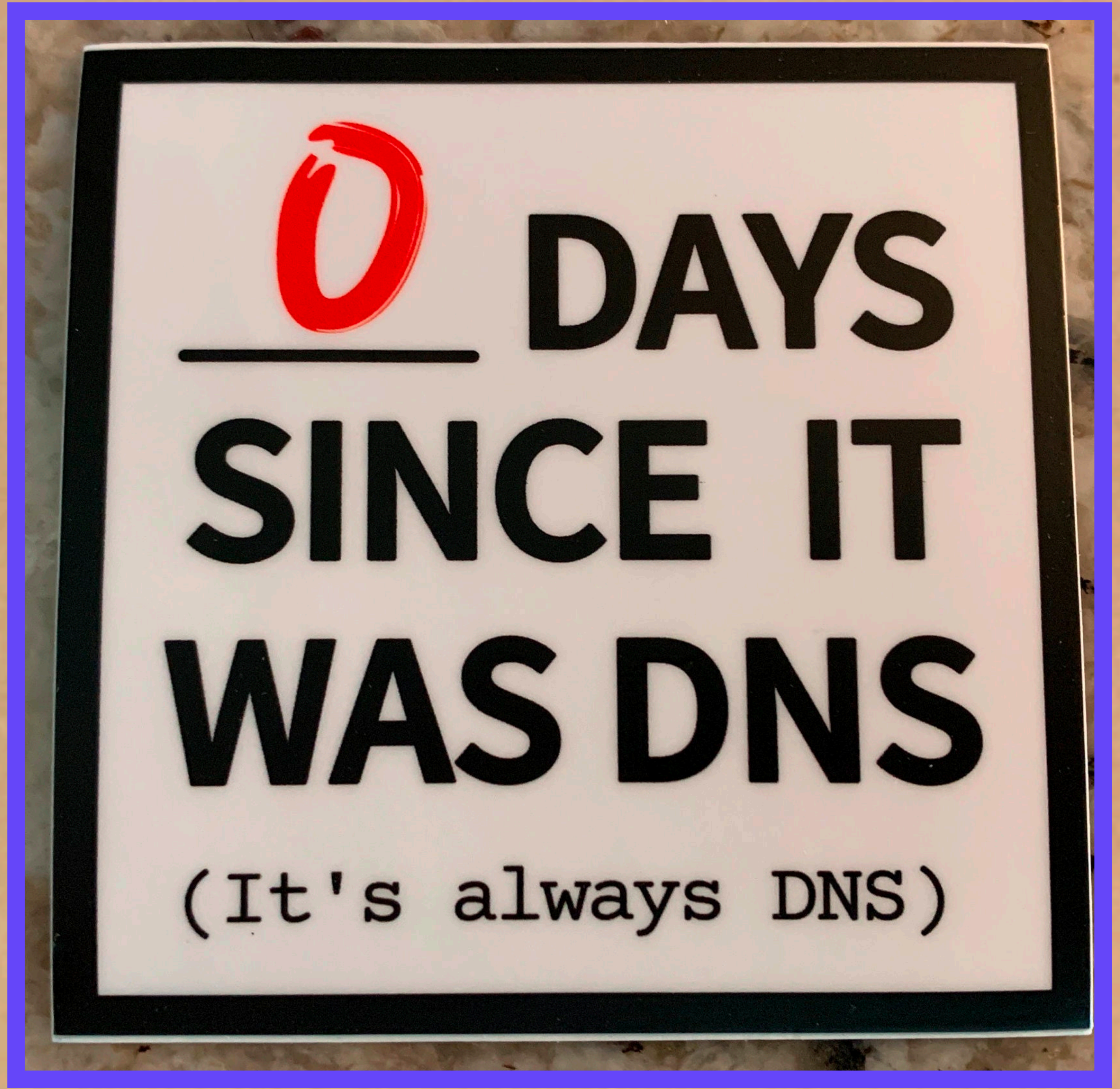
There's no way it's DNS

It was DNS

It's not DNS

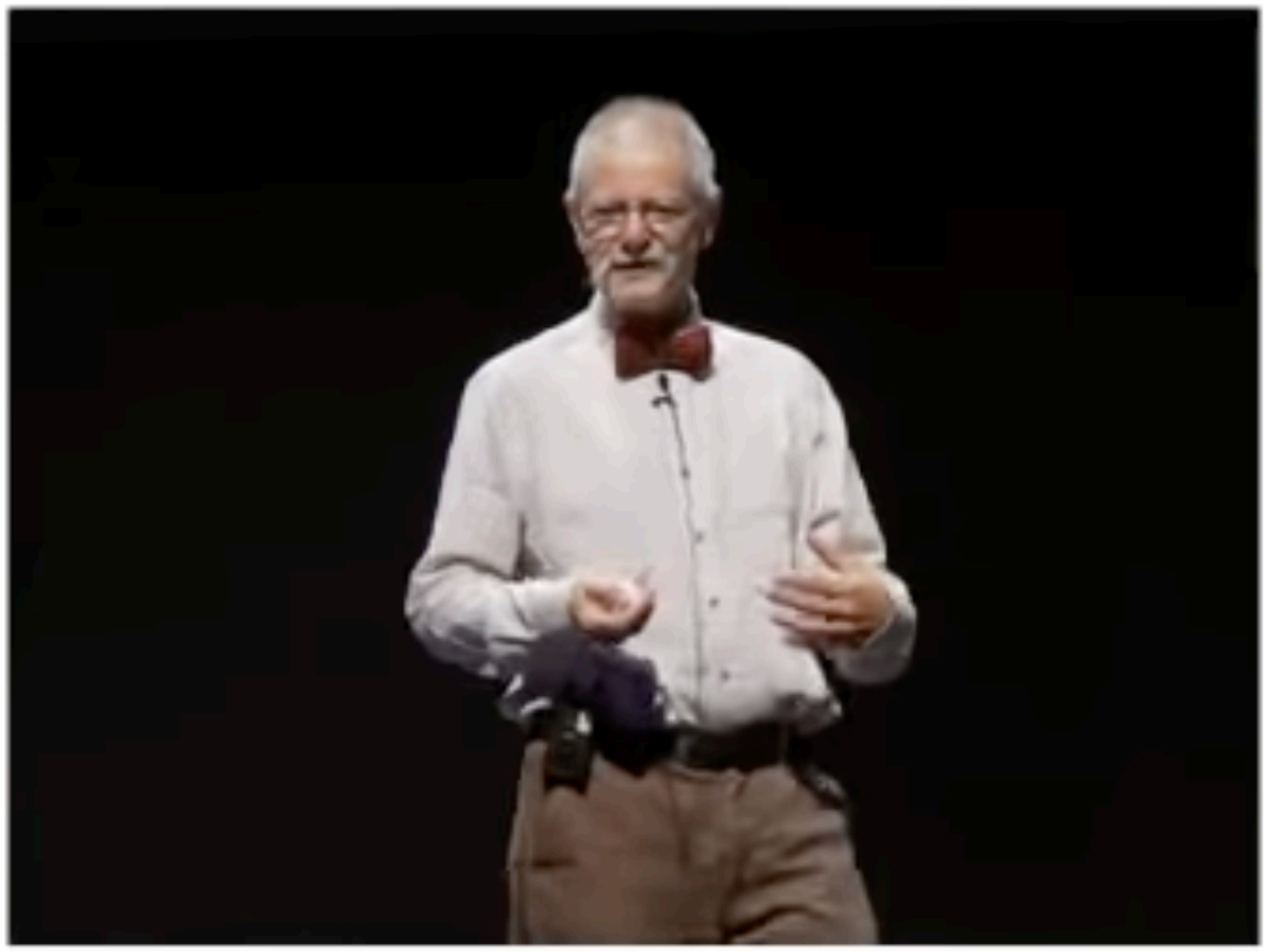
There's no way it's DNS

It was DNS



0 DAYS SINCE IT WAS DNS

(It's always DNS)



strangeloop

Sept 17-19, 2014 - St. Louis, MO
http://thestrangeloop.com

The Mess We're in

Joe Armstrong

ERICSSON

@FissionCodes

Brooklyn Zelenka

@expede

Quinn Wilton

@wilton_quinn

@FissionCodes

Brooklyn Zelenka

@expede

- CTO at Fission
- Distributed auth, data, compute, and discovery
- Author of Witchcraft, Algae, Exceptional, &c

Quinn Wilton

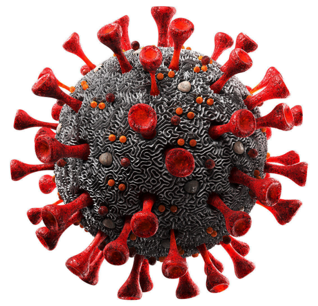
@wilton_quinn

- Applied Researcher at Fission
- Building a planetary scale database for local-first apps
- Contributed to Lumen, Gleam, Burrito, Witchcraft, &c

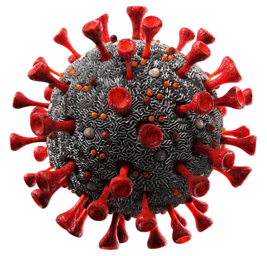
@FissionCodes

Brooklyn Zelenka

@expede

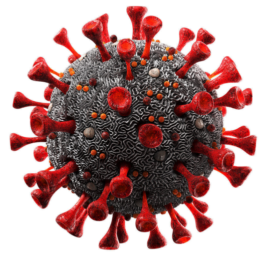


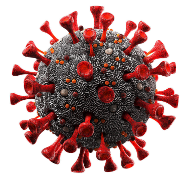
- CTO at Fission
- Distributed auth, data, compute, and discovery
- Author of Witchcraft, Algae, Exceptional, &c



Quinn Wilton

@wilton_quinn





- Applied Researcher at Fission
- Building a planetary scale database for local-first apps
- Contributed to Lumen, Gleam, Burrito, Witchcraft, &c

So Many

Problems

🪨🪐🌌

Problems 🪨🪐🌌

High Level

Problems 🪨🪐🌌

High Level

1. Massive State Space

Problems 🪨🪐🌌

High Level

1. Massive State Space
2. Place Oriented Programming

Problems 🪨🪐🌌

High Level

1. Massive State Space
2. Place Oriented Programming
3. Dependencies, Limited APIs, Inconsistency

Problems 🪨🪐🌌

State Space is Big

Problems 🪨🪐🌌

State Space is Big

Five 32-bit Numbers

$$(2^{32})^5 \approx 10^{48}$$

Problems 🪨🪐🌌

State Space is Big

Five 32-bit Numbers

$$(2^{32})^5 \approx 10^{48}$$

atoms on Earth

Problems 🪨🪐🌌

State Space is Big

Five 32-bit Numbers

$$(2^{32})^5 \approx 10^{48}$$

atoms on Earth

Six 32-bit Numbers

$$(2^{32})^6 \approx 10^{57}$$

Problems 🪨🪐🌌

State Space is Big

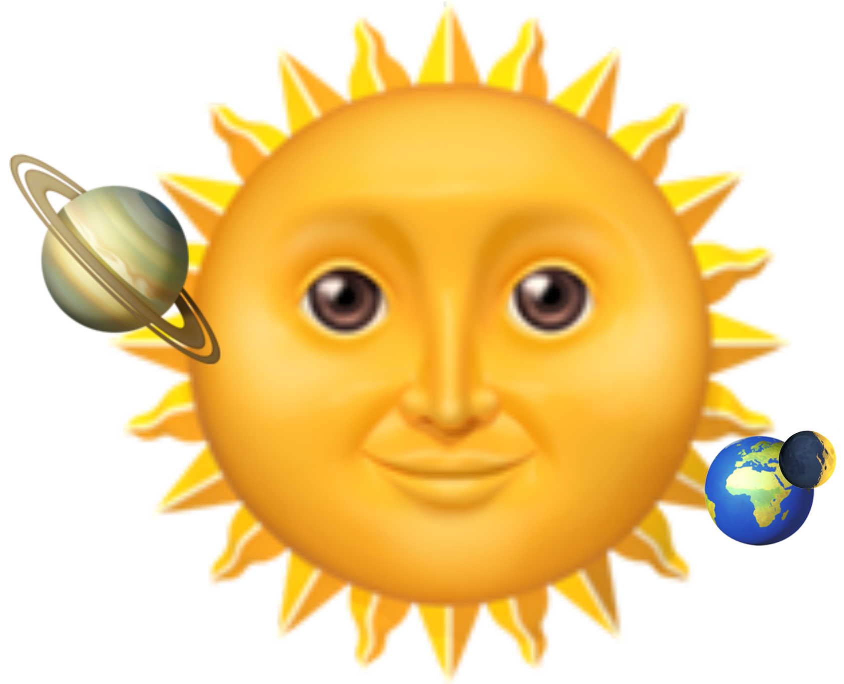
Five 32-bit Numbers

$$(2^{32})^5 \approx 10^{48}$$

atoms on Earth

Six 32-bit Numbers

$$(2^{32})^6 \approx 10^{57}$$



atoms in solar system

Problems 🪨🪐🌌

State Space is Big

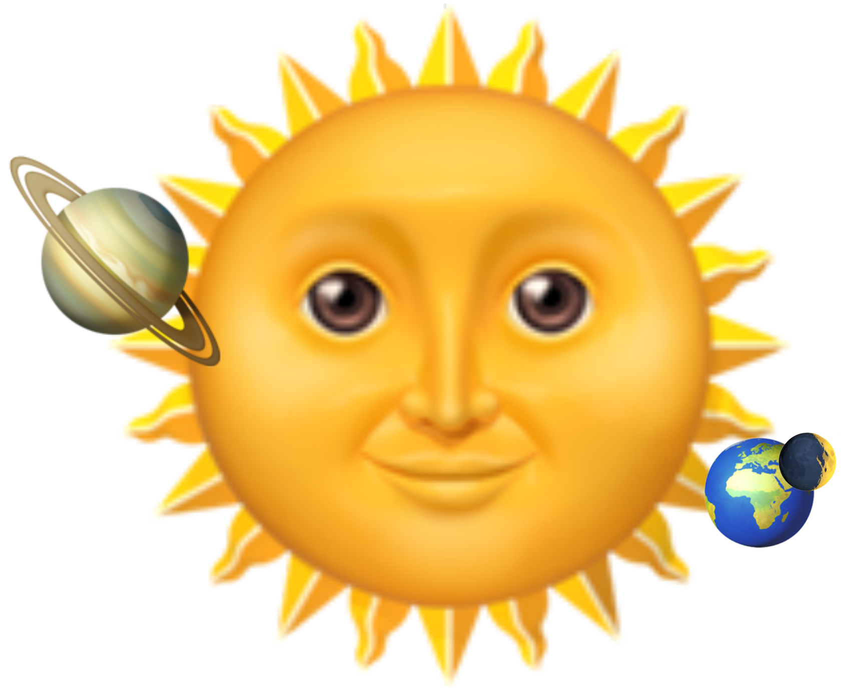
Five 32-bit Numbers

$(2^{32})^5 \approx 10^{48}$

atoms on Earth

Six 32-bit Numbers

$(2^{32})^6 \approx 10^{57}$



atoms in solar system

Single Receiver

$(2^{32+1})^6 \times 6! \approx 10^{62}$

Problems 🪨🪐🌌

State Space is Big

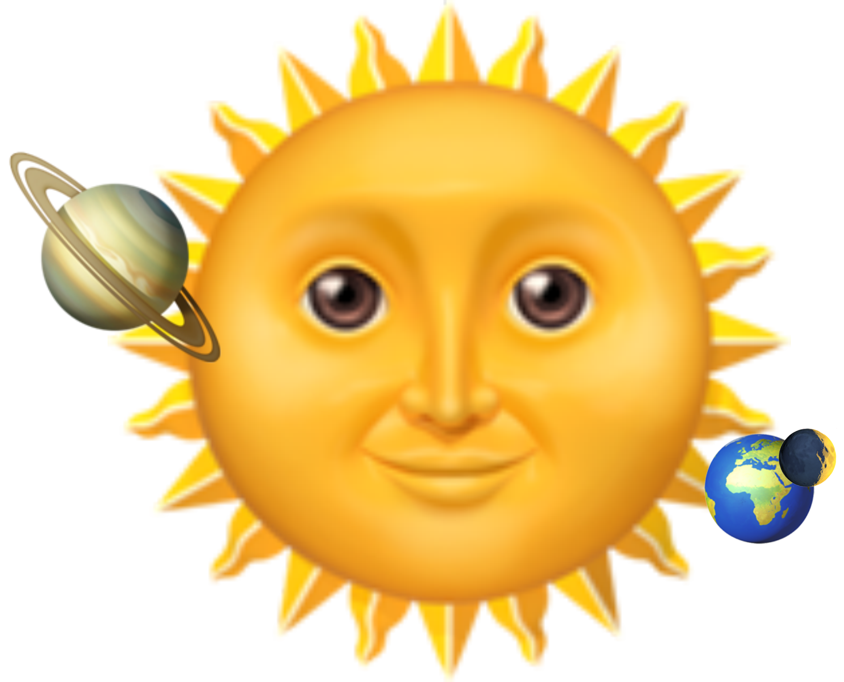
Five 32-bit Numbers

$$(2^{32})^5 \approx 10^{48}$$

atoms on Earth

Six 32-bit Numbers

$$(2^{32})^6 \approx 10^{57}$$



atoms in solar system

Single Receiver

$$(2^{32+1})^6 \times 6! \approx 10^{62}$$

atoms in Milky Way

Problems 🪨🪐🌌

State Space is Big

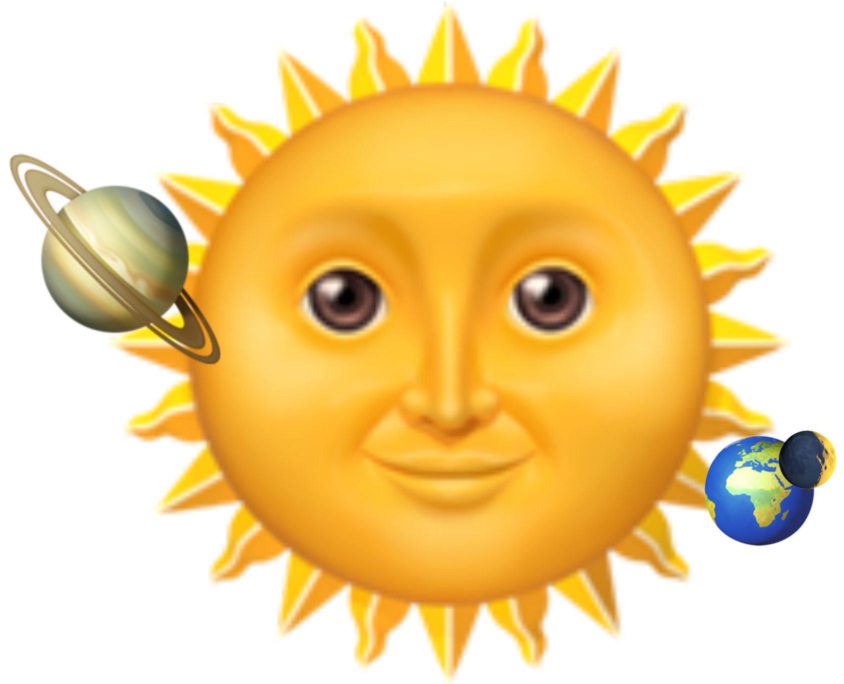
Five 32-bit Numbers

$(2^{32})^5 \approx 10^{48}$

atoms on Earth

Six 32-bit Numbers

$(2^{32})^6 \approx 10^{57}$



atoms in solar system

Single Receiver

$(2^{32+1})^6 \times 6! \approx 10^{62}$

atoms in Milky Way

Two Receivers

$((2^{32+1})^6 \times 6!)^2 \approx 10^{124}$

Problems 🪨🪐🌌

State Space is Big

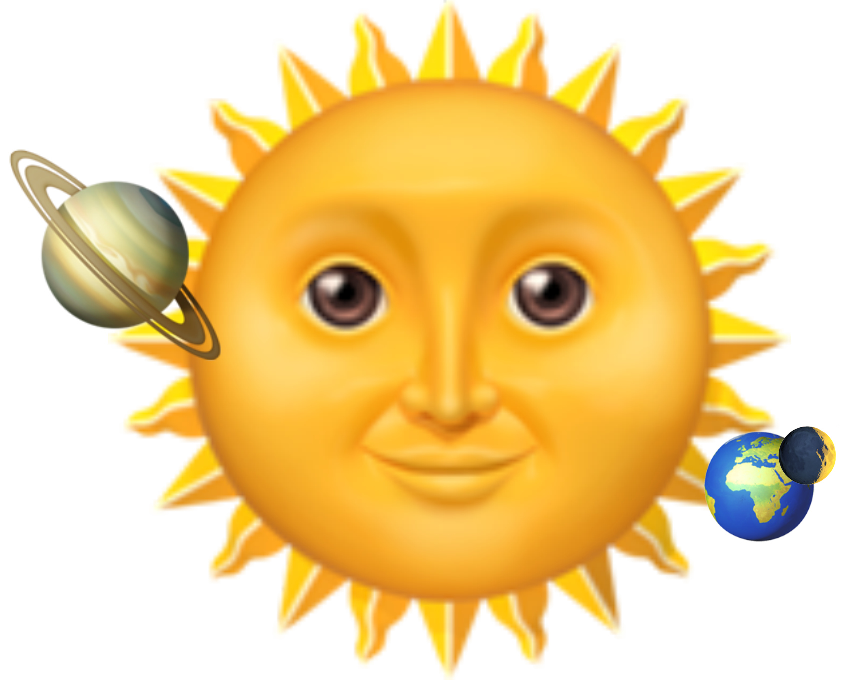
Five 32-bit Numbers

$(2^{32})^5 \approx 10^{48}$

atoms on Earth

Six 32-bit Numbers

$(2^{32})^6 \approx 10^{57}$



atoms in solar system

Single Receiver

$(2^{32+1})^6 \times 6! \approx 10^{62}$

atoms in Milky Way

Two Receivers

$((2^{32+1})^6 \times 6!)^2 \approx 10^{124}$

More than observable universe

Problems 🪨🪐🌌

Problems 🪨🪐🌌

Distributed systems introduce significant nondeterminism to our programs. Sources of non-determinism include unsynchronized parallelism, unreliable components, and networks with unpredictable delays. As a result, a distributed program can ***exhibit a large space of possible behaviors*** on a given input.

— Hellerstein & Alvaro, Keeping CALM: When Distributed Consistency is Easy

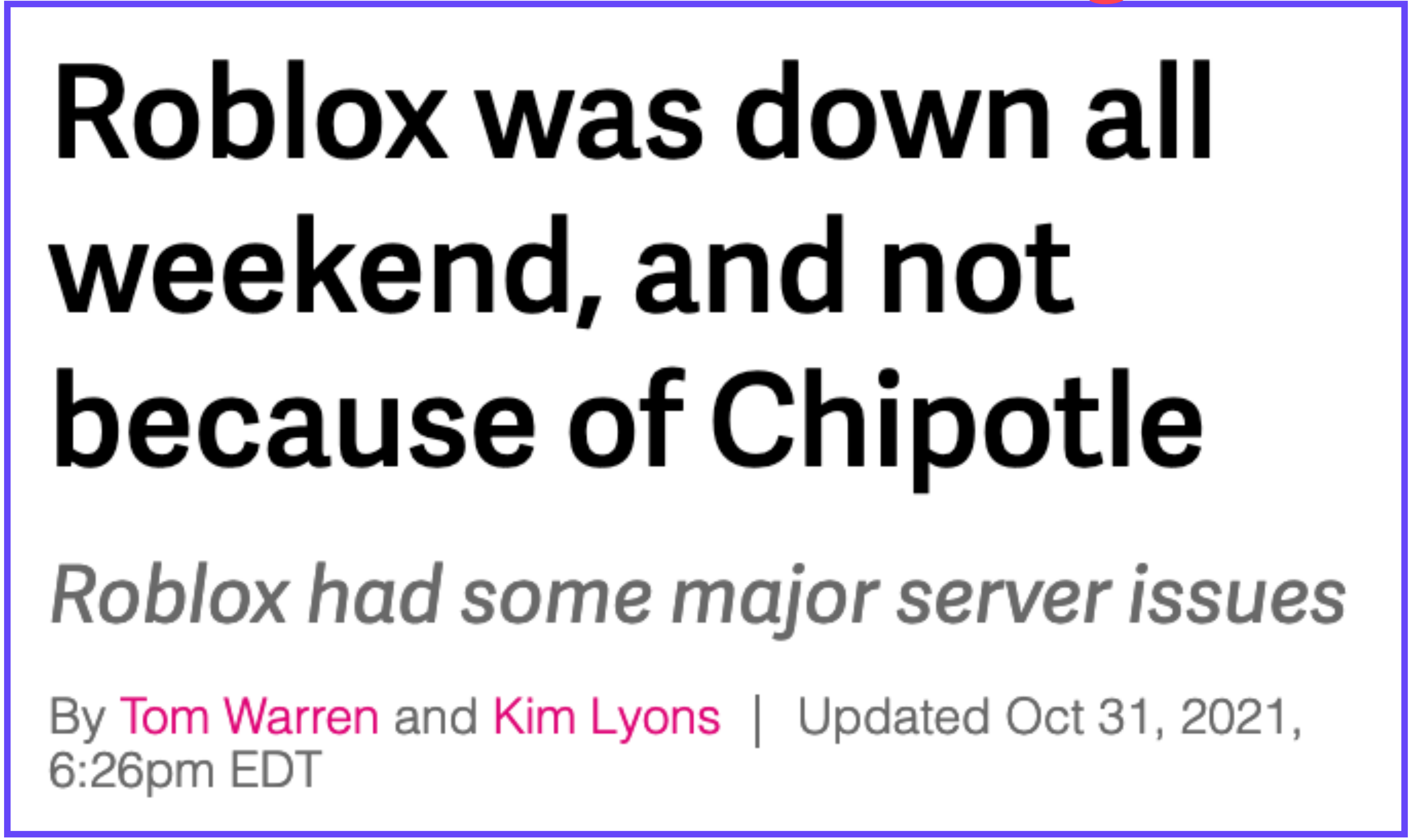
Problems 🪨🪐🌌

The Great 73-Hour Roblox Outage of 2021

https://blog.roblox.com/2022/01/roblox-return-to-service-10-28-10-31-2021/
https://www.theverge.com/2021/10/30/22754107/roblox-down-outage-chipotle-server-issues-status

Problems 🪨🪐🌌

The Great 73-Hour Roblox Outage of 2021



Roblox was down all weekend, and not because of Chipotle

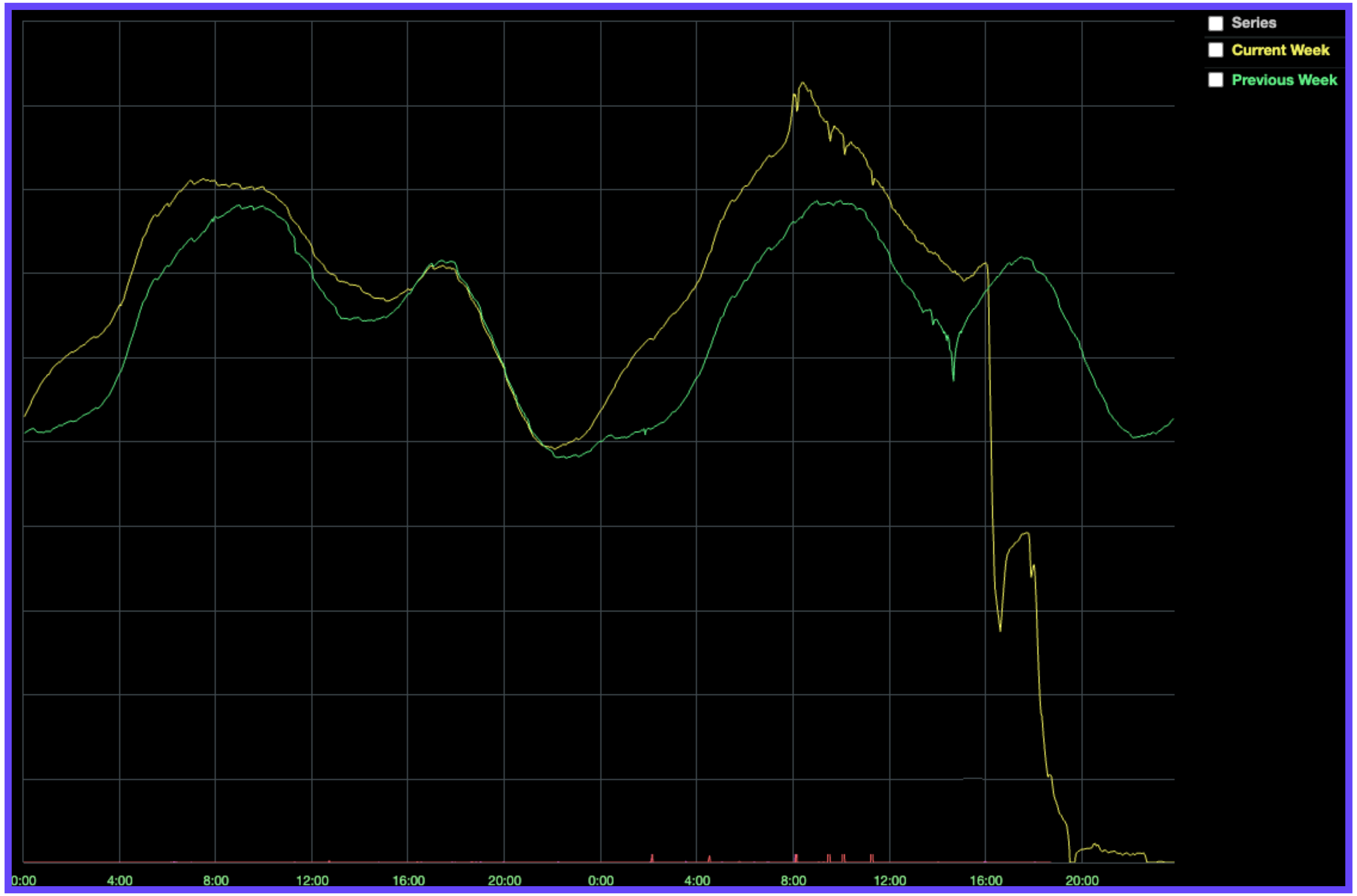
Roblox had some major server issues

By Tom Warren and Kim Lyons | Updated Oct 31, 2021, 6:26pm EDT

https://blog.roblox.com/2022/01/roblox-return-to-service-10-28-10-31-2021/
https://www.theverge.com/2021/10/30/22754107/roblox-down-outage-chipotle-server-issues-status

Problems 🪨🪐🌌

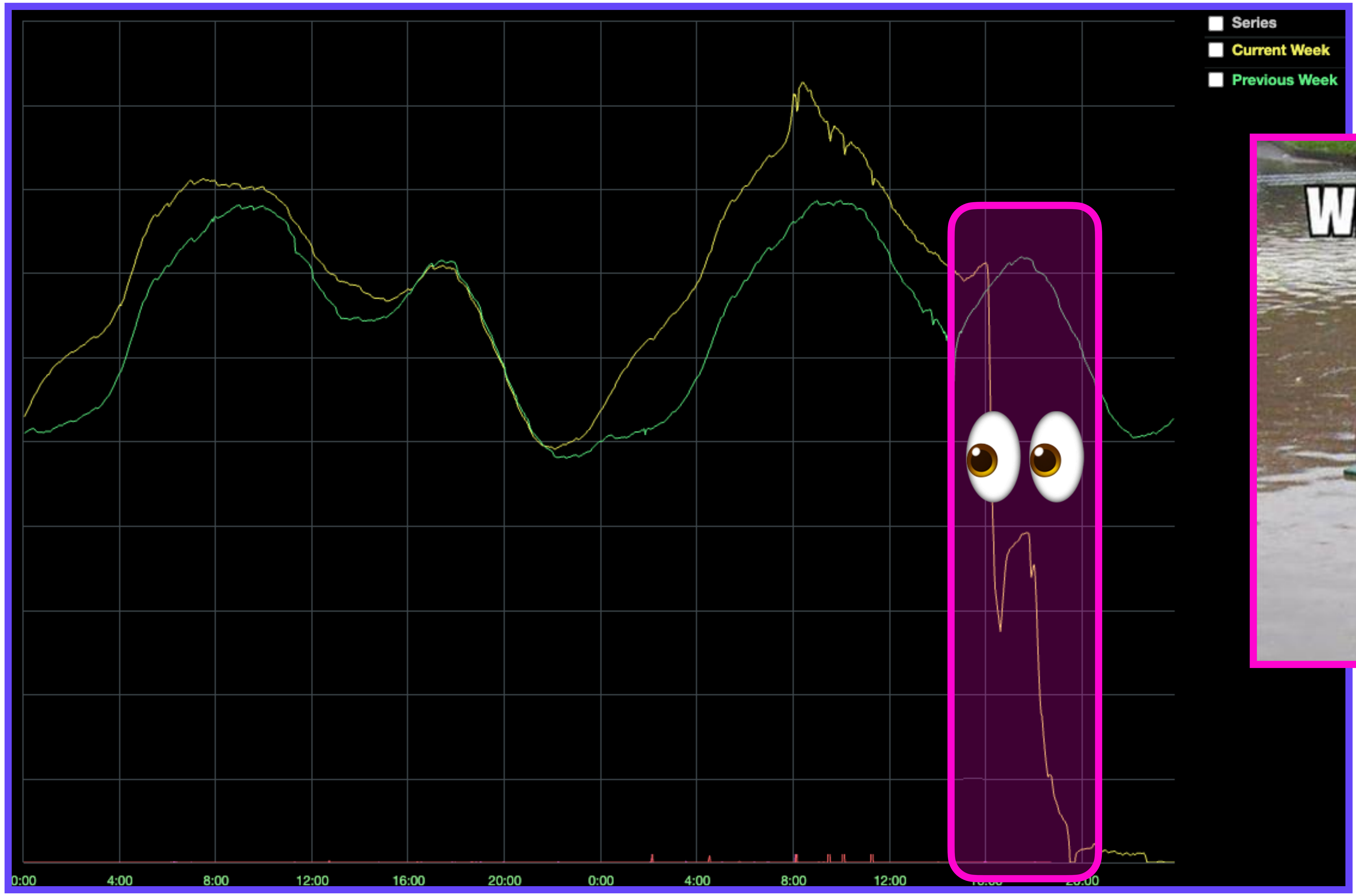
The Great 73-Hour Roblox Outage of 2021



https://blog.roblox.com/2022/01/roblox-return-to-service-10-28-10-31-2021/
https://www.theverge.com/2021/10/30/22754107/roblox-down-outage-chipotle-server-issues-status

Problems 🪨🪐🌌

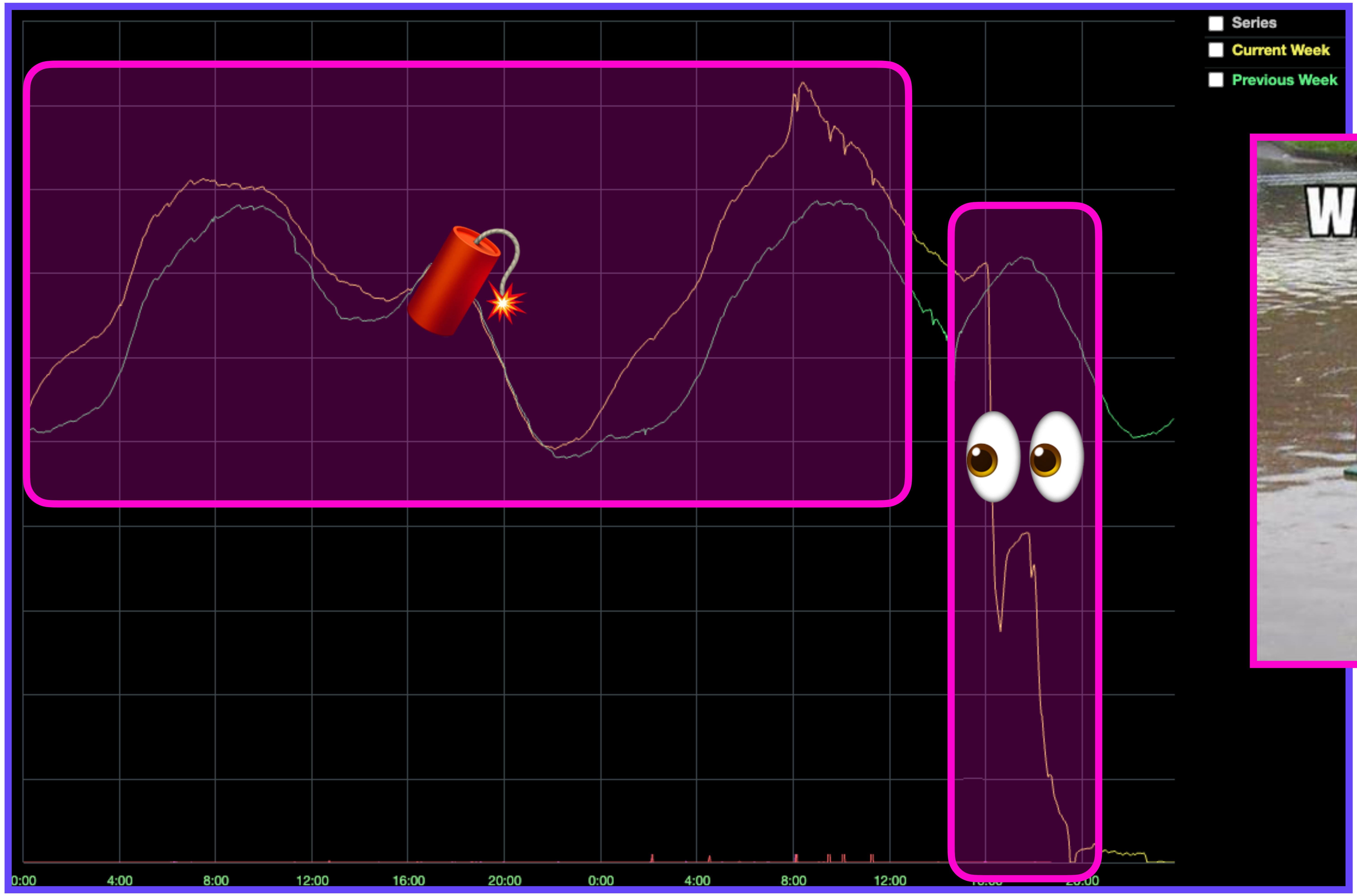
The Great 73-Hour Roblox Outage of 2021



https://blog.roblox.com/2022/01/roblox-return-to-service-10-28-10-31-2021/
https://www.theverge.com/2021/10/30/22754107/roblox-down-outage-chipotle-server-issues-status

Problems 🪨🪐🌌

The Great 73-Hour Roblox Outage of 2021



https://blog.roblox.com/2022/01/roblox-return-to-service-10-28-10-31-2021/
https://www.theverge.com/2021/10/30/22754107/roblox-down-outage-chipotle-server-issues-status

Problems 🪨🪐🌌

And Yet...

Problems 🪨🪐🌌

And Yet...

These metastable failures have caused ***widespread outages*** at large internet companies, lasting from minutes to hours. ***Paradoxically***, the root cause of these failures is ***often features that improve the efficiency or reliability of the system.***

– Bronson et al, Metastable Failures in Distributed Systems

Problems 🪨🪐🌌

Metastable Mechanism

Problems 🪨🪐🌌

Metastable Mechanism

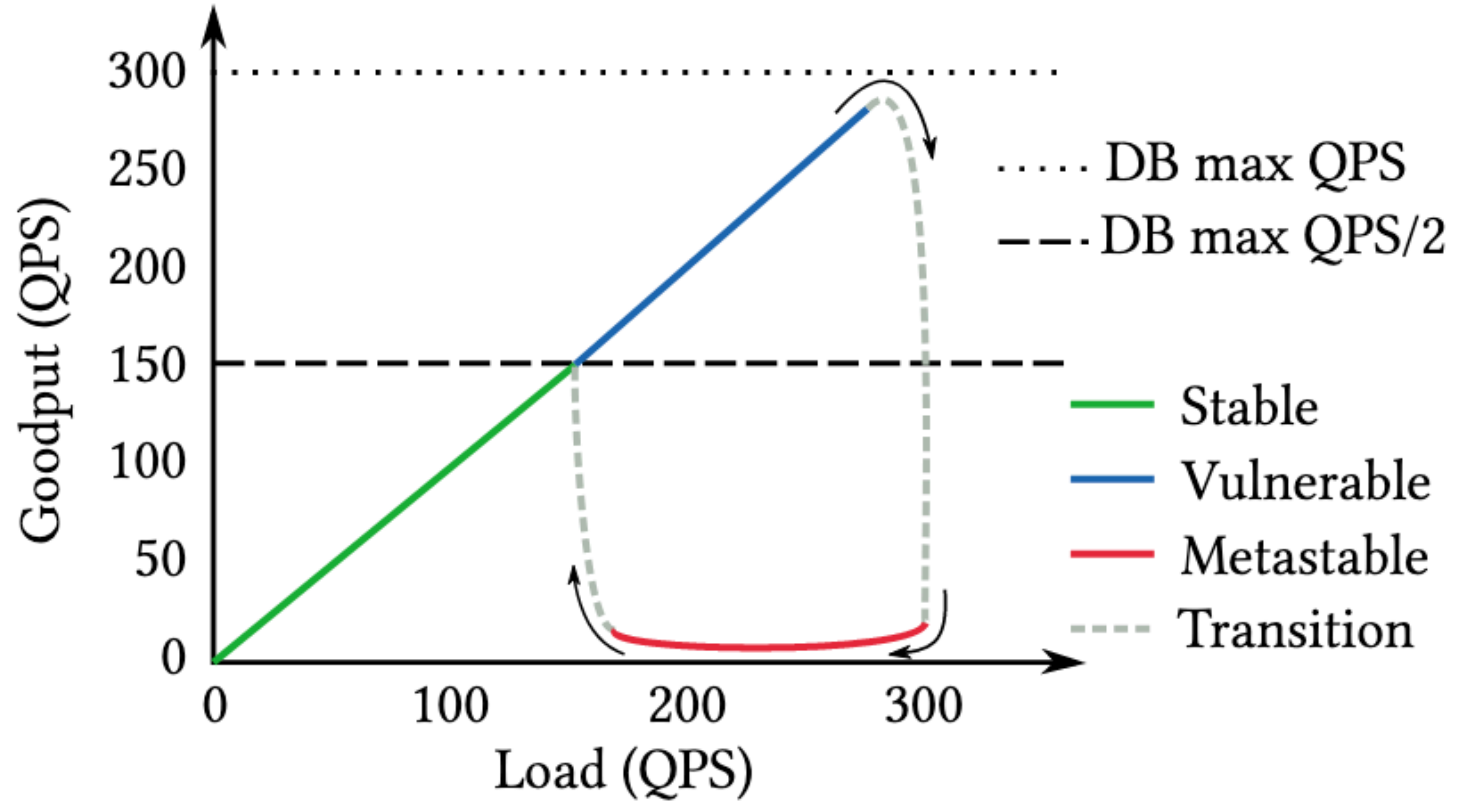


Figure 2: Goodput of an idealized web application with a database backend in stable, vulnerable, and metastable states.

Problems 🪨🪐🌌

Metastable Mechanism

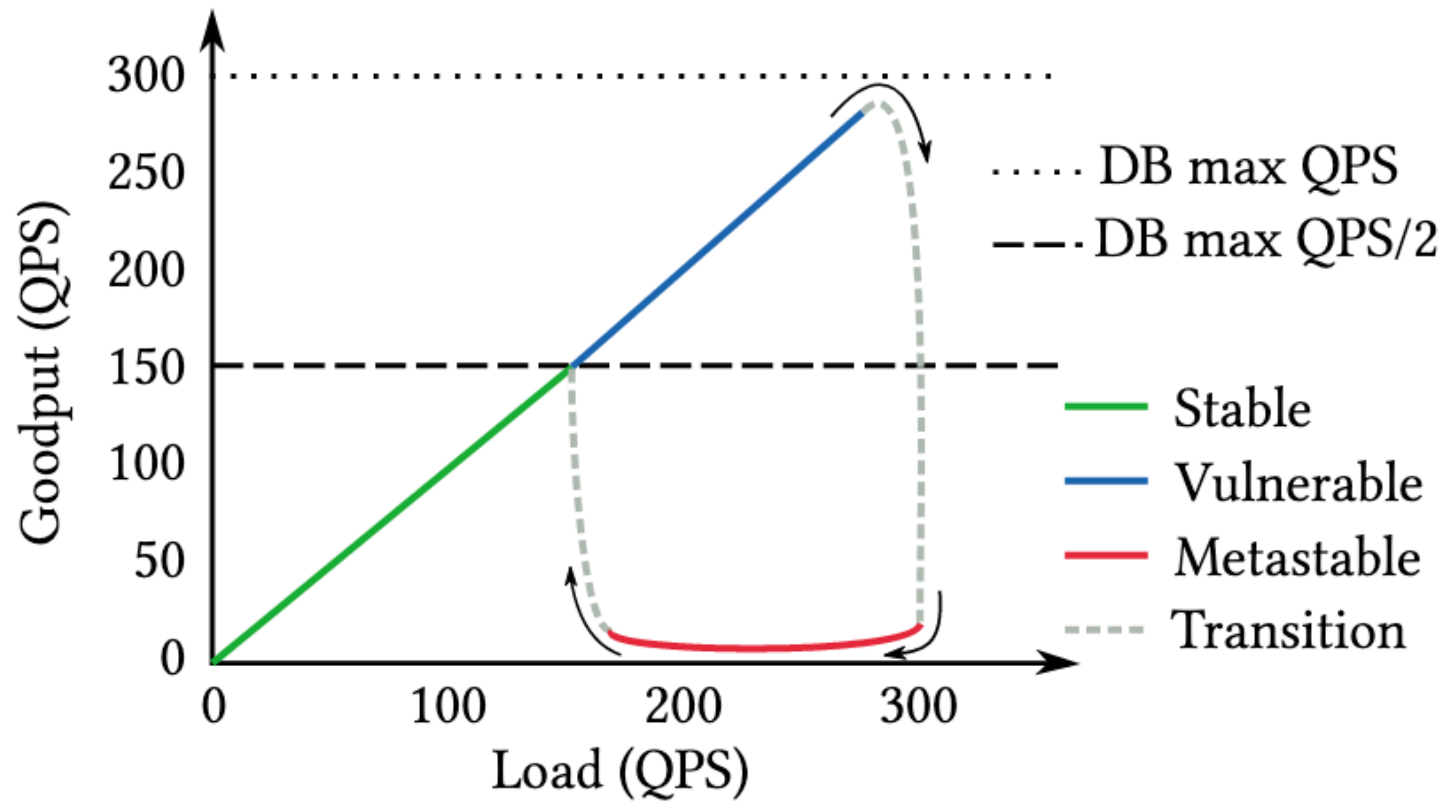
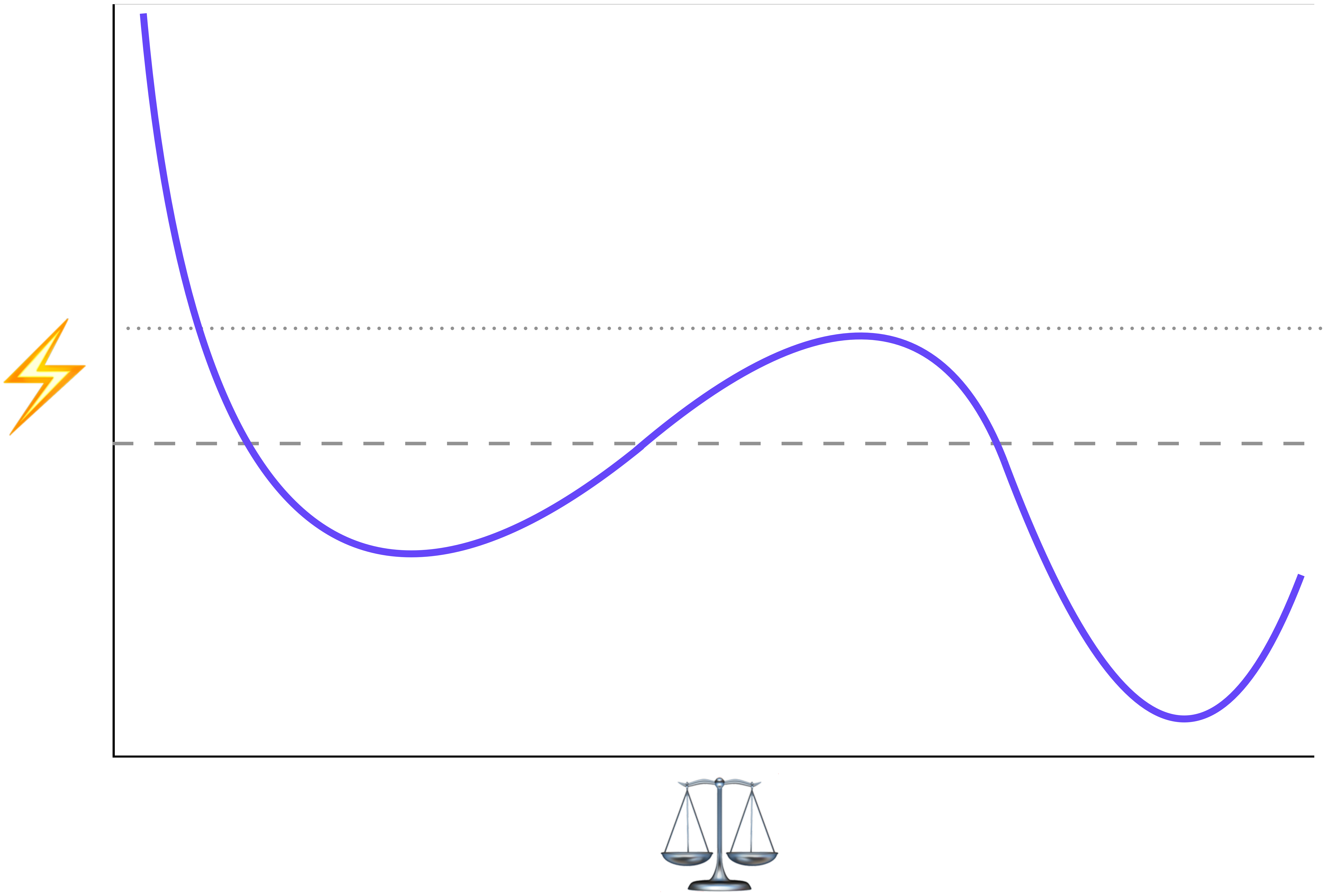


Figure 2: Goodput of an idealized web application with a database backend in stable, vulnerable, and metastable states.

Problems 🪨🪐🌌

Metastable Mechanism



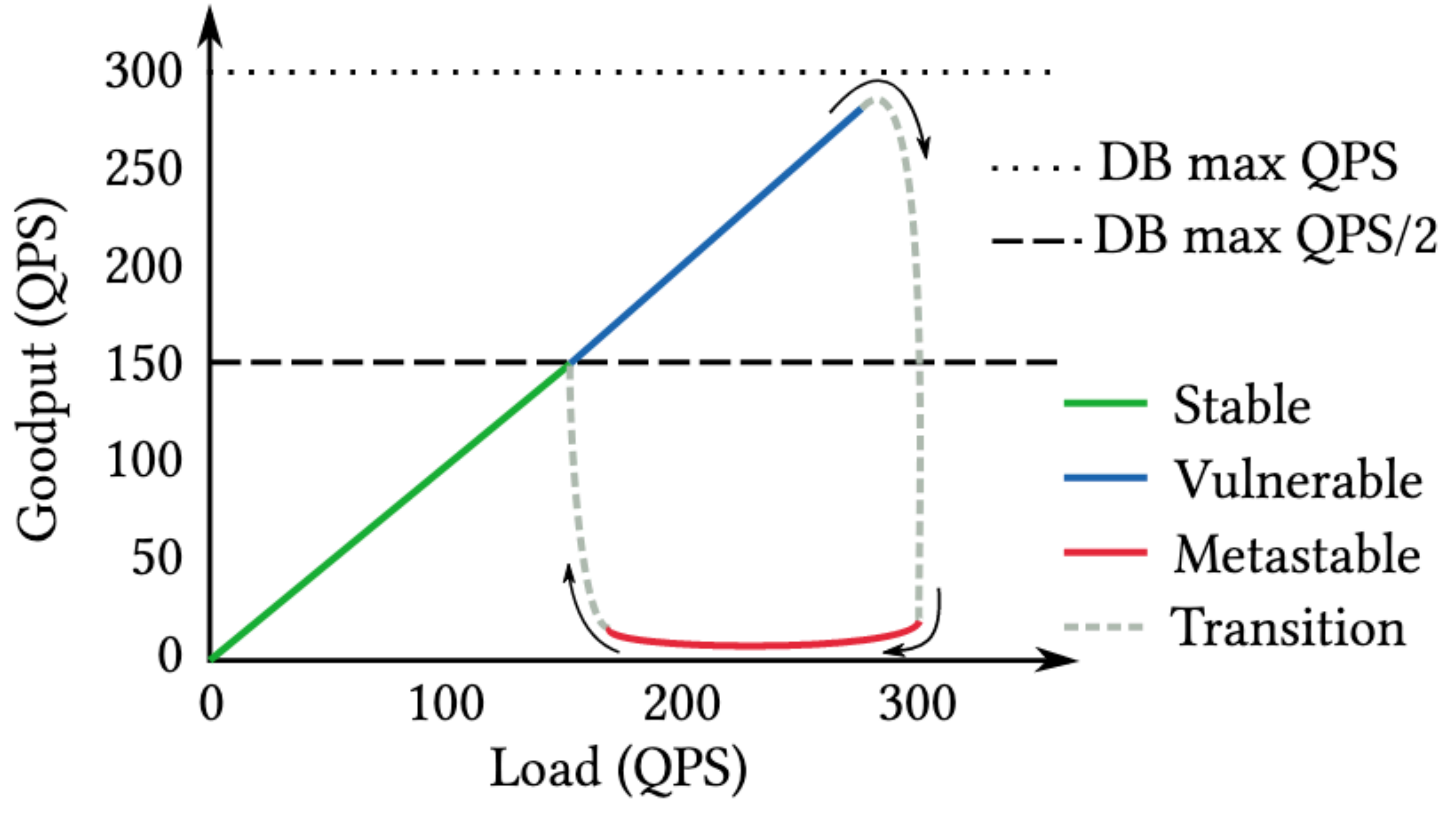
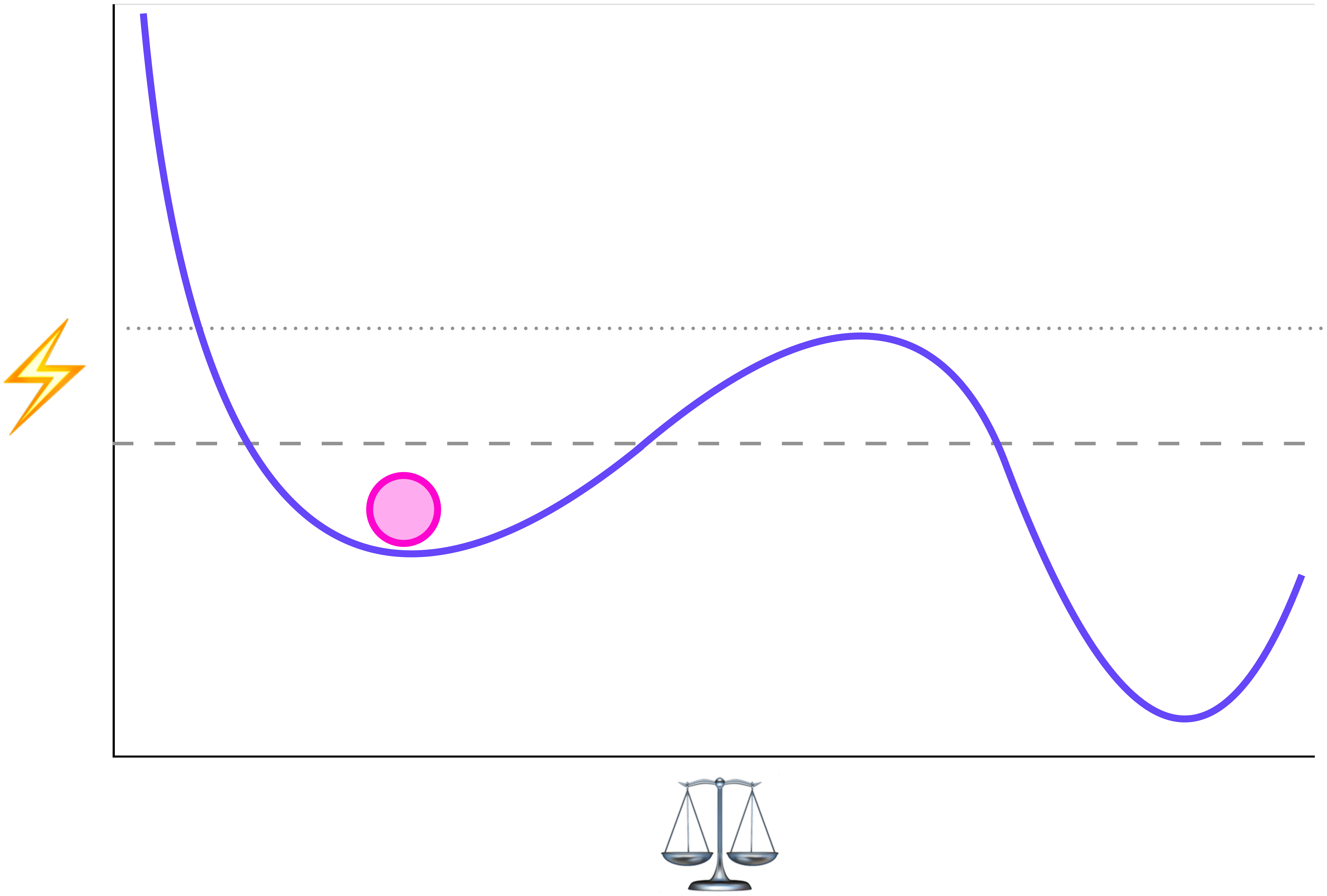


Figure 2: Goodput of an idealized web application with a database backend in stable, vulnerable, and metastable states.

Problems 🪨🪐🌌

Metastable Mechanism



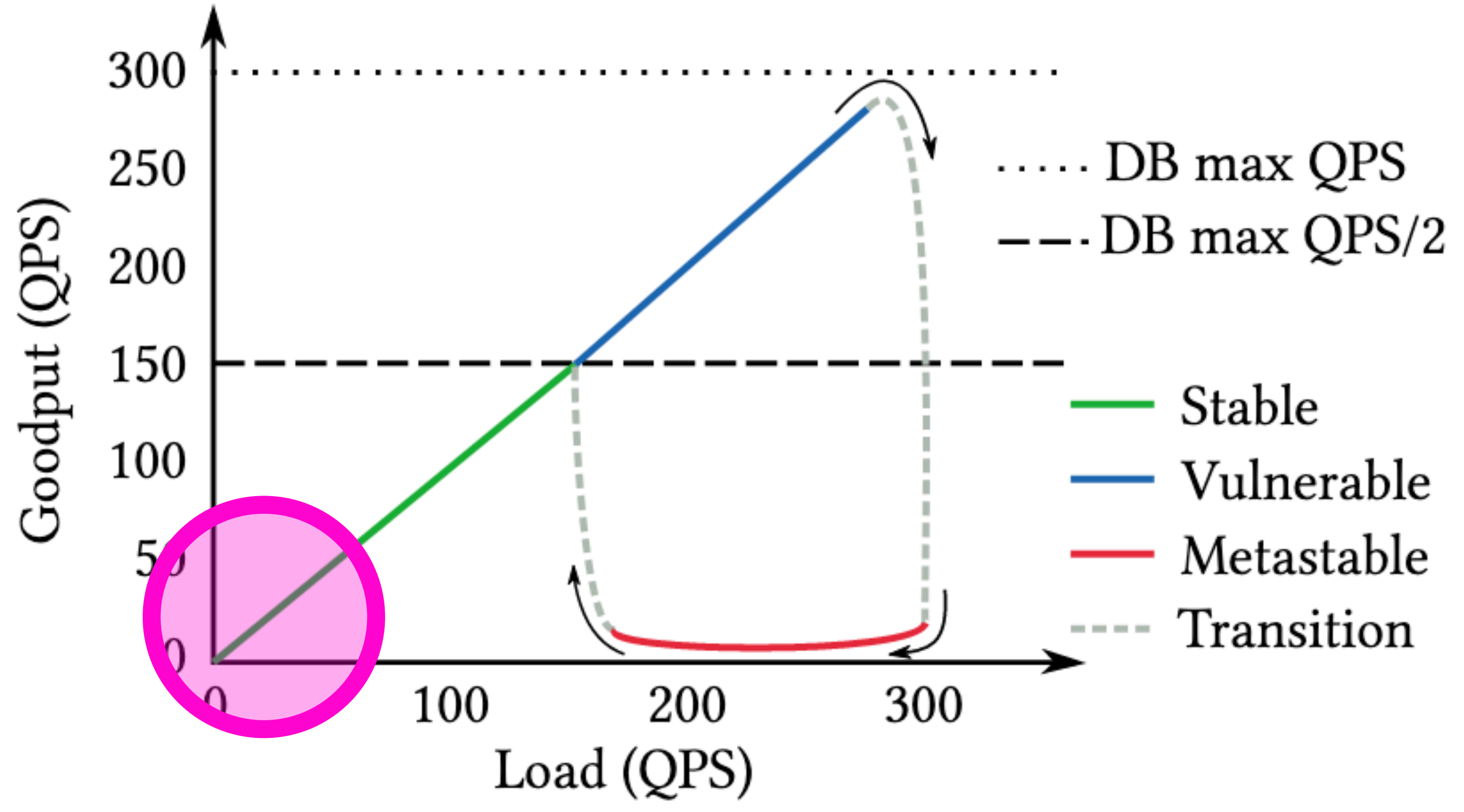
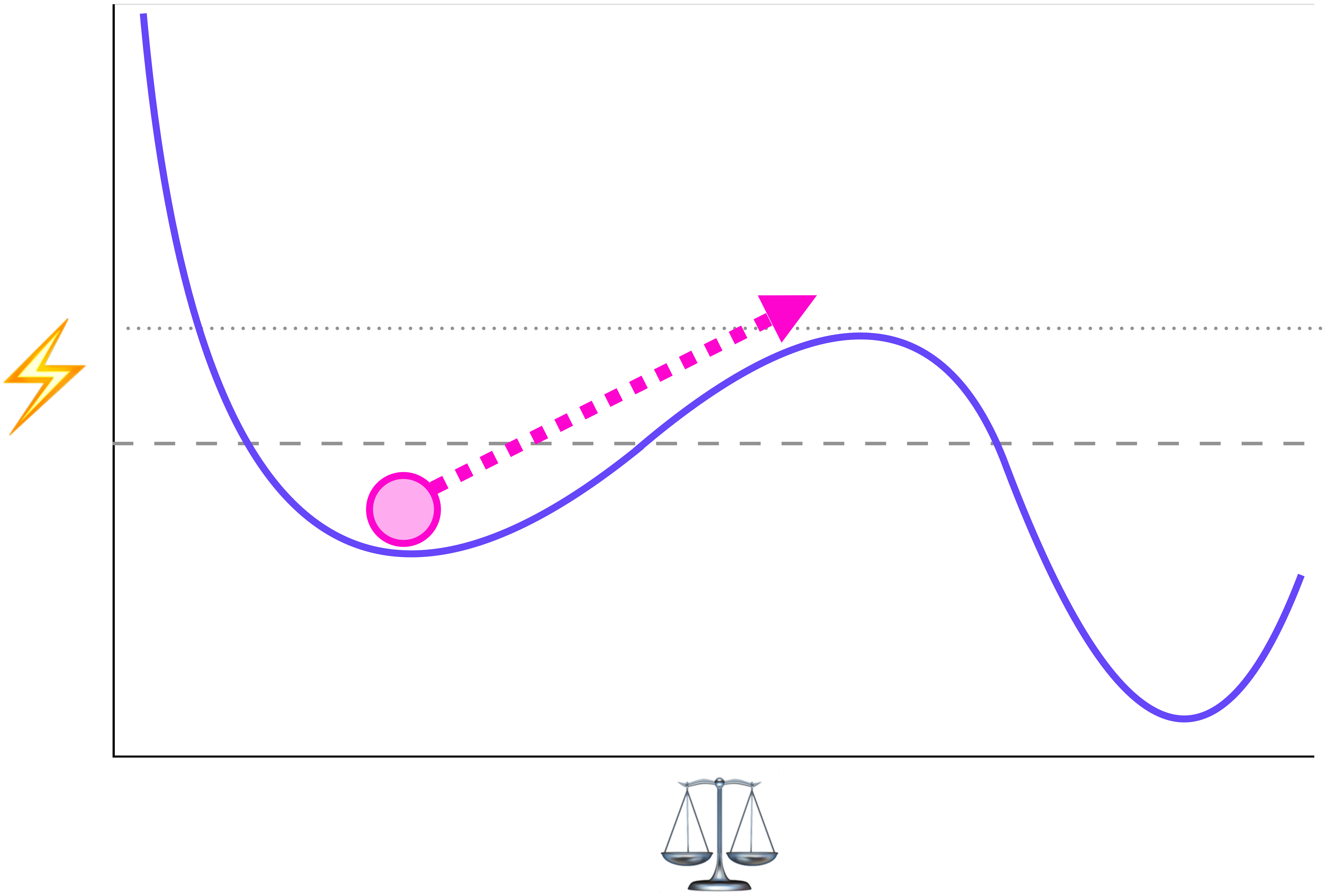


Figure 2: Goodput of an idealized web application with a database backend in stable, vulnerable, and metastable states.

Problems 🪨🪐🌌

Metastable Mechanism



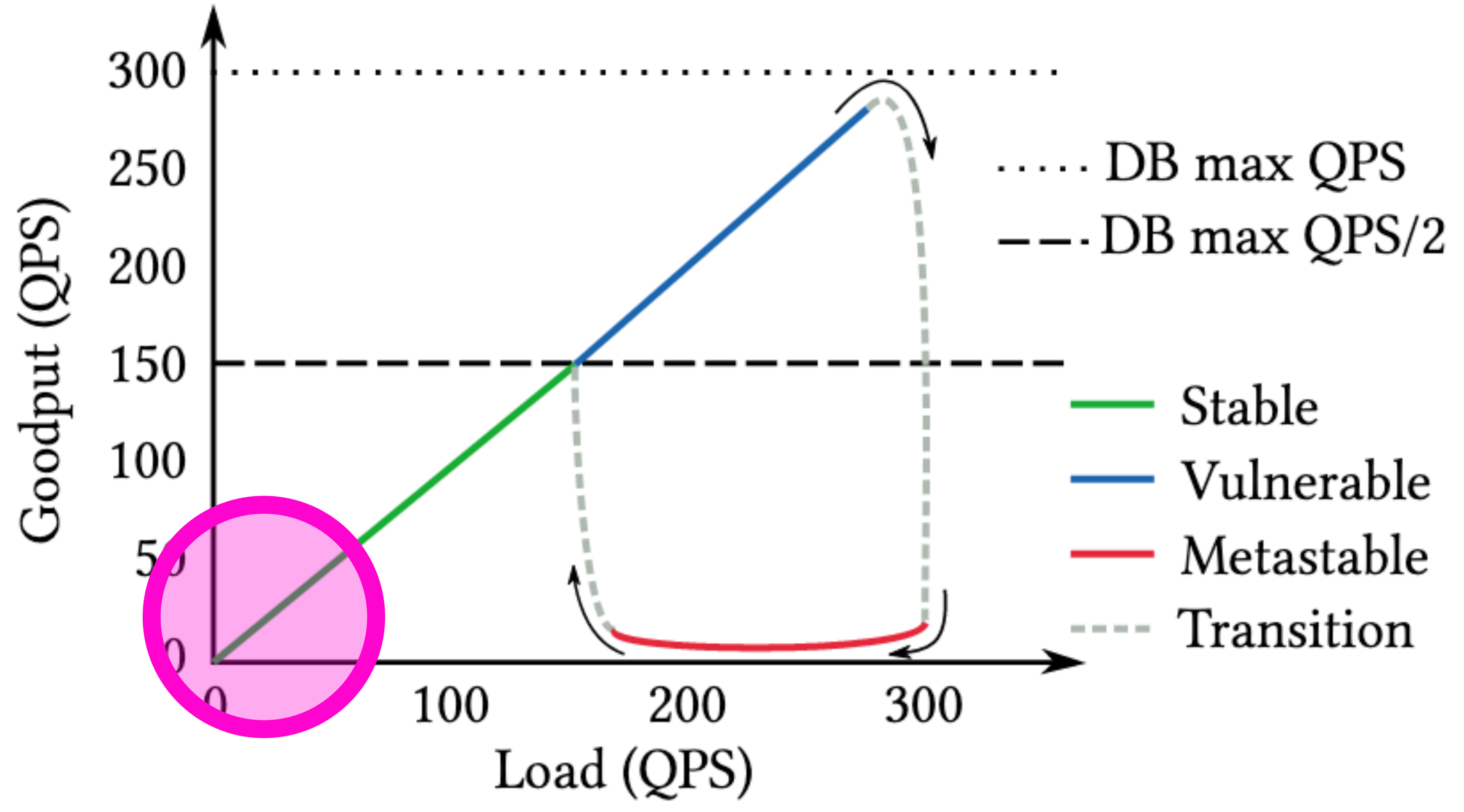
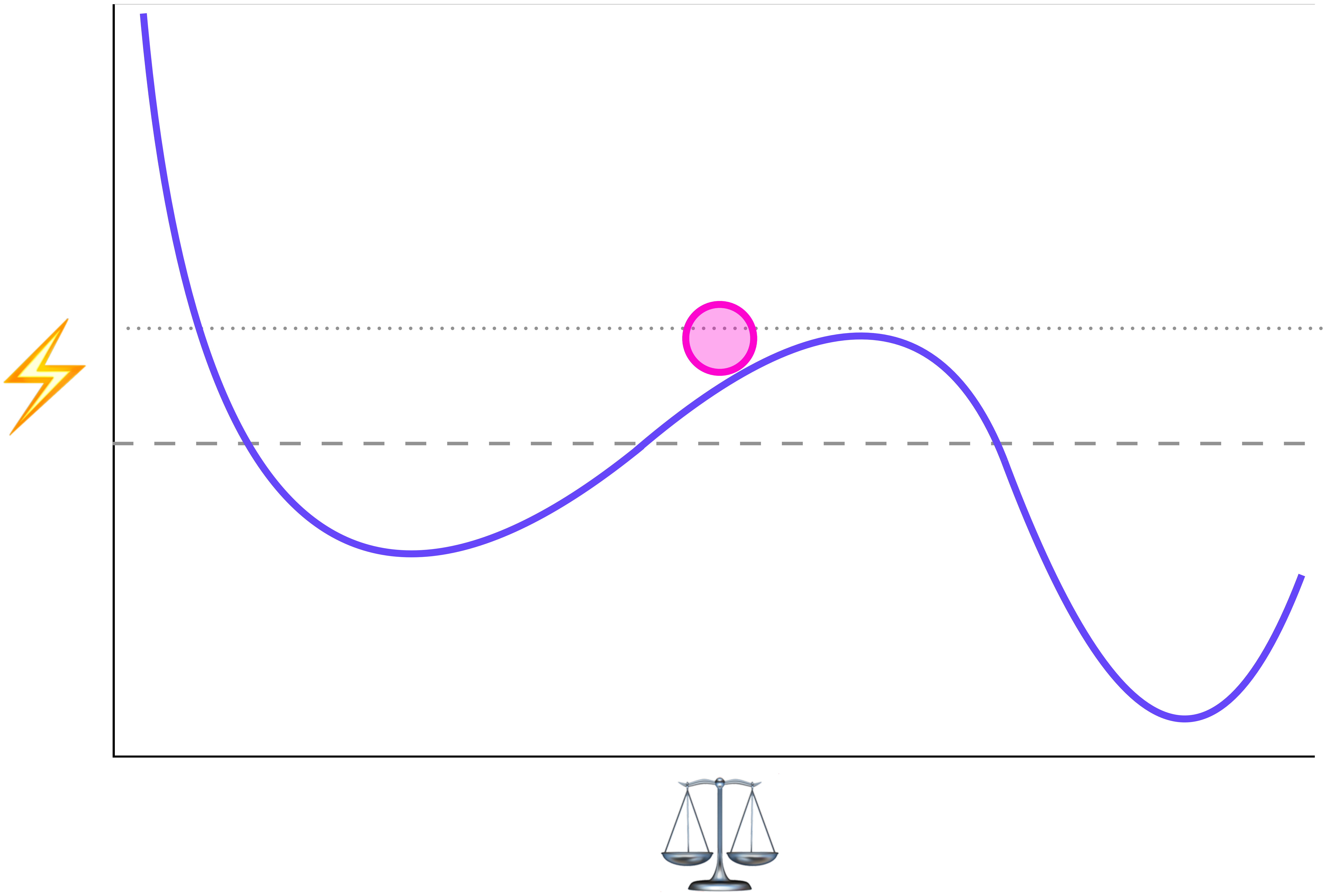


Figure 2: Goodput of an idealized web application with a database backend in stable, vulnerable, and metastable states.

Problems 🪨🪐🌌

Metastable Mechanism



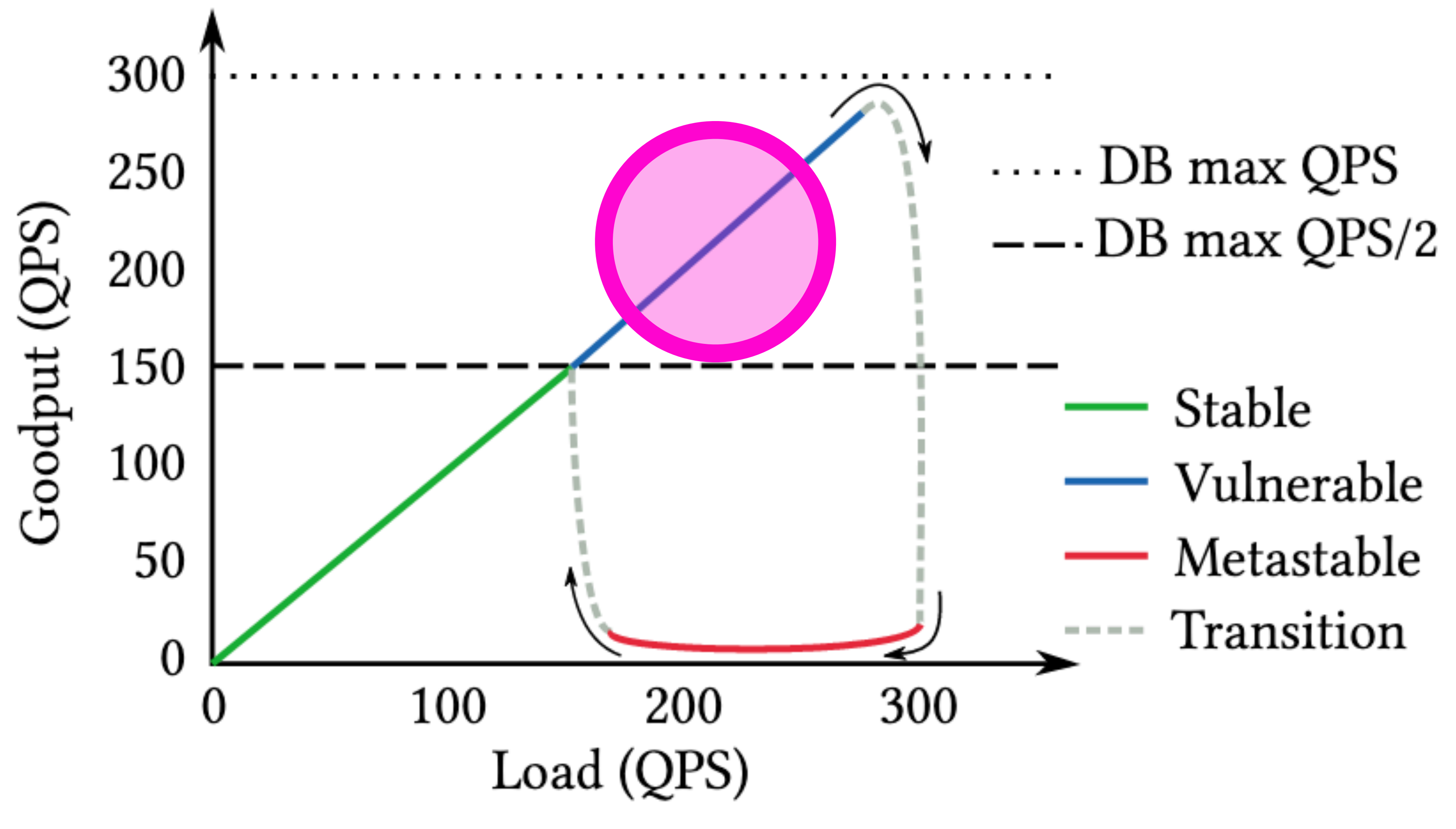
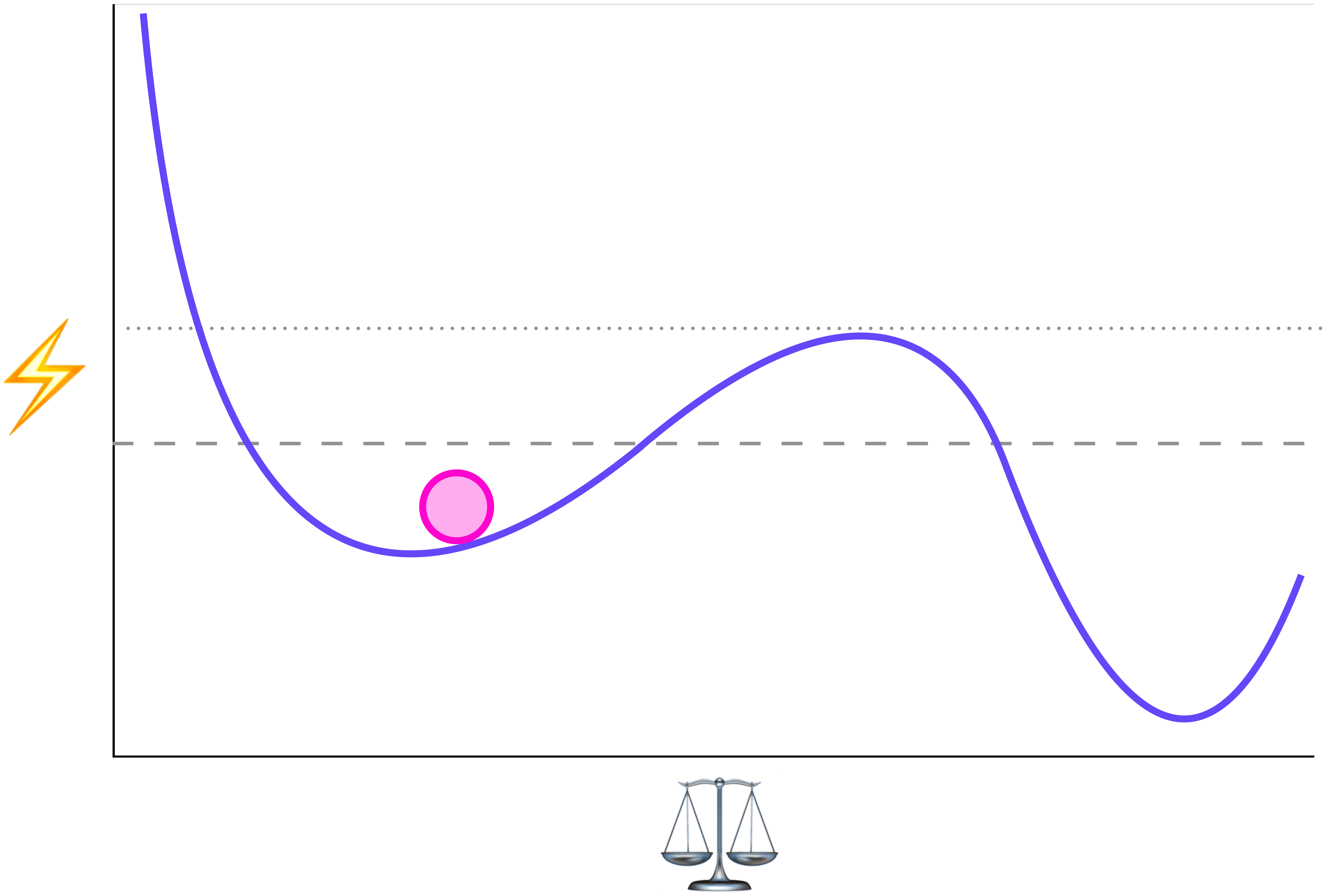


Figure 2: Goodput of an idealized web application with a database backend in stable, vulnerable, and metastable states.

Problems 🪨🪐🌌

Metastable Mechanism



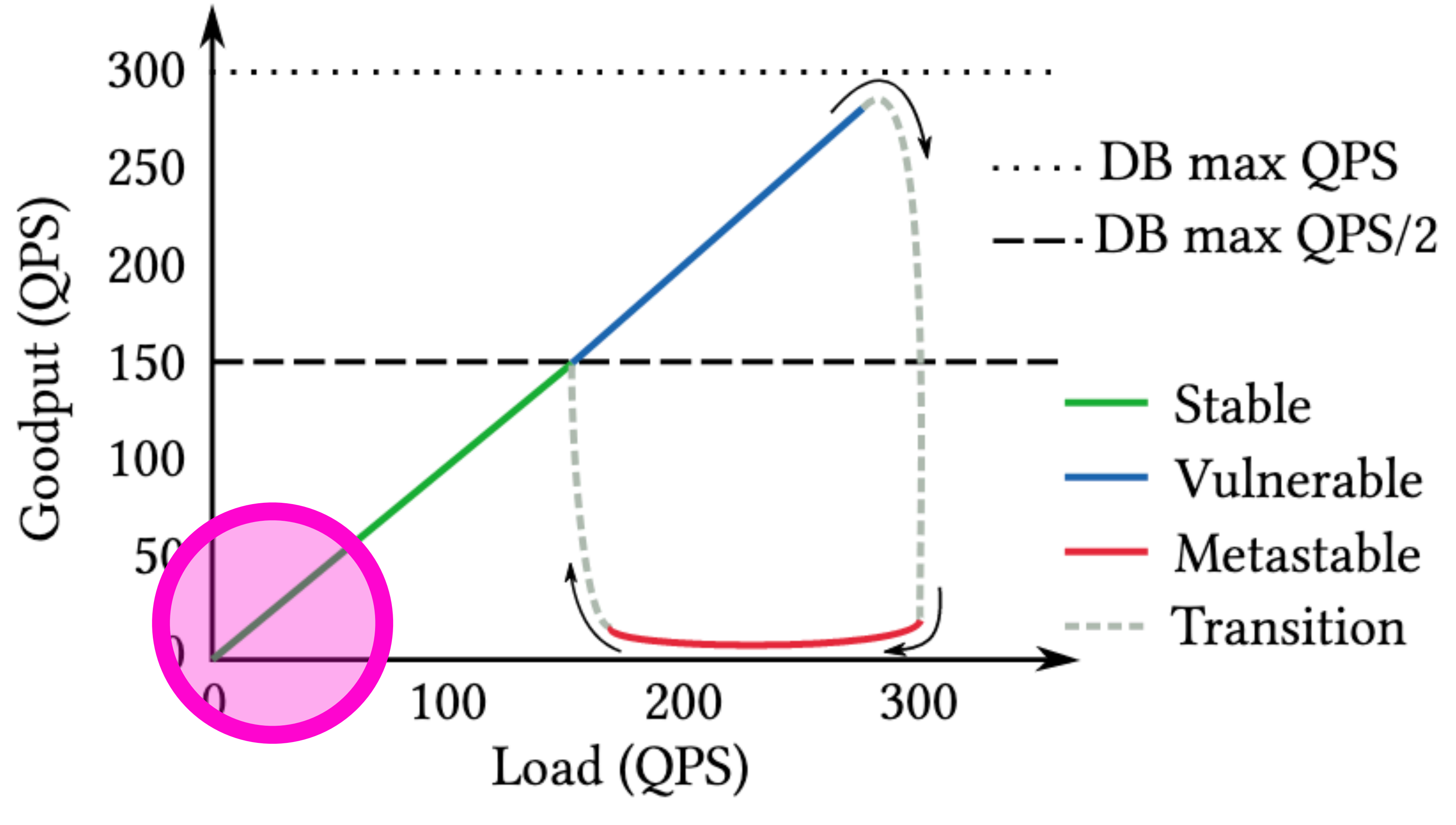
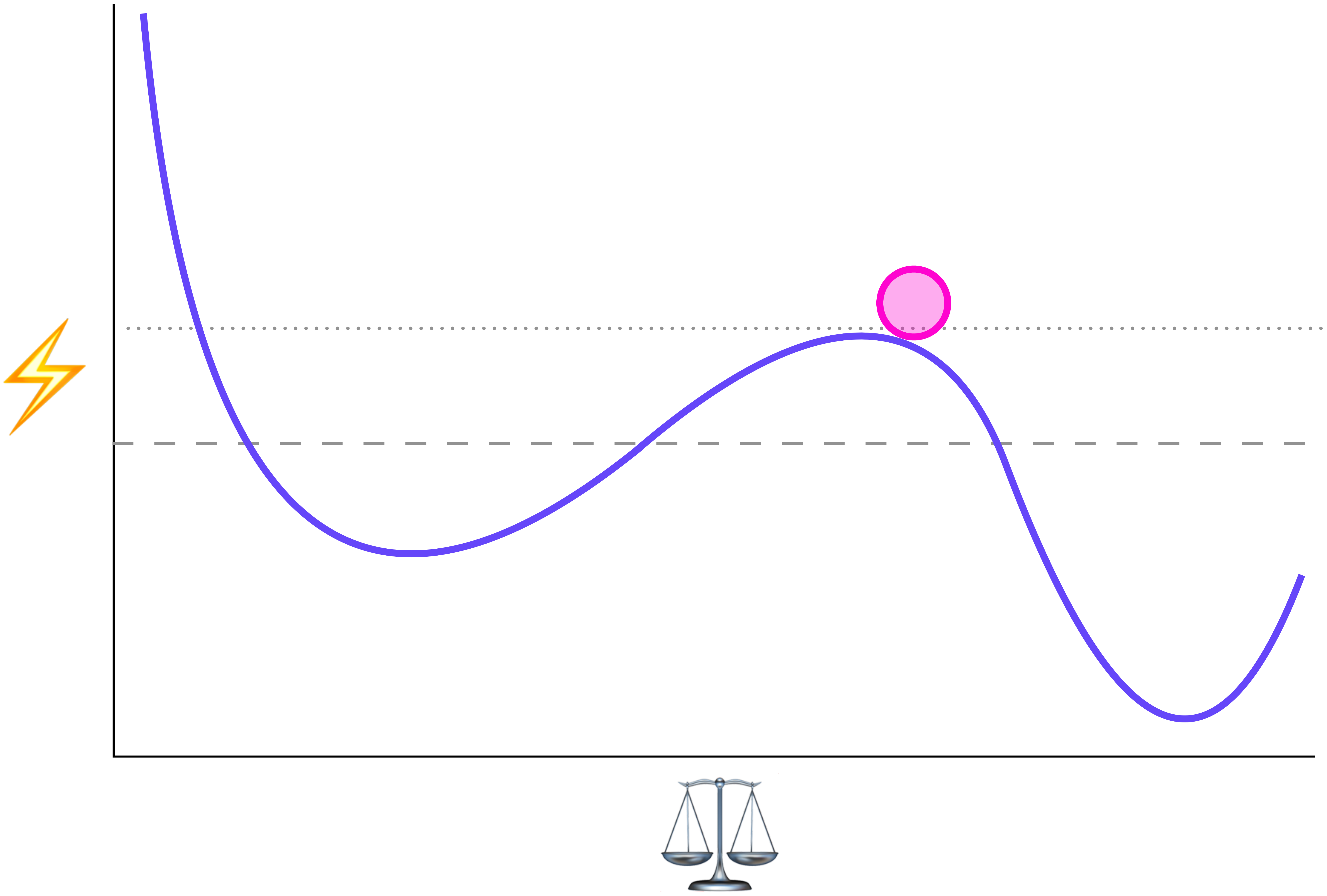


Figure 2: Goodput of an idealized web application with a database backend in stable, vulnerable, and metastable states.

Problems 🪨🪐🌌

Metastable Mechanism



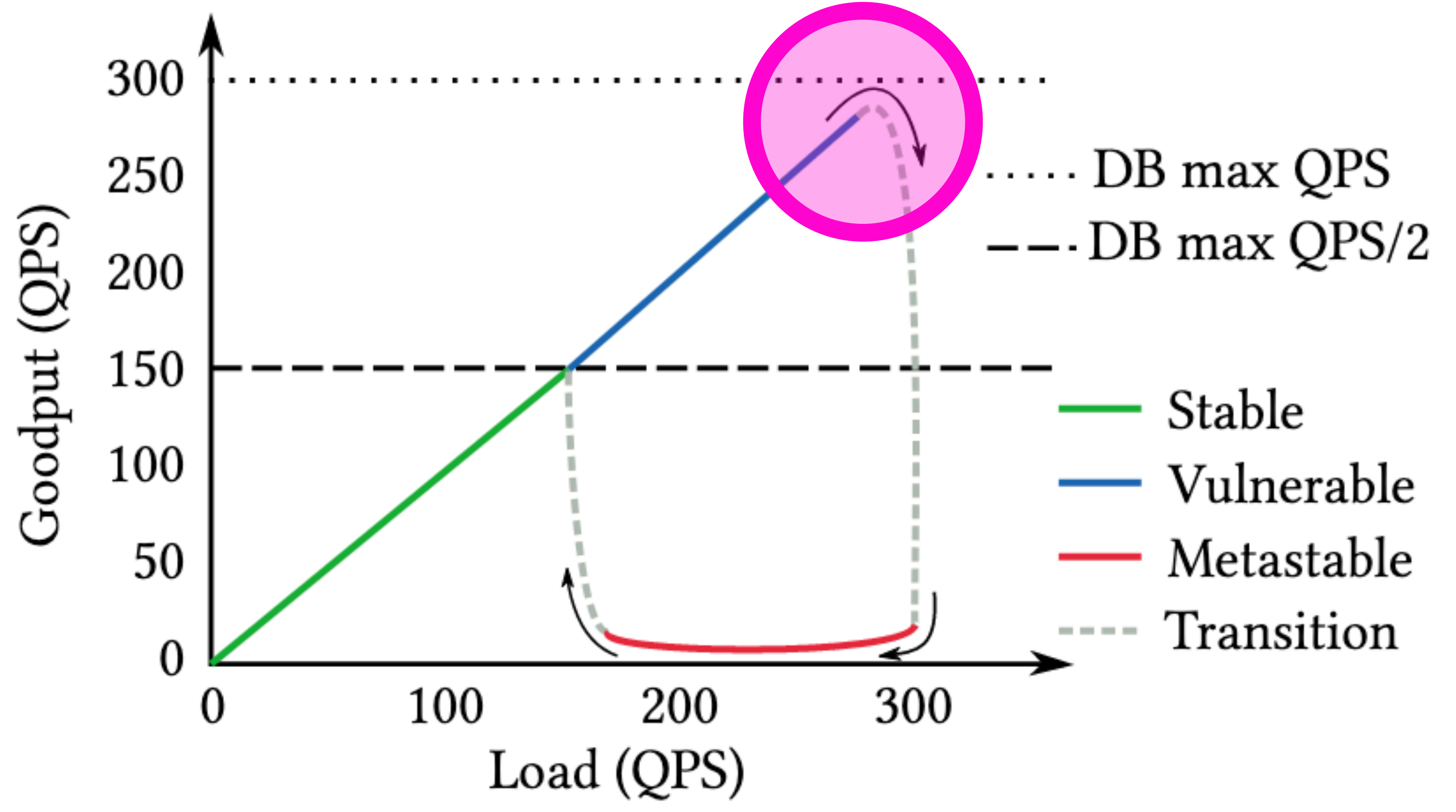
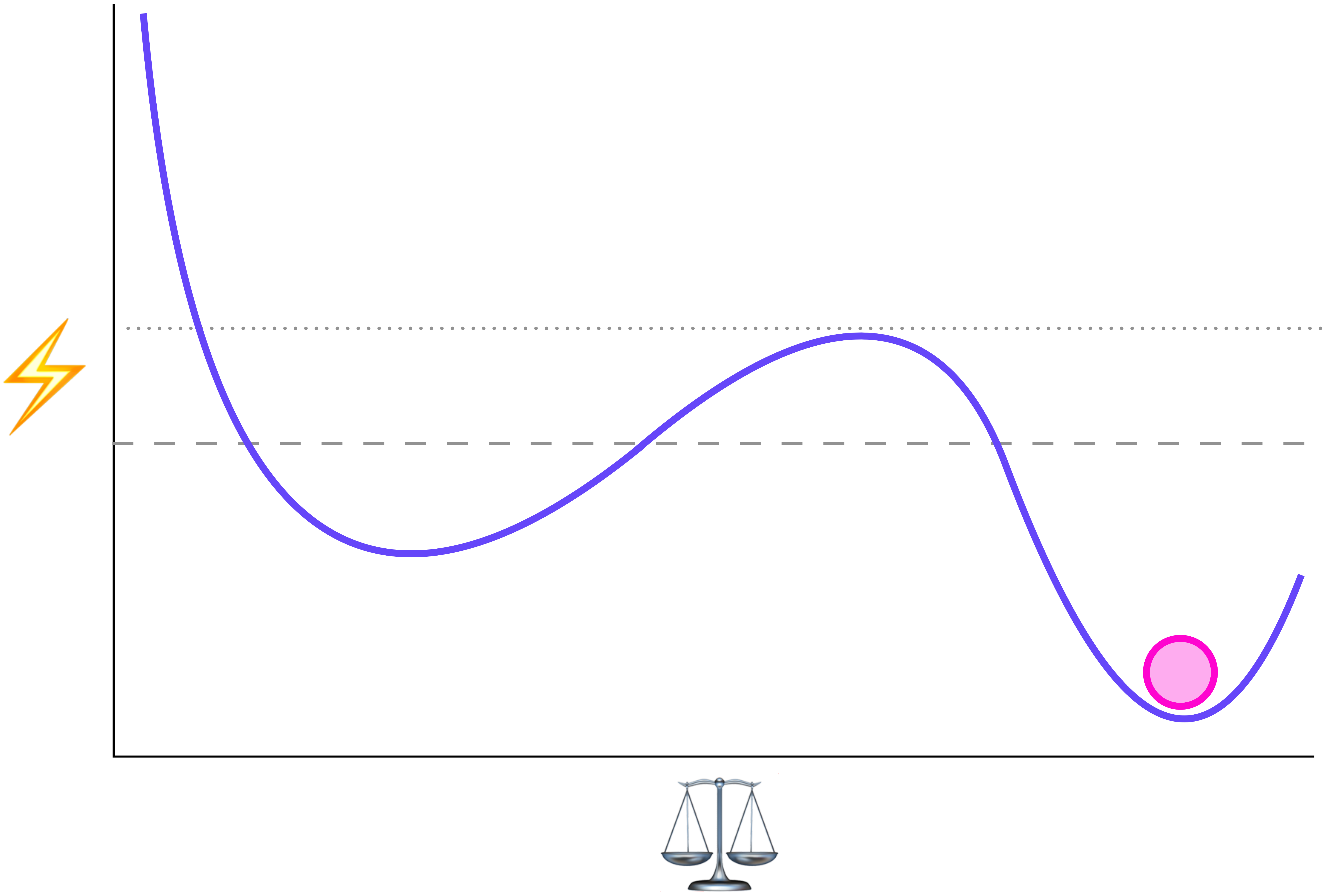


Figure 2: Goodput of an idealized web application with a database backend in stable, vulnerable, and metastable states.

Problems 🪨🪐🌌

Metastable Mechanism



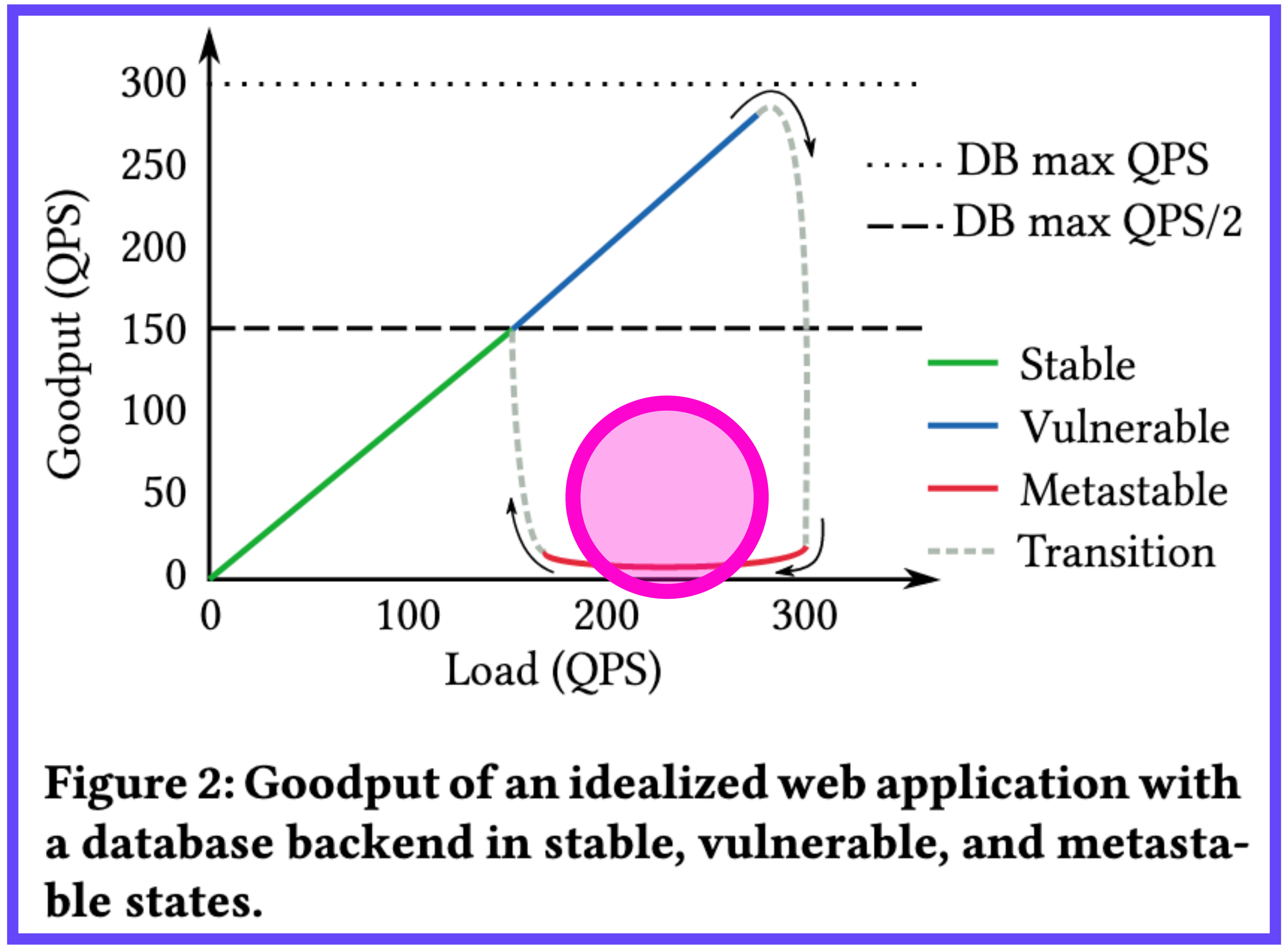
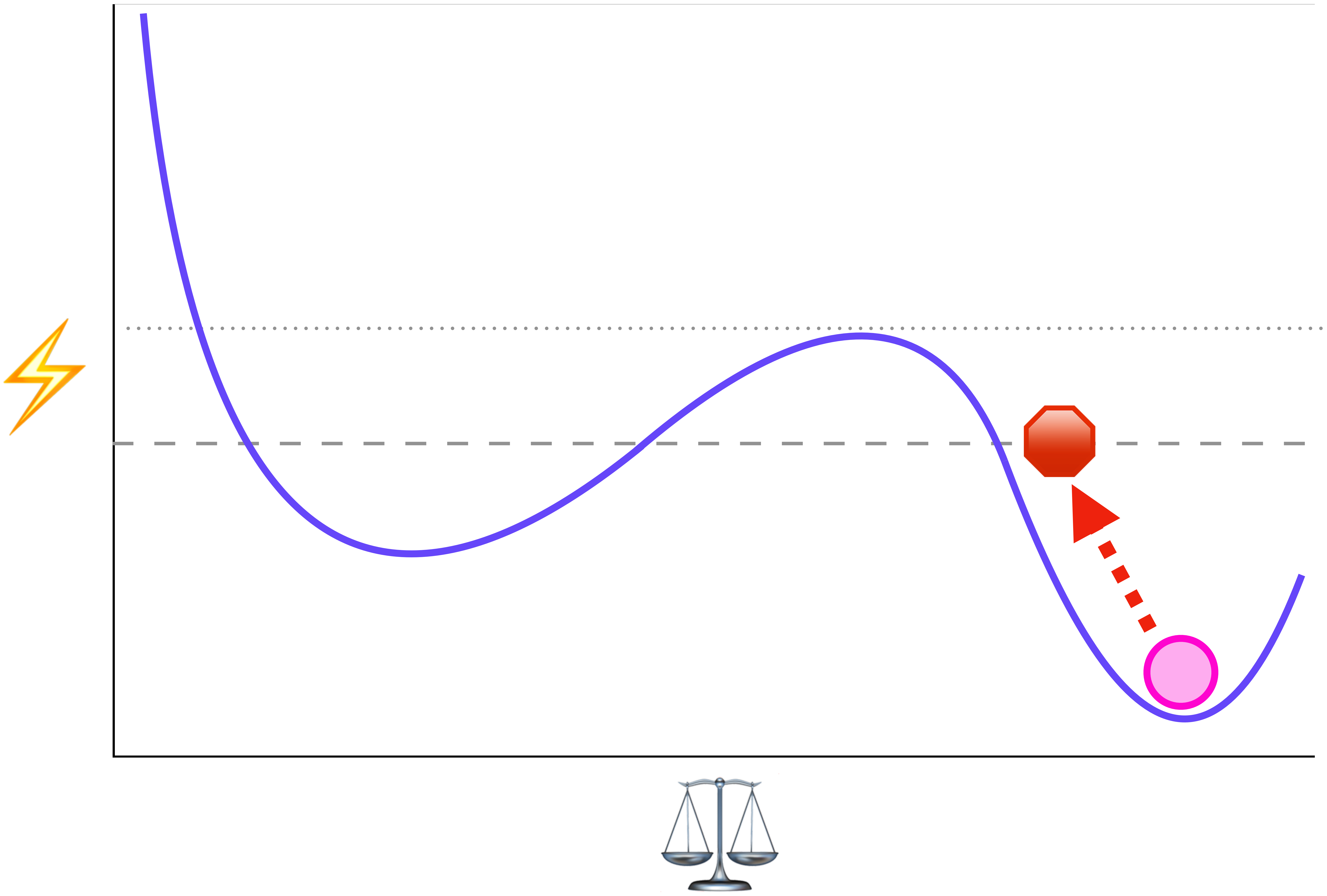


Figure 2: Goodput of an idealized web application with a database backend in stable, vulnerable, and metastable states.

Problems 🪨🪐🌌

Metastable Mechanism



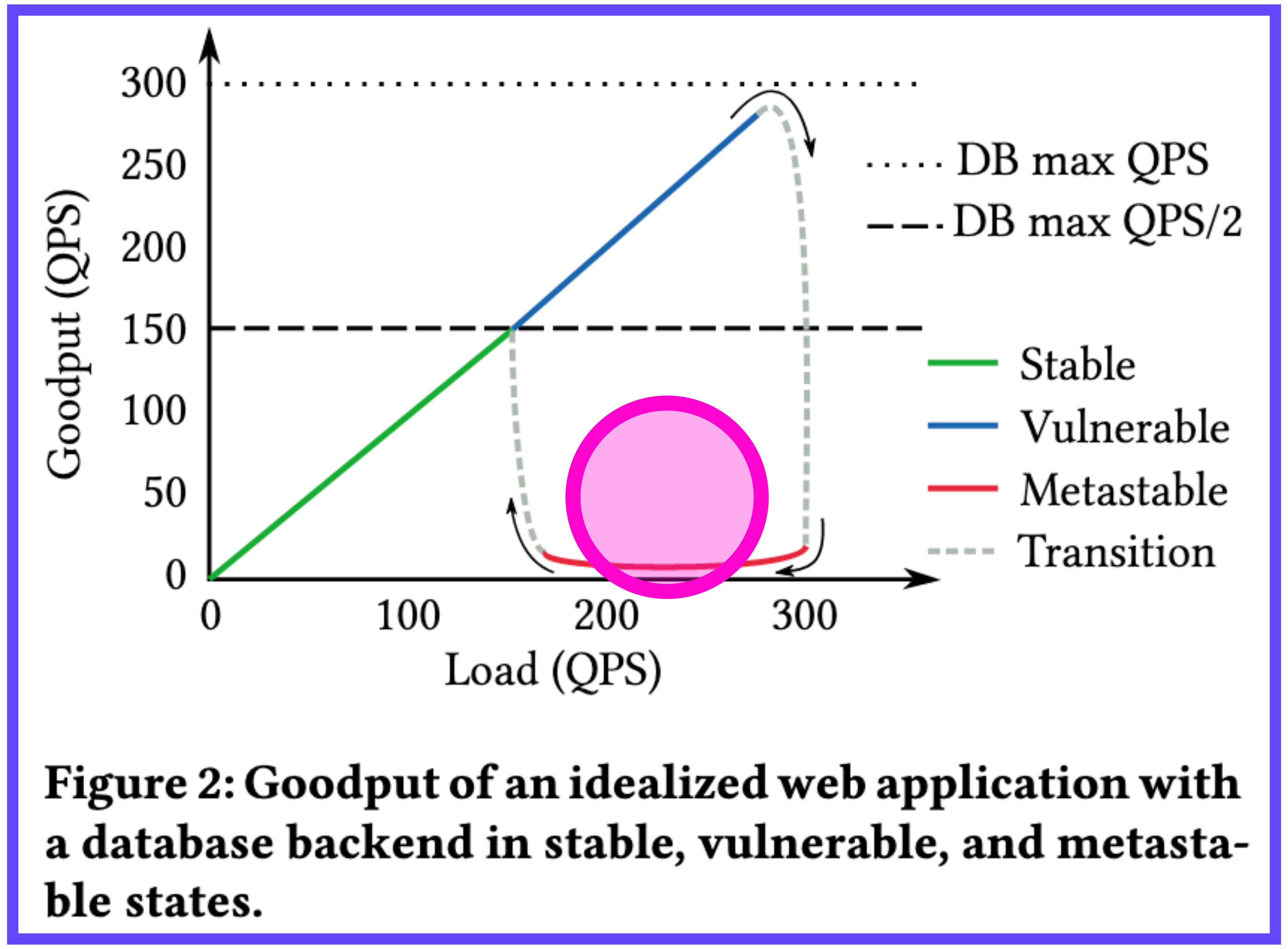
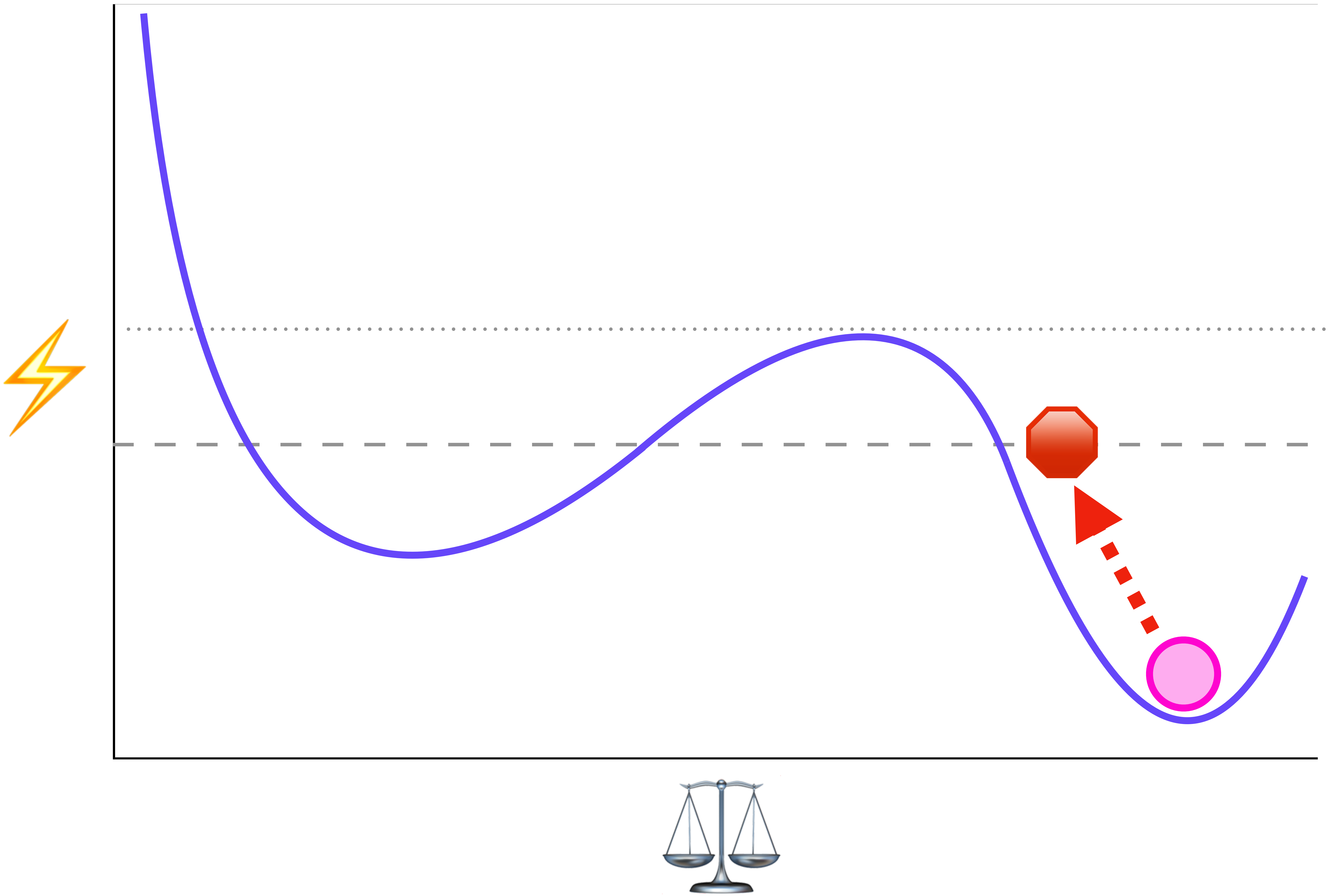


Figure 2: Goodput of an idealized web application with a database backend in stable, vulnerable, and metastable states.

Problems 🪨🪐🌌

Metastable Mechanism



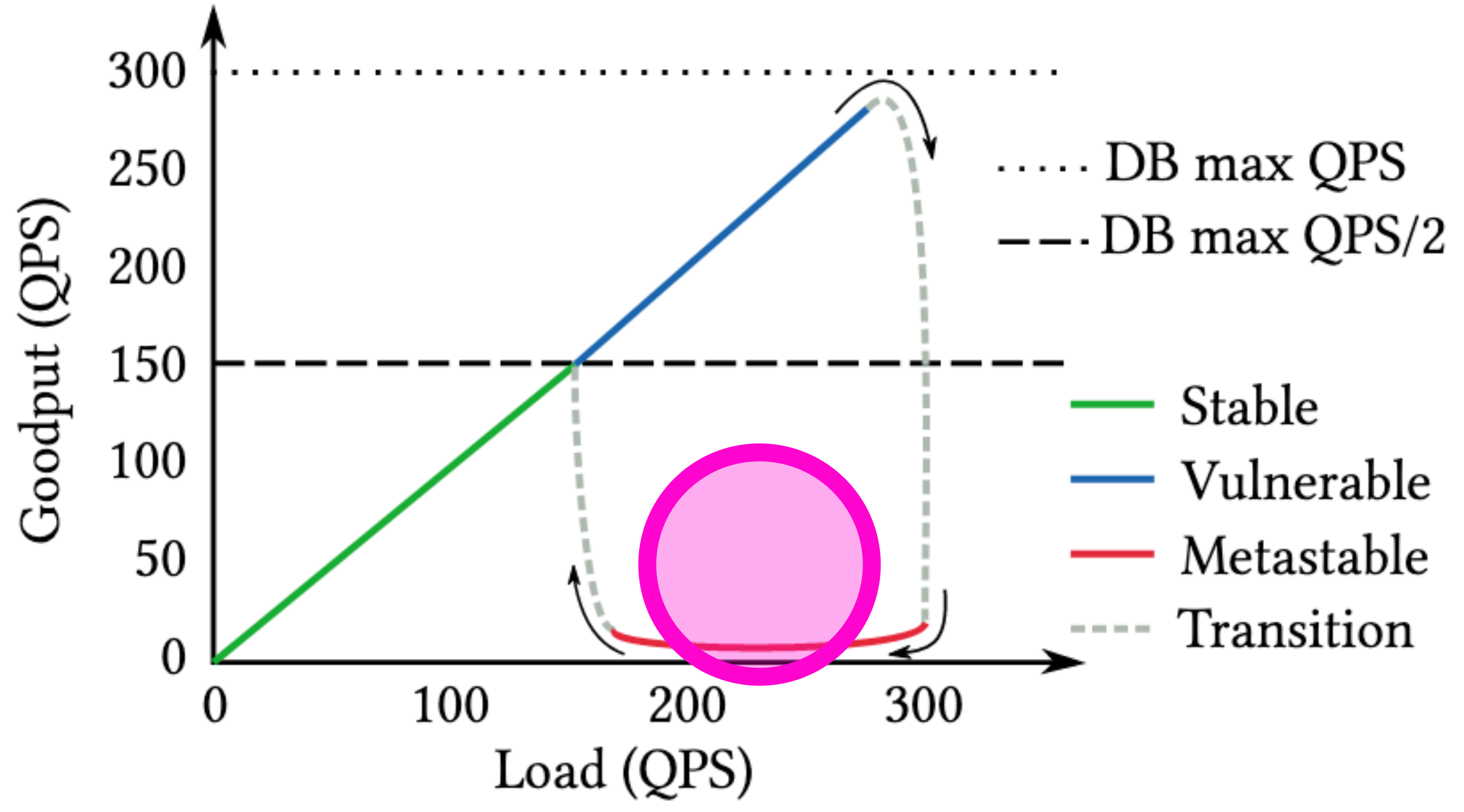


Figure 2: Goodput of an idealized web application with a database backend in stable, vulnerable, and metastable states.

- Retries / let it crash
- Work amplification
- General thrash 🫡

Problems 🪨🪐🌌

Places Fight

Light

💫🤺

Problems 🪨🪐🌌

Places Fight Light 💫🤺

Problems 🪨🪐🌌

Places Fight Light 💫🤺

The limitation of ***local knowledge***
is the ***fundamental fact***
about the setting in which we work,
and it is ***a very powerful limitation***

– Nancy Lynch, A Hundred Impossibility Proofs for Distributed Computing

Problems 🪨🪐🌌

Places Fight Light 💫🤺

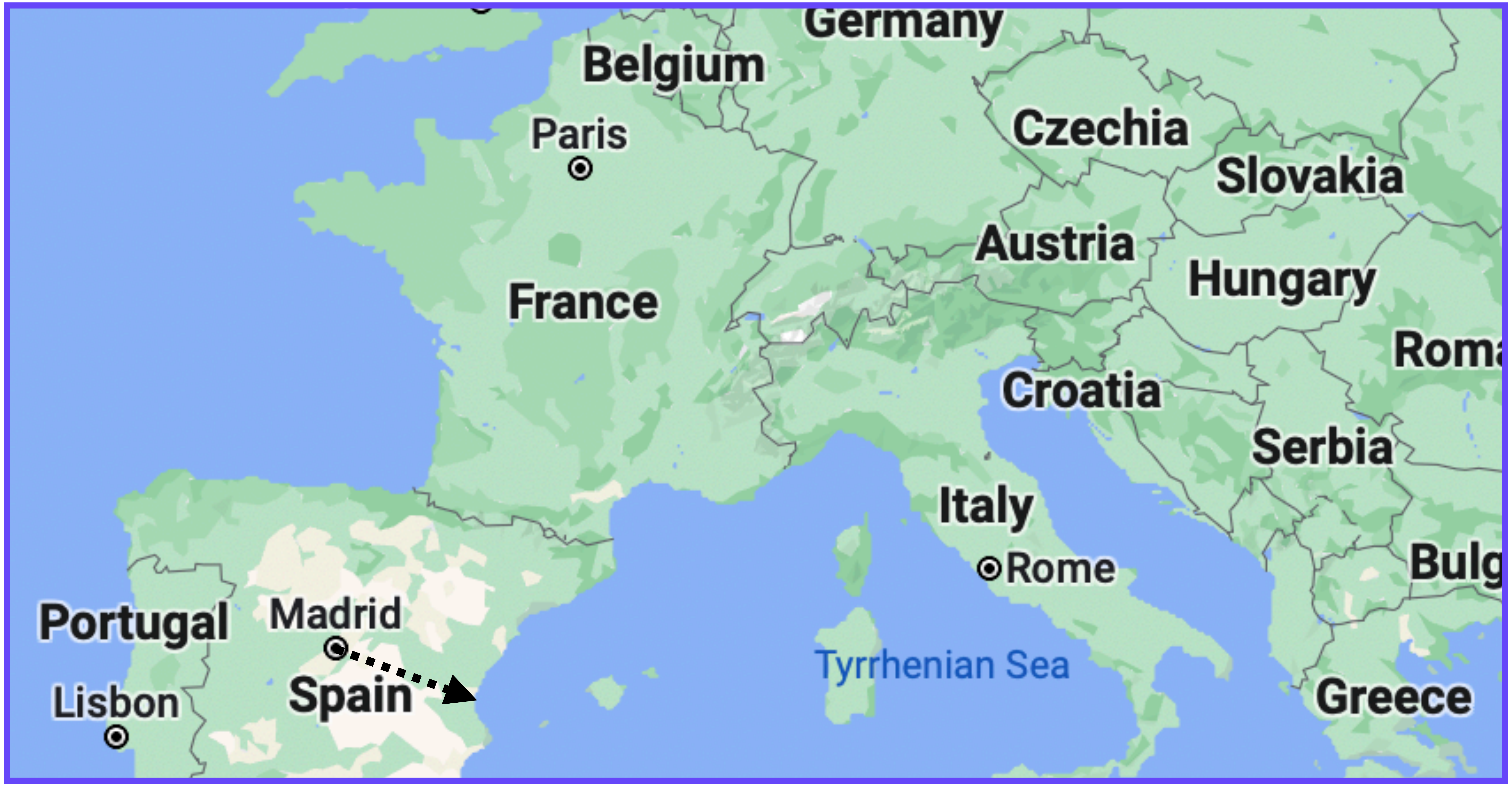
Sending a "Direct" Message

Germany
Belgium
Paris
Czechia
Slovakia
Austria
France
Hungary
Roma
Croatia
Serbia
Italy
Rome
Bulg
Portugal
Madrid
Spain
Lisbon
Tyrrhenian Sea
Greece

Problems 🪨🪐🌌

Places Fight Light 💫🤺

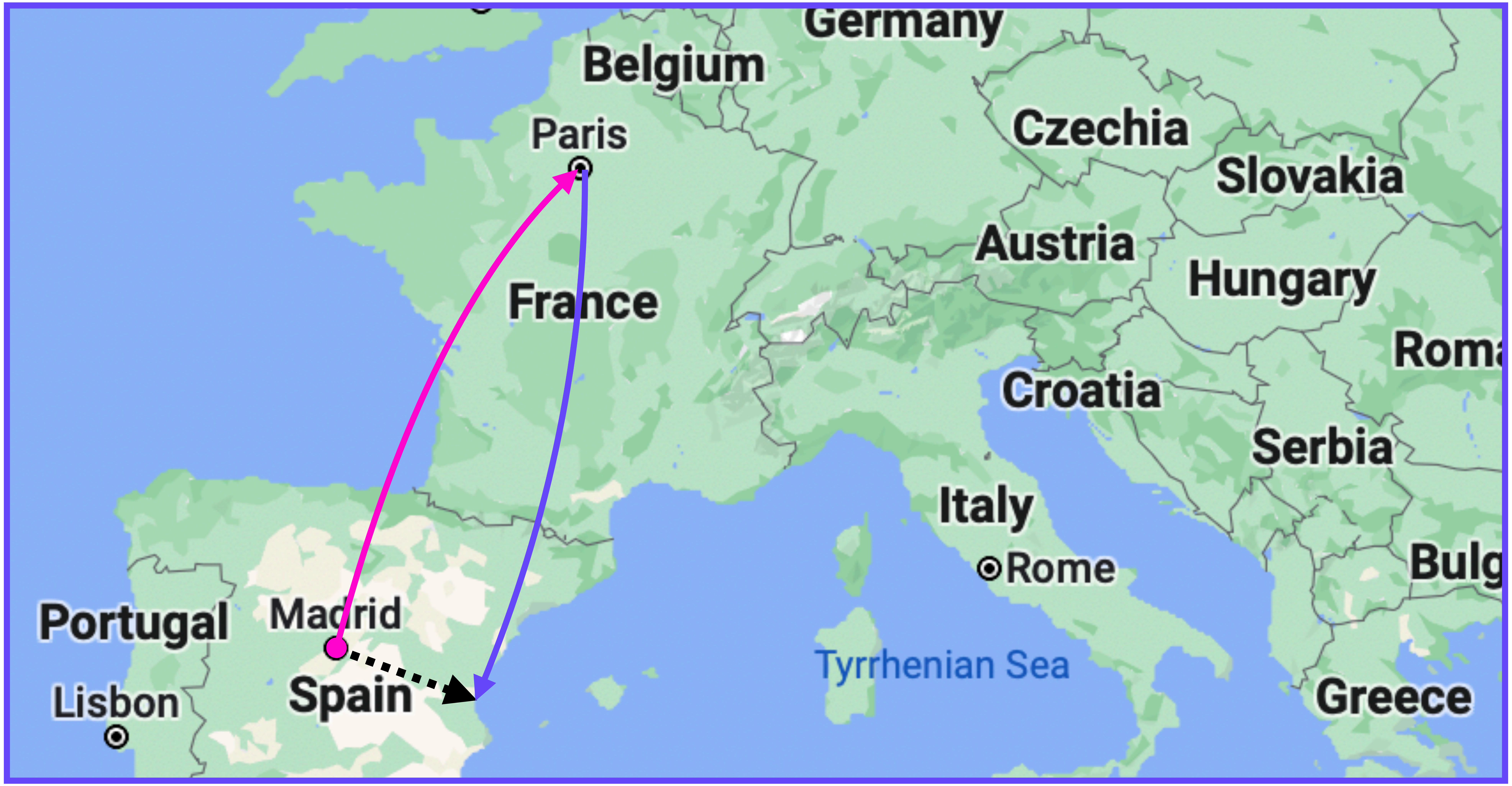
Sending a “Direct” Message



Problems 🪨🪐🌌

Places Fight Light 💫🤺

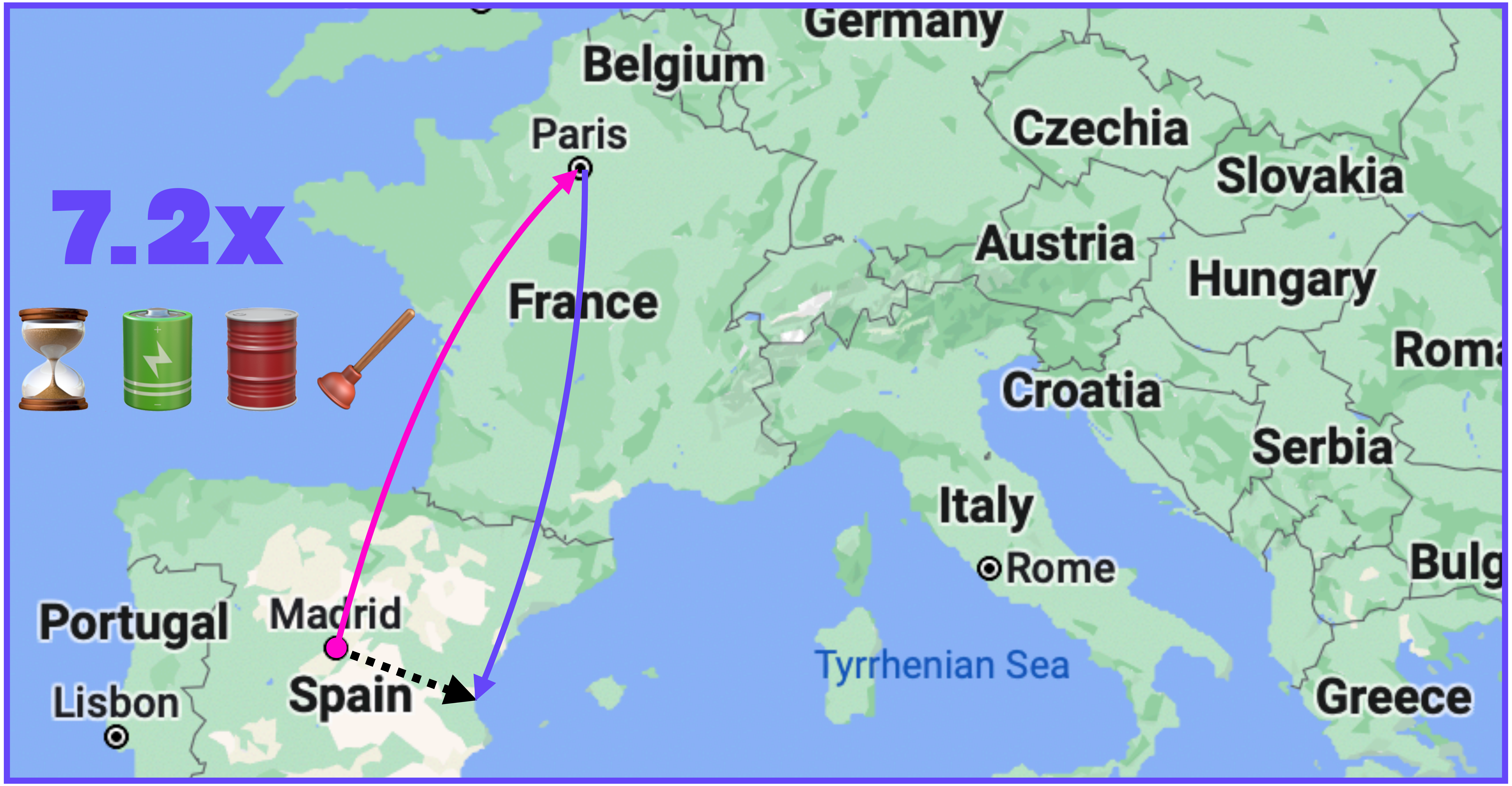
Sending a "Direct" Message



Problems 🪨🪐🌌

Places Fight Light 💫🤺

Sending a "Direct" Message



Problems 🪨🪐🌌

Data Behind

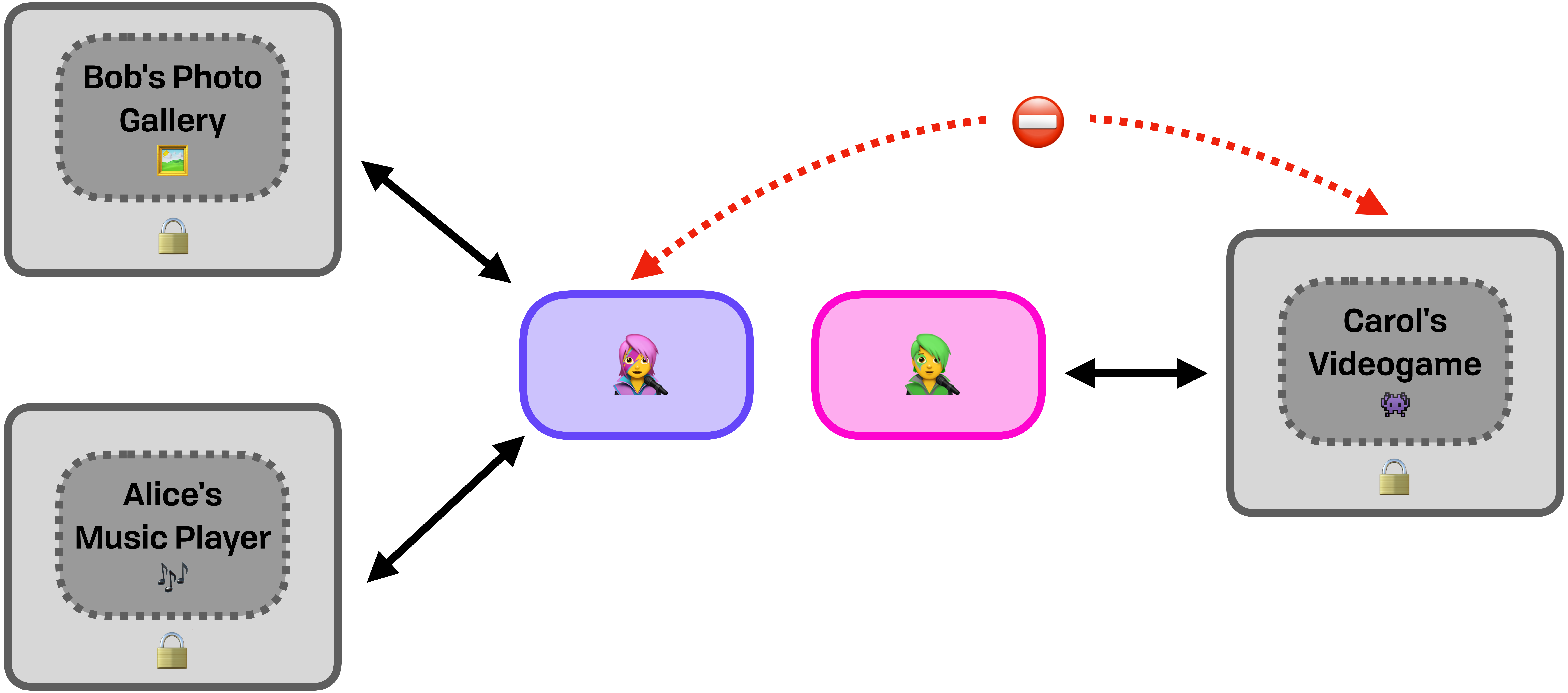
Walls

🏰

Problems 🪨🪐🌌

Data Behind Walls 🏰

Dependencies & Integration



Problems 🪨🪐🌌

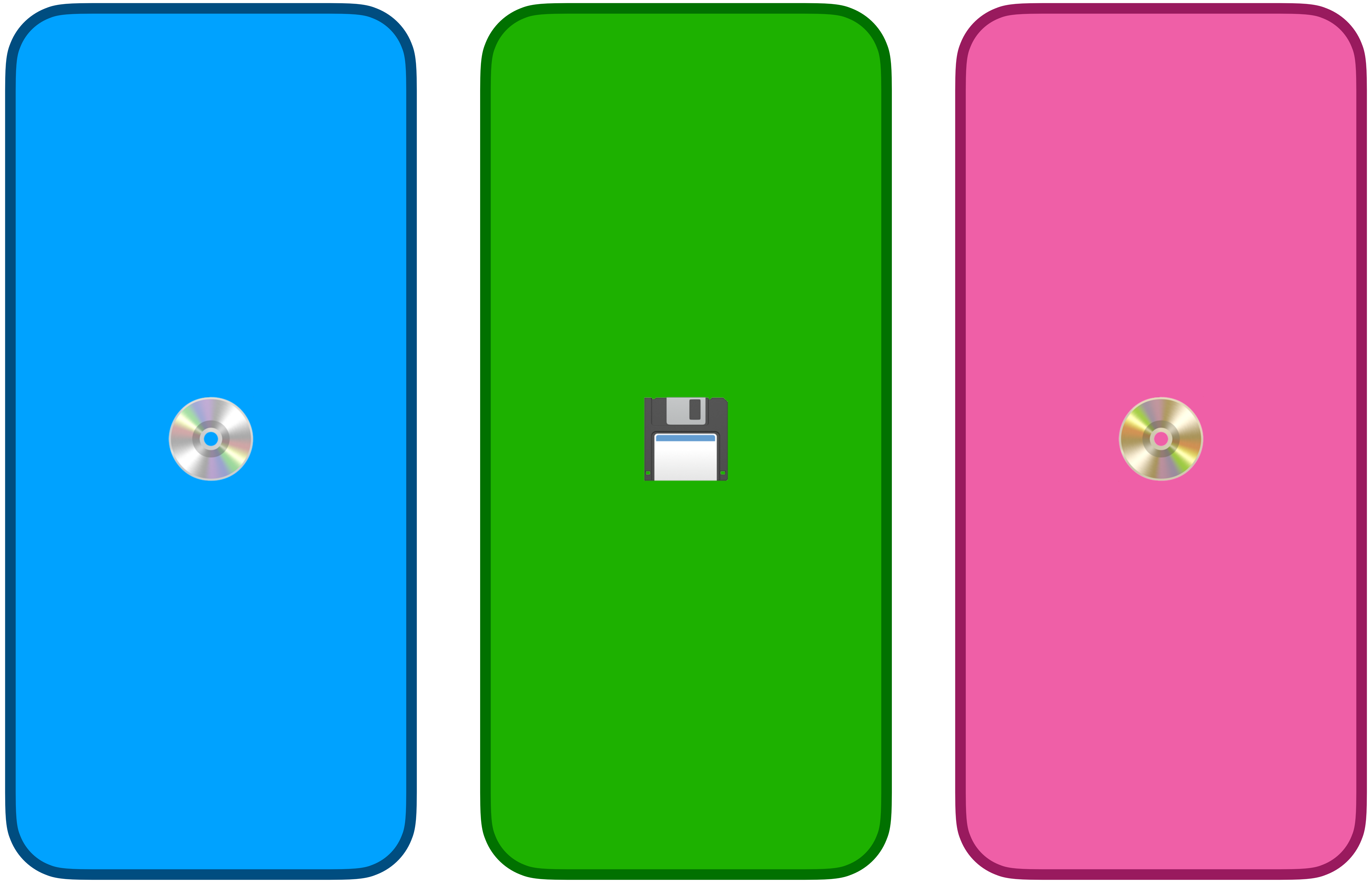
Data Behind Walls 🏰

Dependencies & Integration

Problems 🪨🪐🌌

Data Behind Walls 🏰

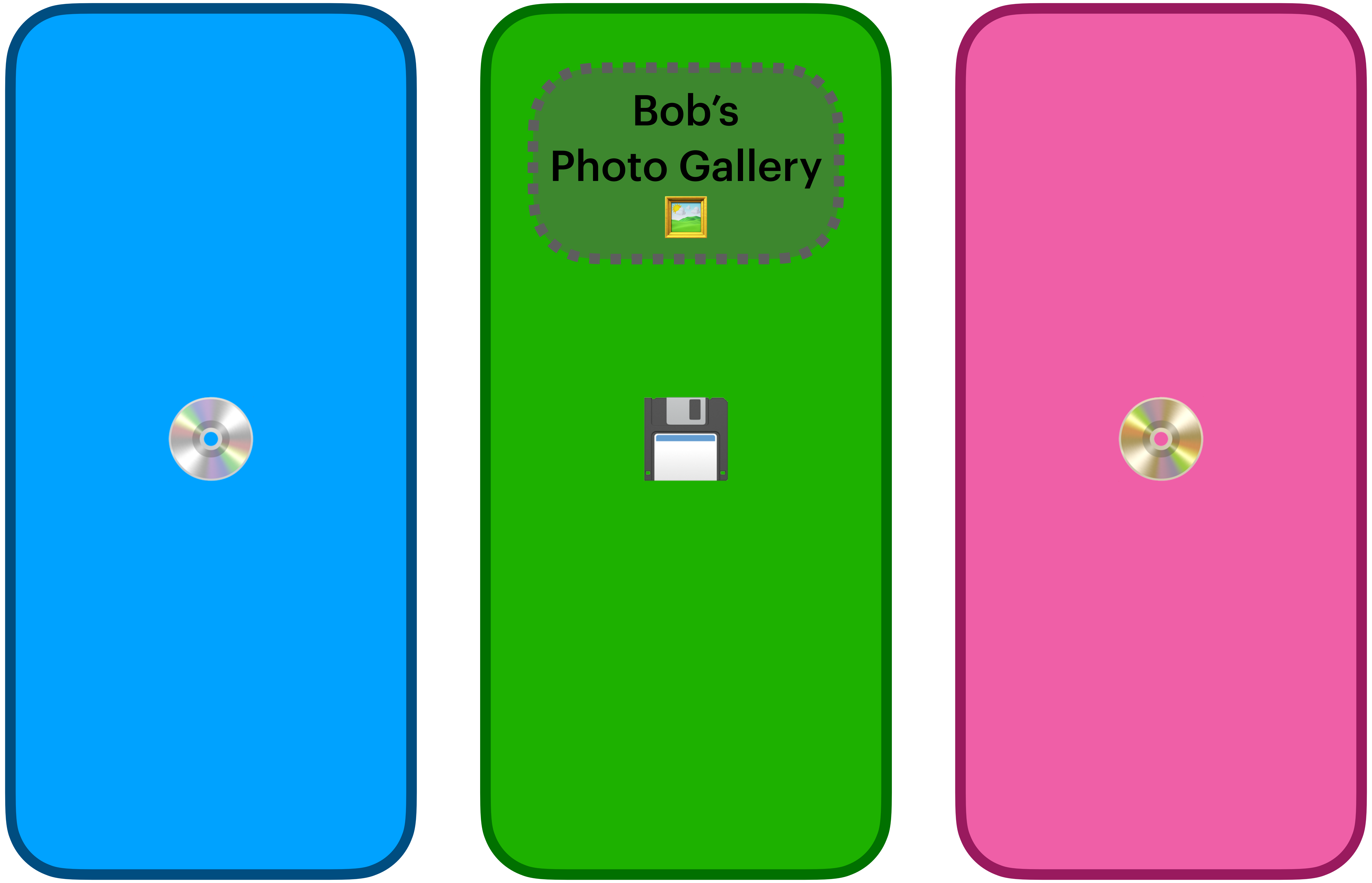
Dependencies & Integration



Problems 🪨🪐🌌

Data Behind Walls 🏰

Dependencies & Integration

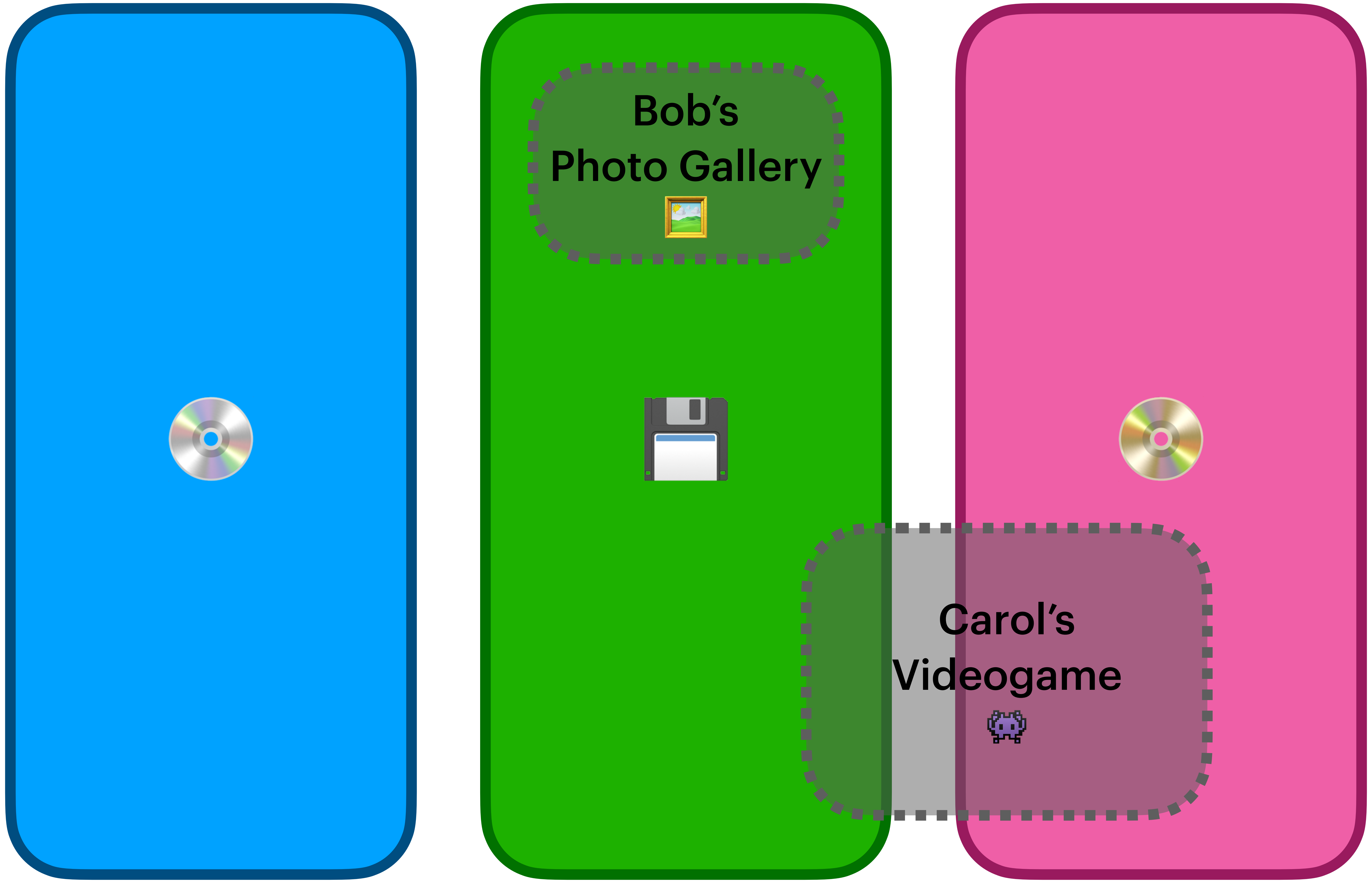


Bob’s
Photo Gallery
🖼️

Problems 🪨🪐🌌

Data Behind Walls 🏰

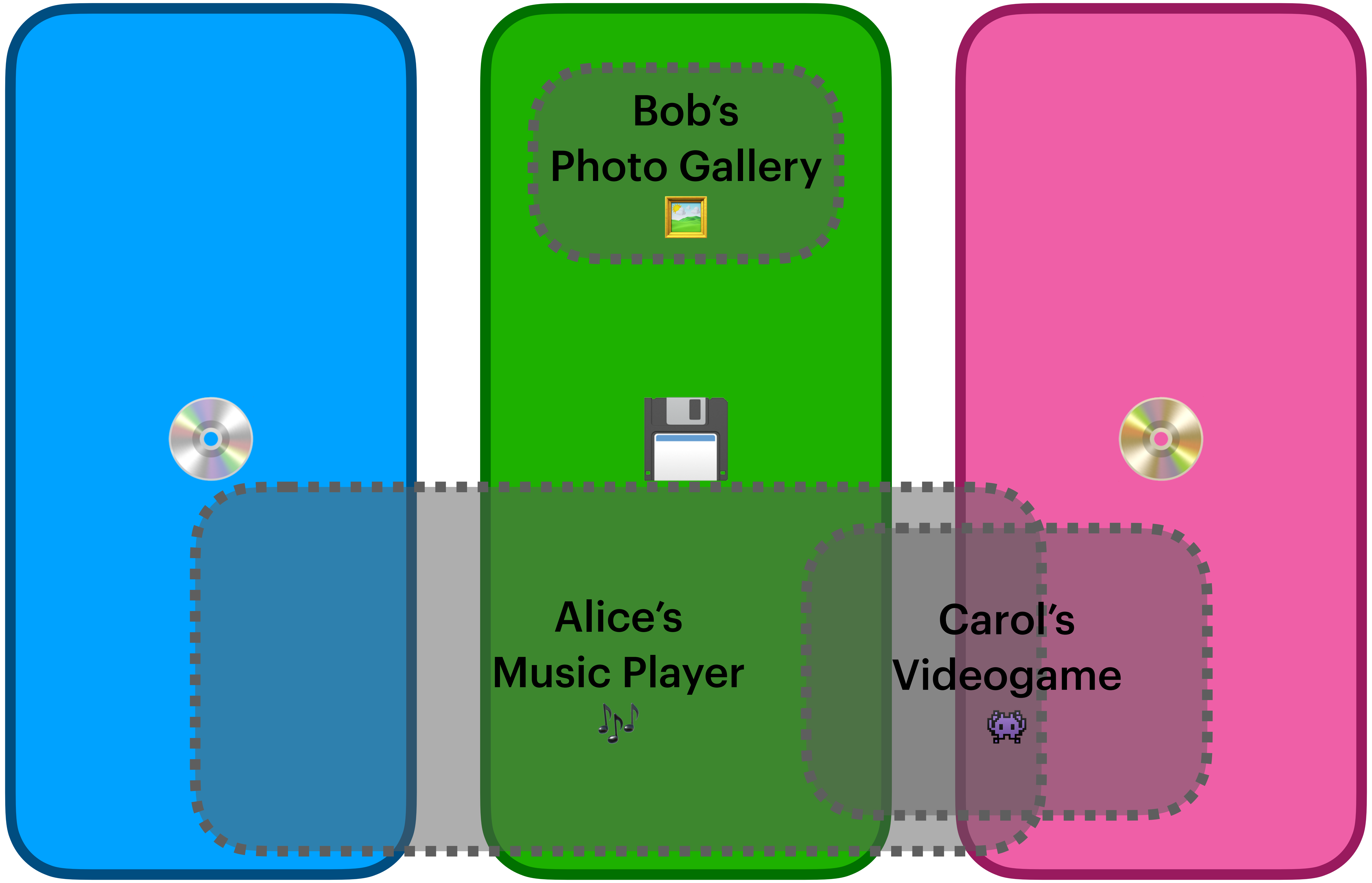
Dependencies & Integration



Problems 🪨🪐🌌

Data Behind Walls 🏰

Dependencies & Integration



Problems 🪨🪐🌌

Data Behind Walls 🏰

Inconsistency

Problems 🪨🪐🌌

Data Behind Walls 🏰

Inconsistency

- Even with FOSS!

Problems 🪨🪐🌌

Data Behind Walls 🏰

Inconsistency

- Even with FOSS!
- Annual migration to the latest hipster HTTP client
  - HTTPotion → HTTPoison → Hackney → Tesla → Finch → Req

Problems 🪨🪐🌌

Data Behind Walls 🏰

Problems 🪨🪐🌌

Data Behind Walls 🏰

> If people in a few hundred years from now want to see what their ancestors wrote, what will they find, ***a mess of badly formatted crap?!***

— Joe Armstrong, Why Markdown Sucks

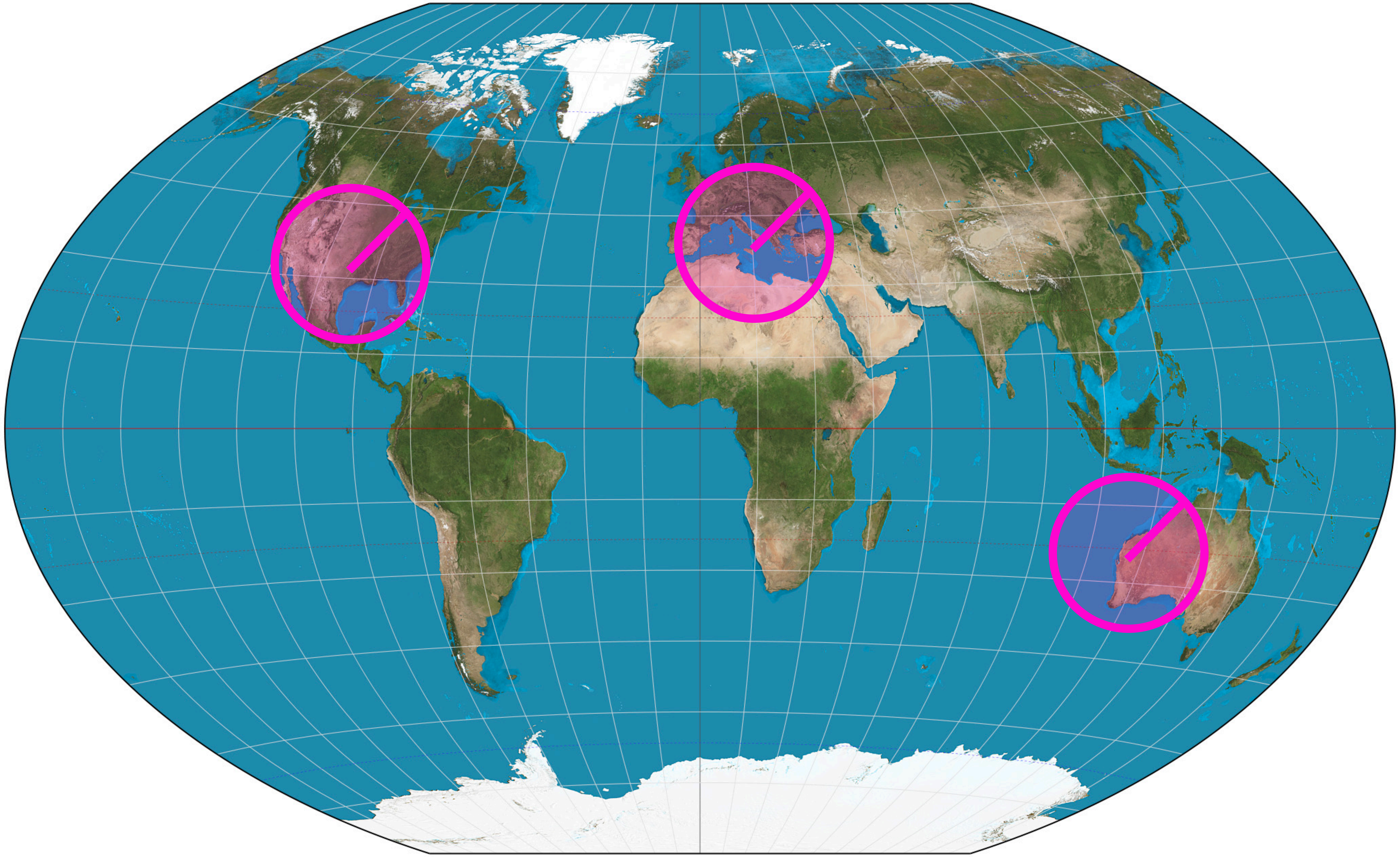
Cause & Effect

Mental Framework

🏗️🧠

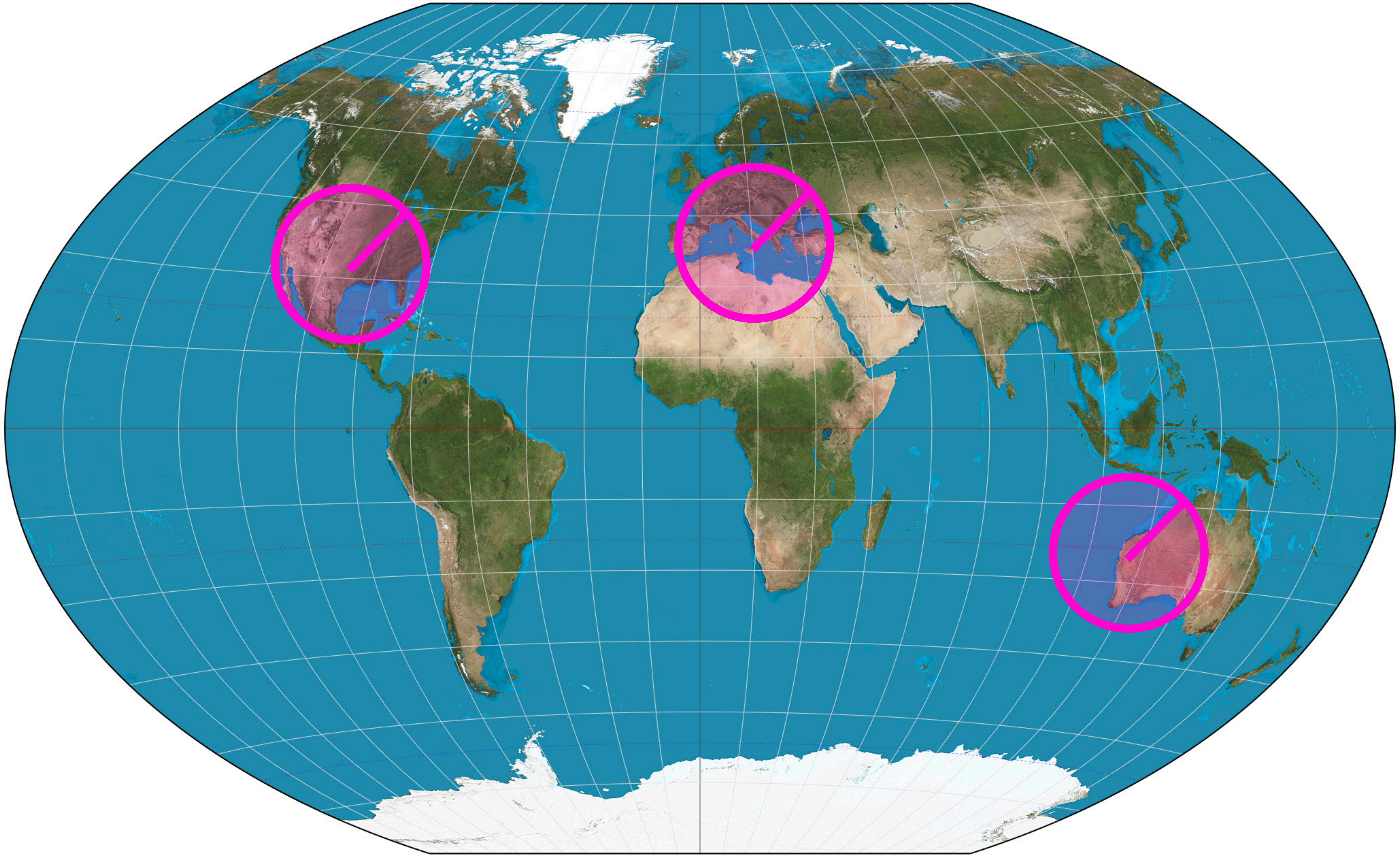
Mental Framework 🏗️🧠

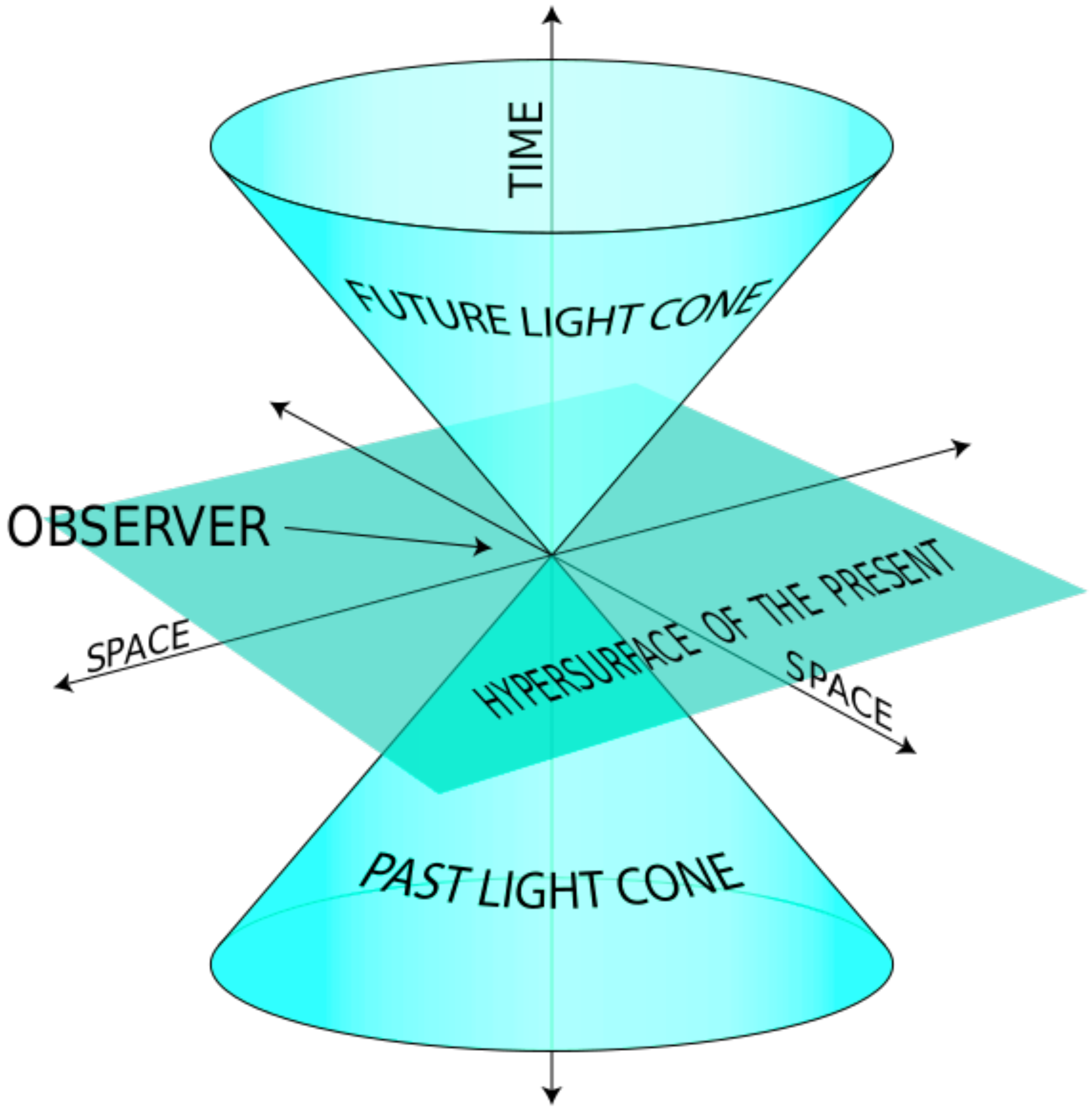
Causal Islands 🏖️🏝️



Mental Framework 🏗️🧠

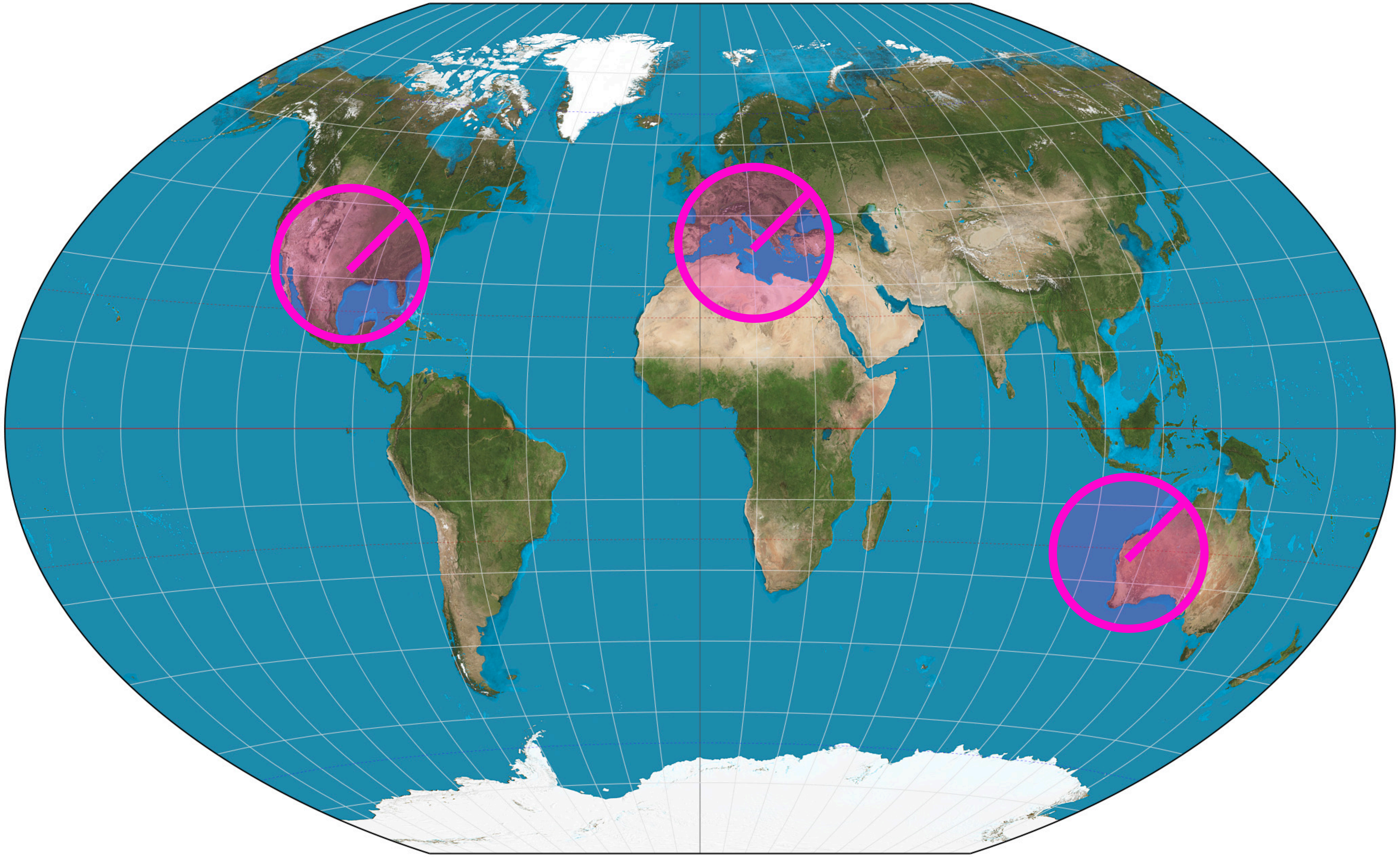
Causal Islands 🏖️🏝️

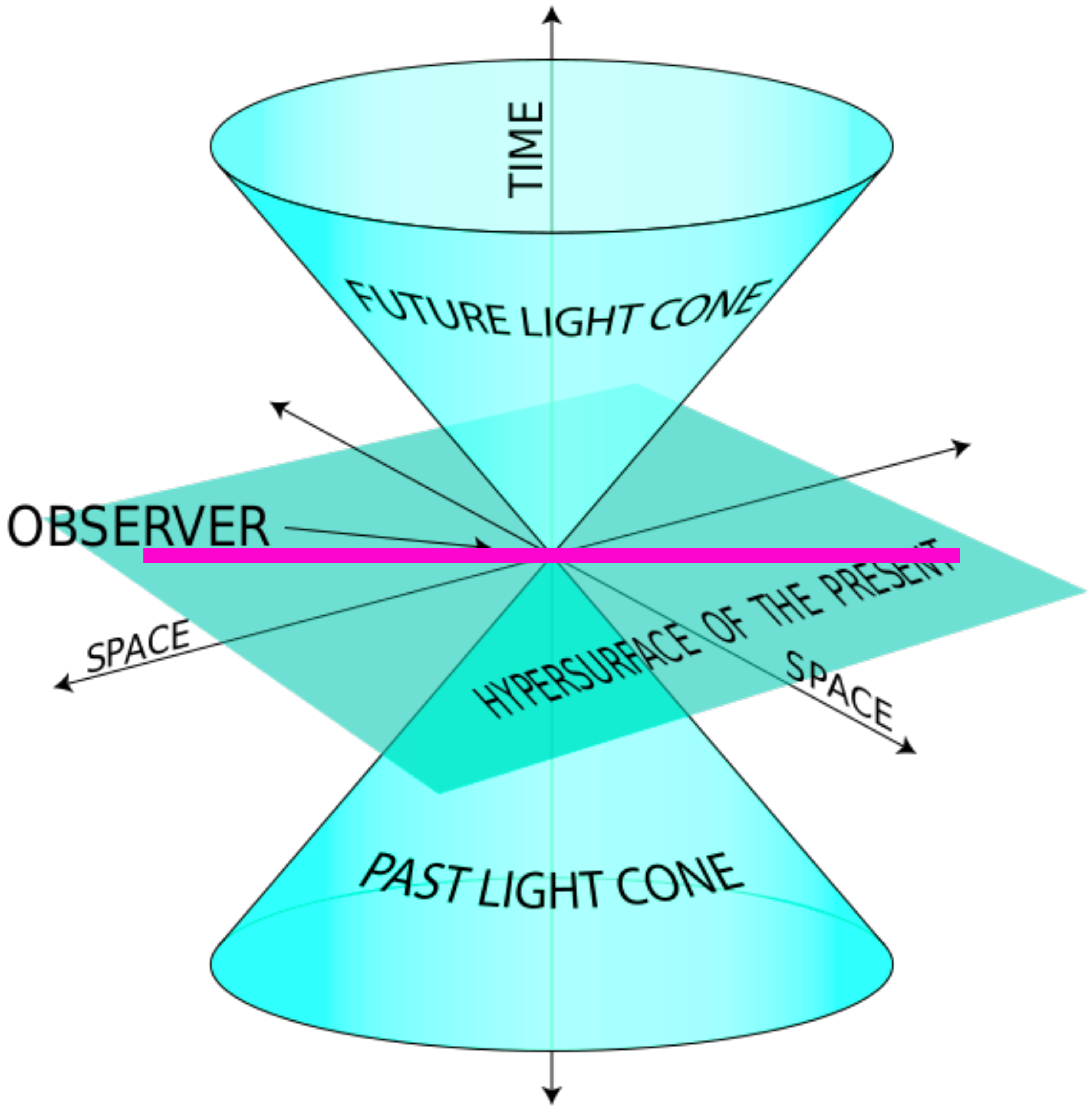




Mental Framework 🏗️🧠

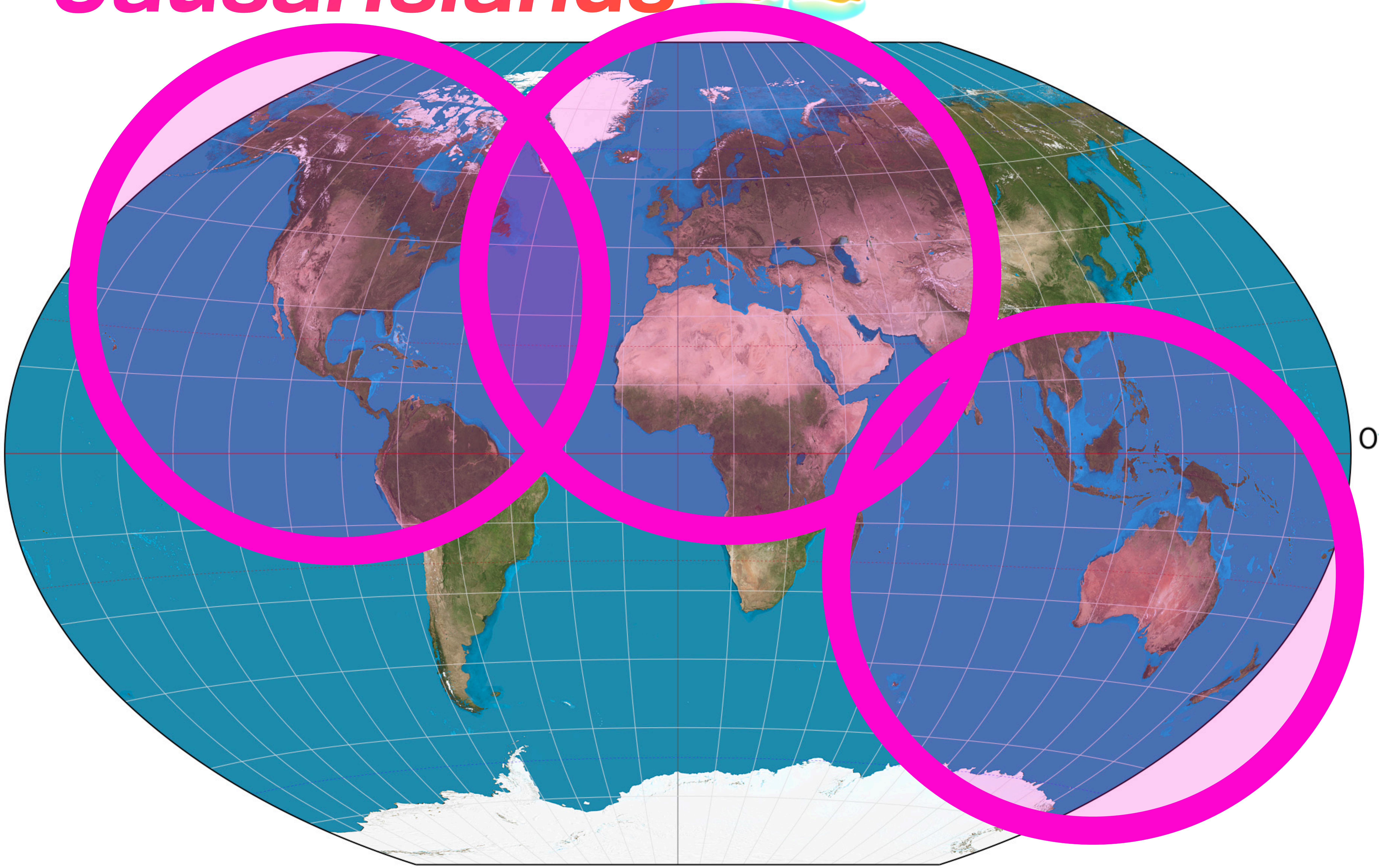
Causal Islands 🏖️🏝️

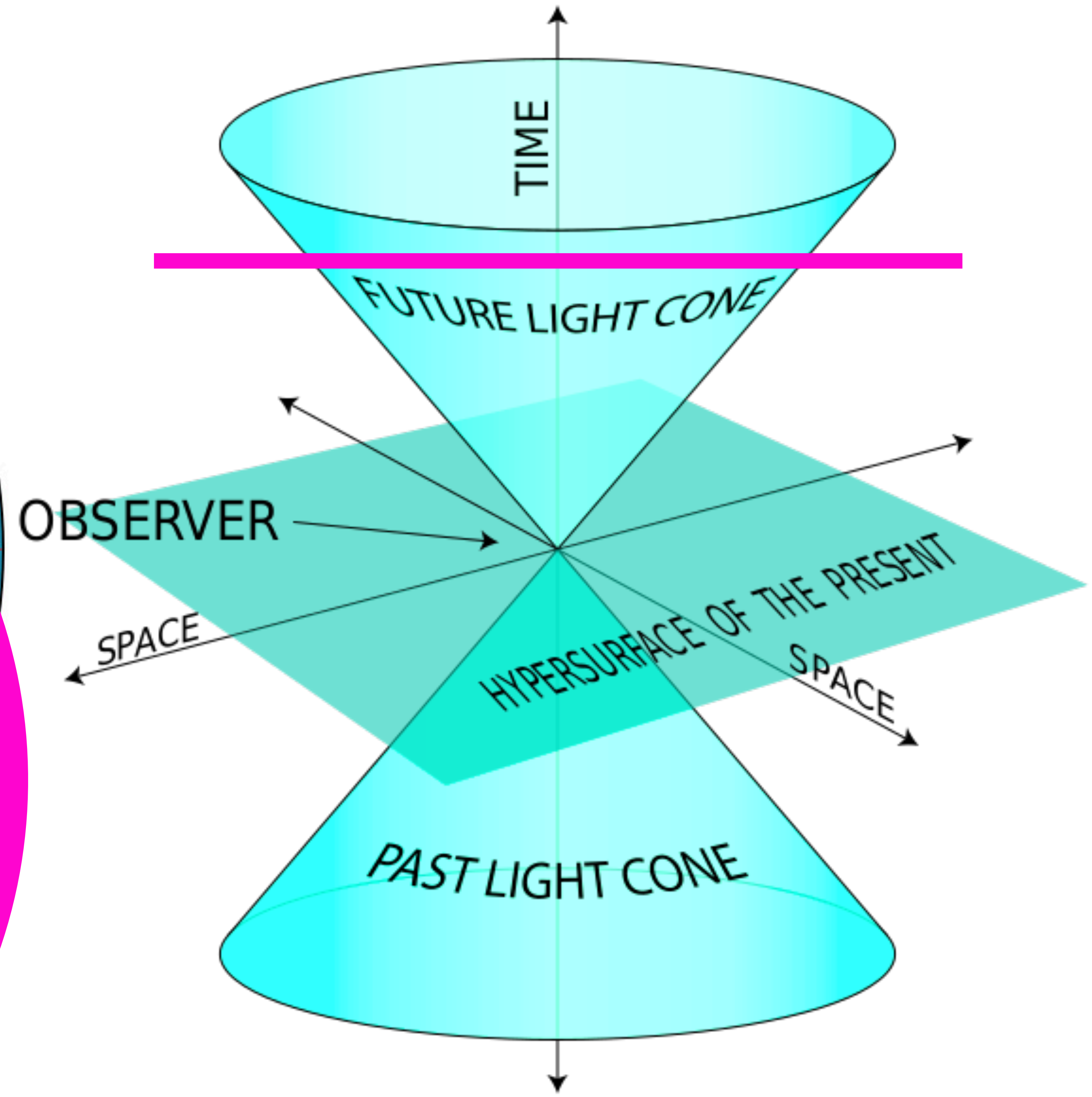




Mental Framework 🏗️🧠

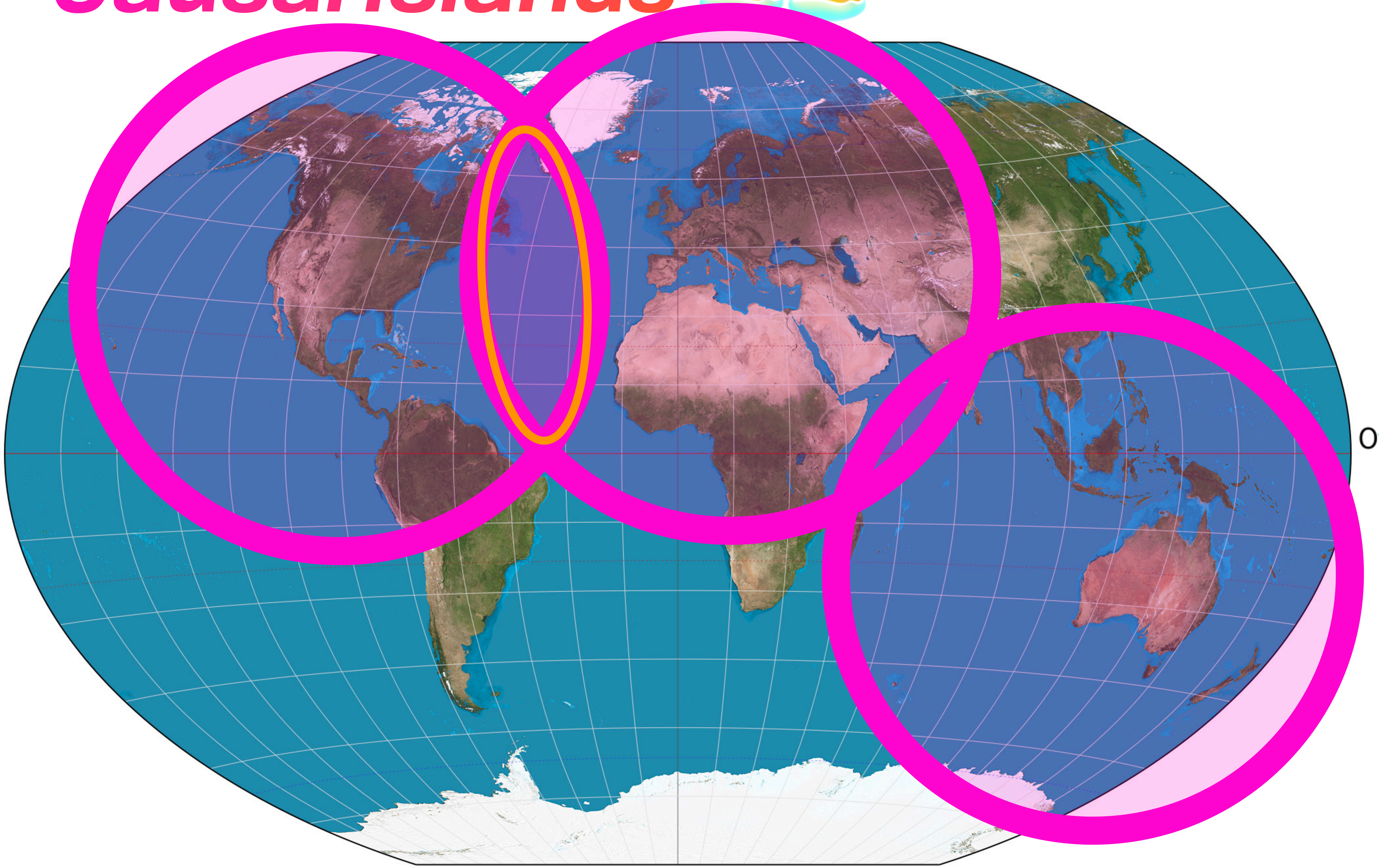
Causal Islands ⛱️🏝️

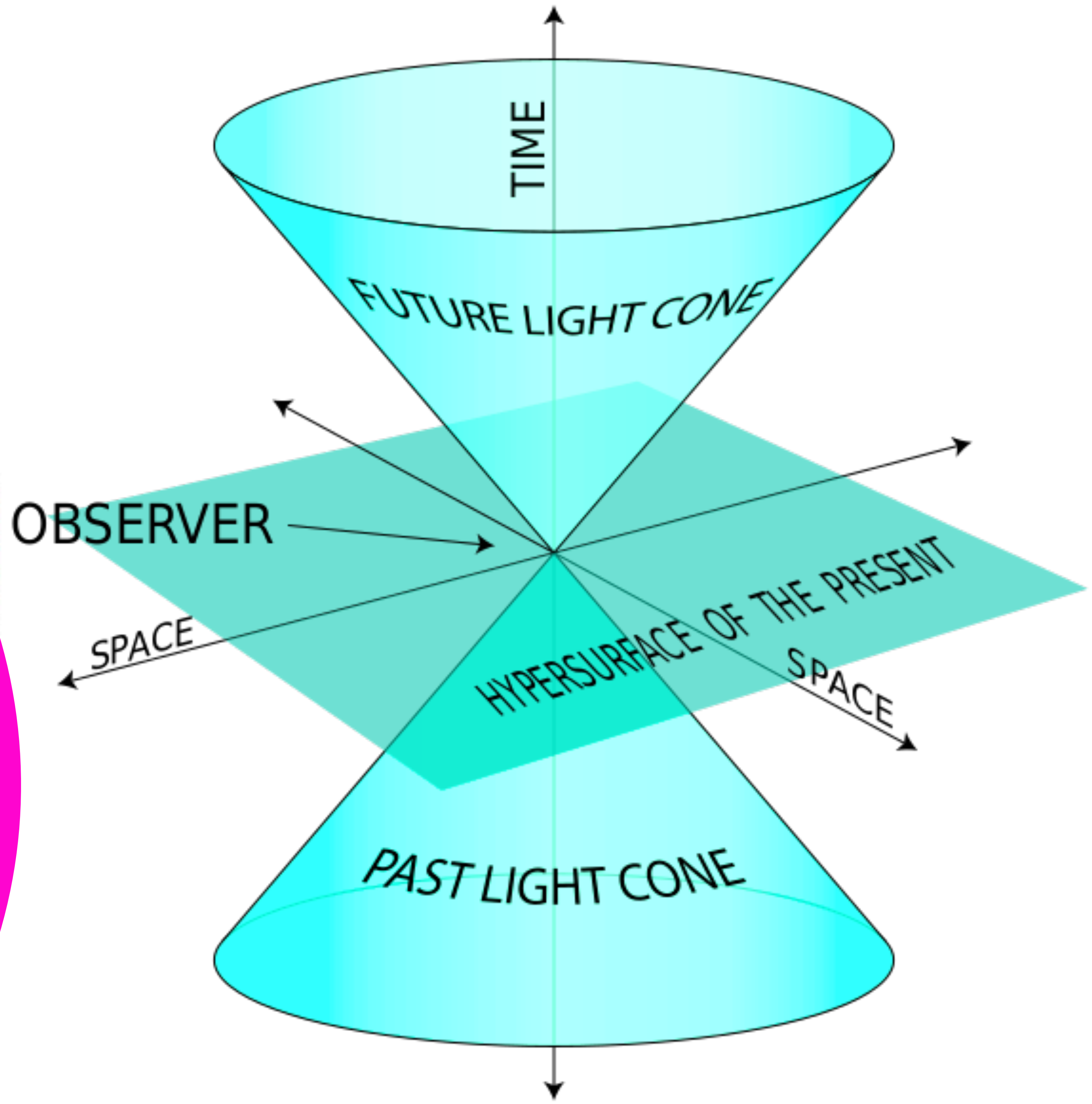




Mental Framework 🏗️🧠

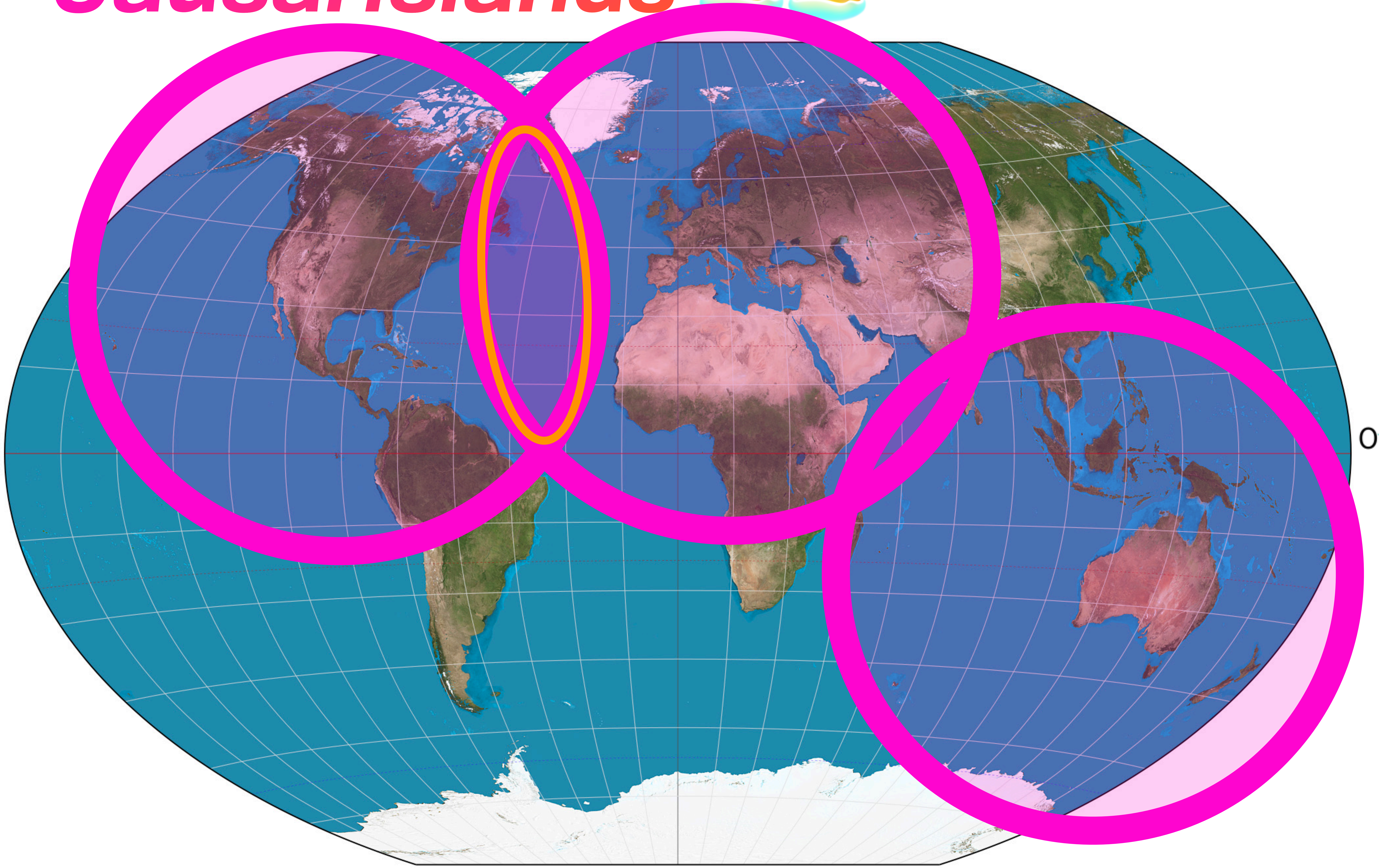
Causal Islands 🏖️🏝️

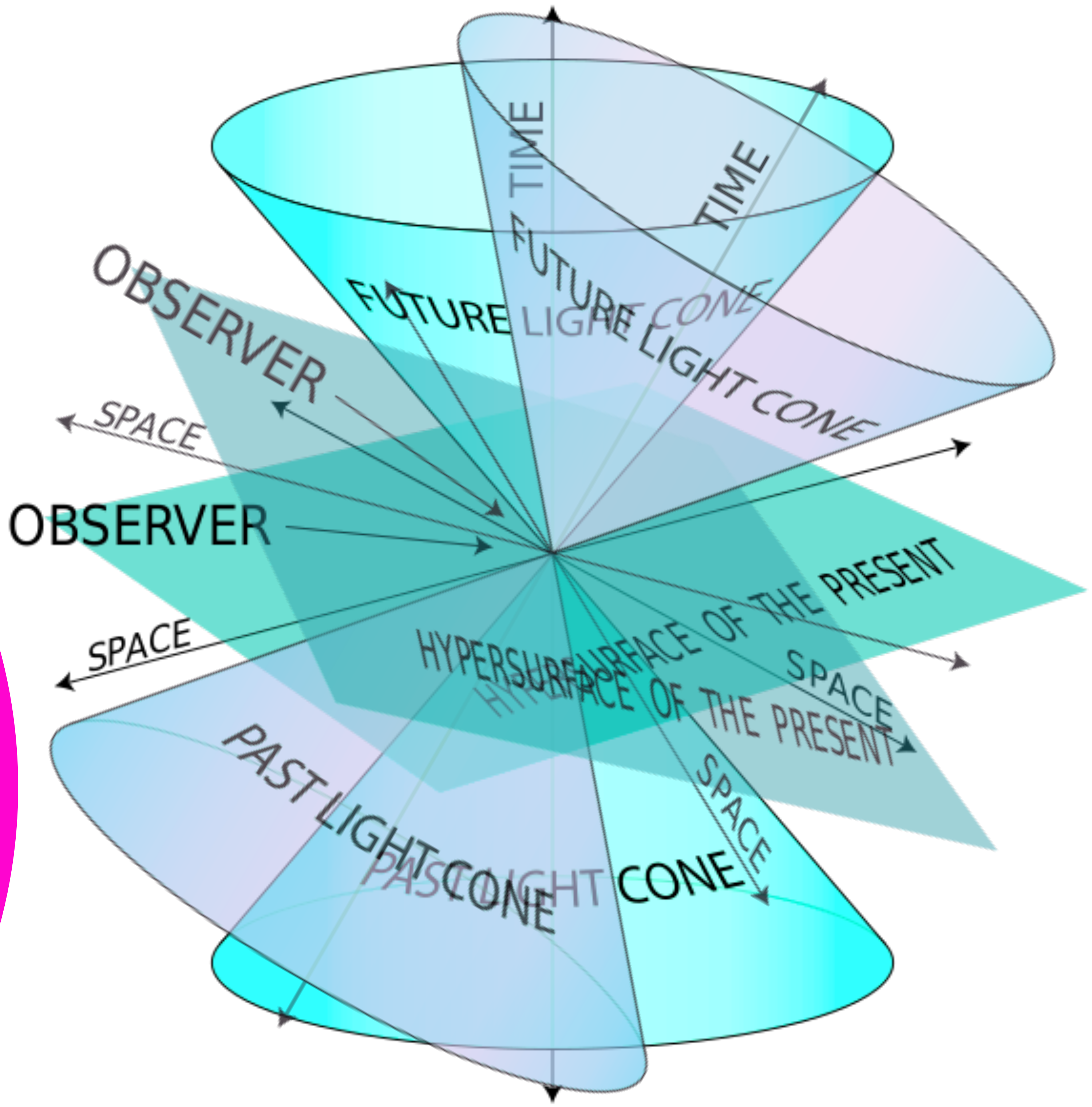




Mental Framework 🏗️🧠

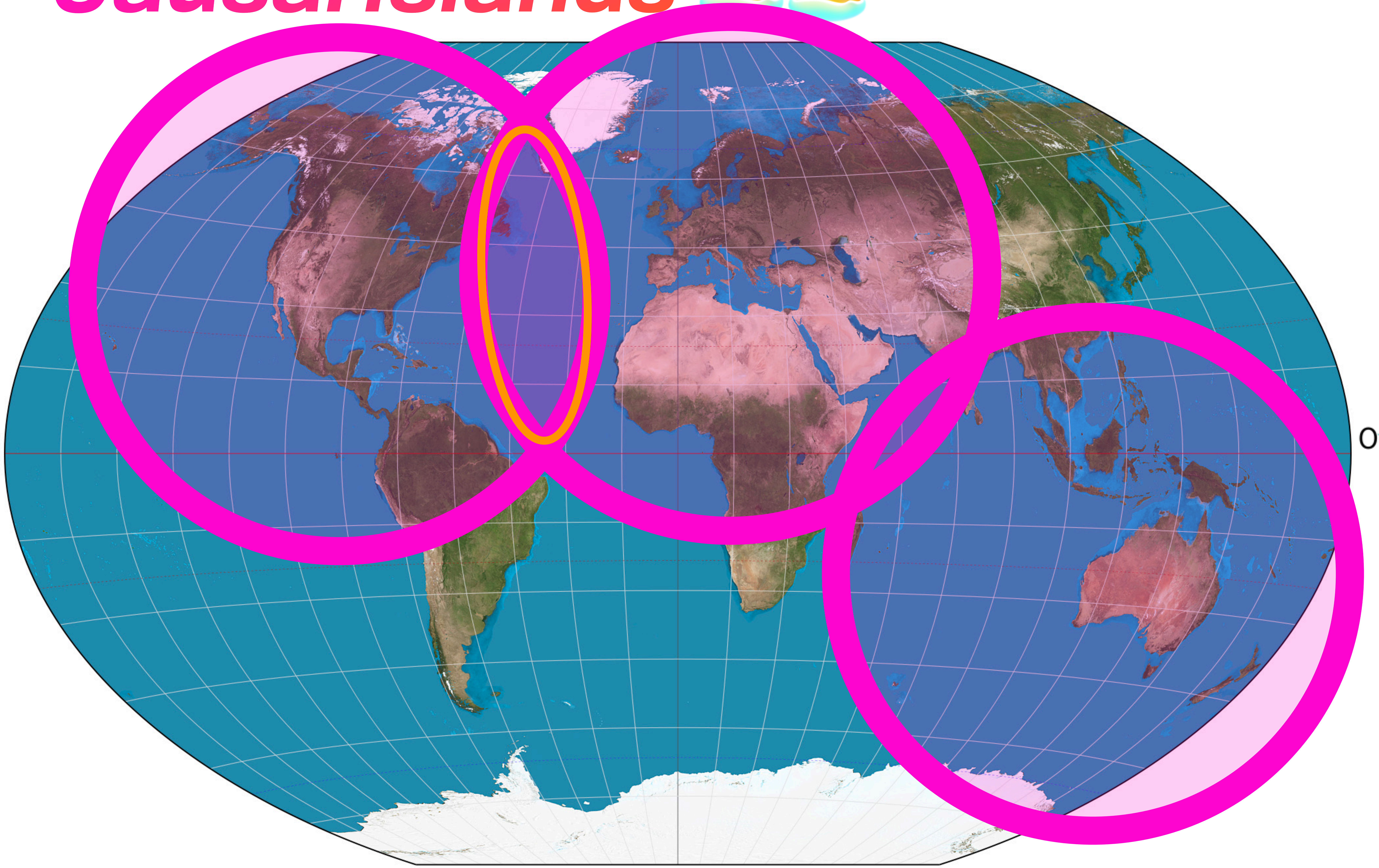
Causal Islands 🏖️🏝️

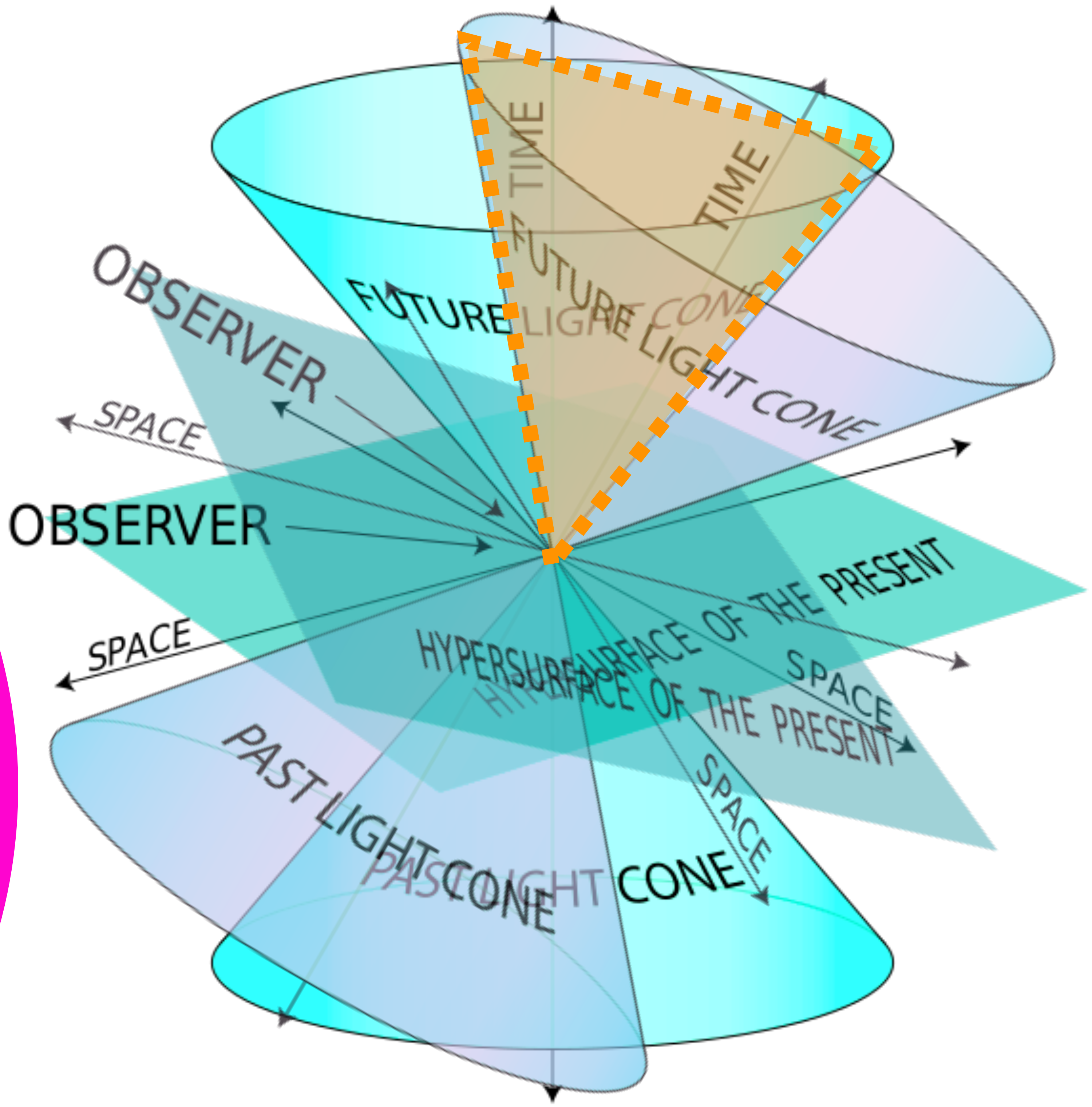




Mental Framework 🏗️🧠

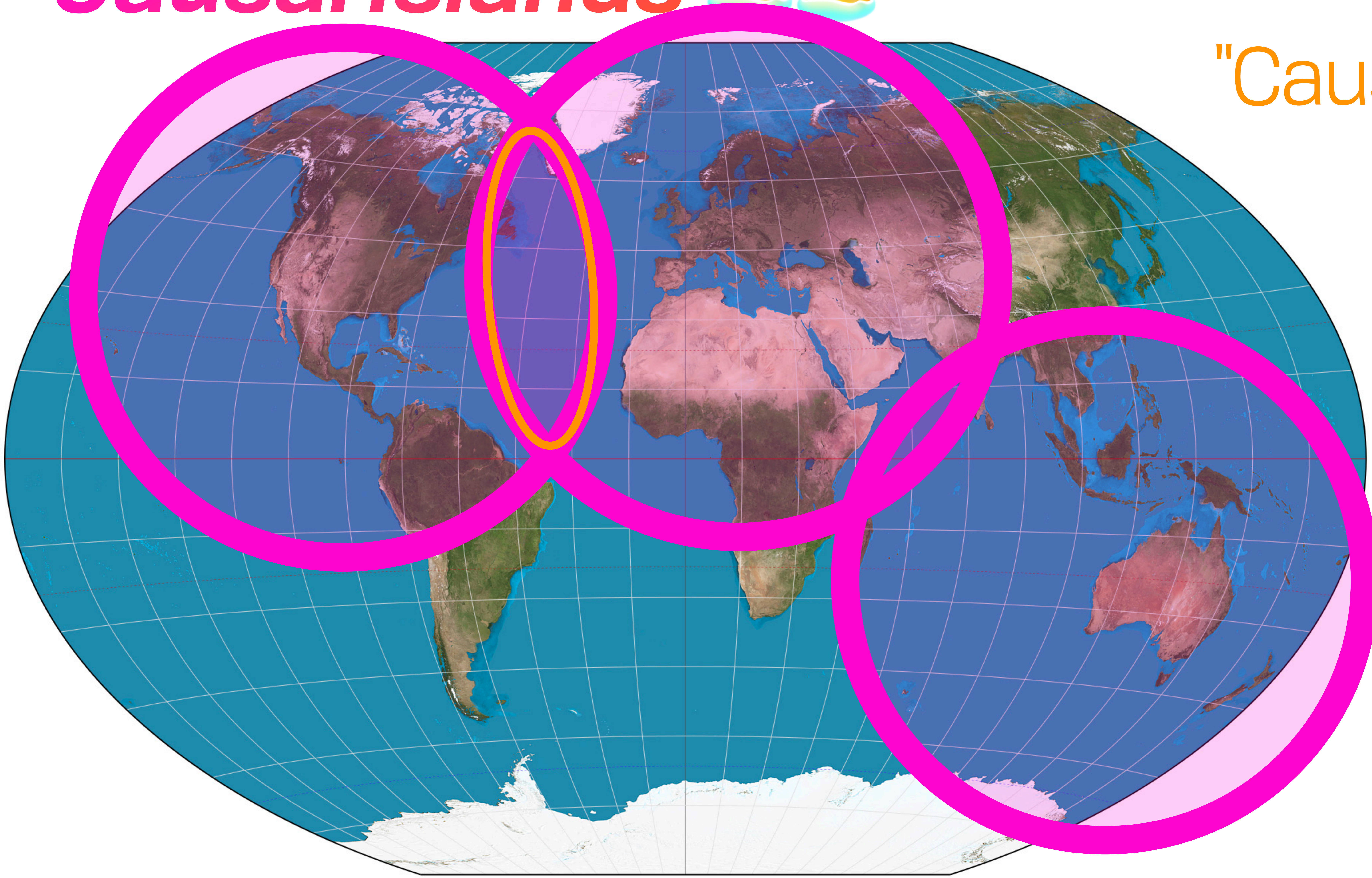
Causal Islands 🏖️🏝️



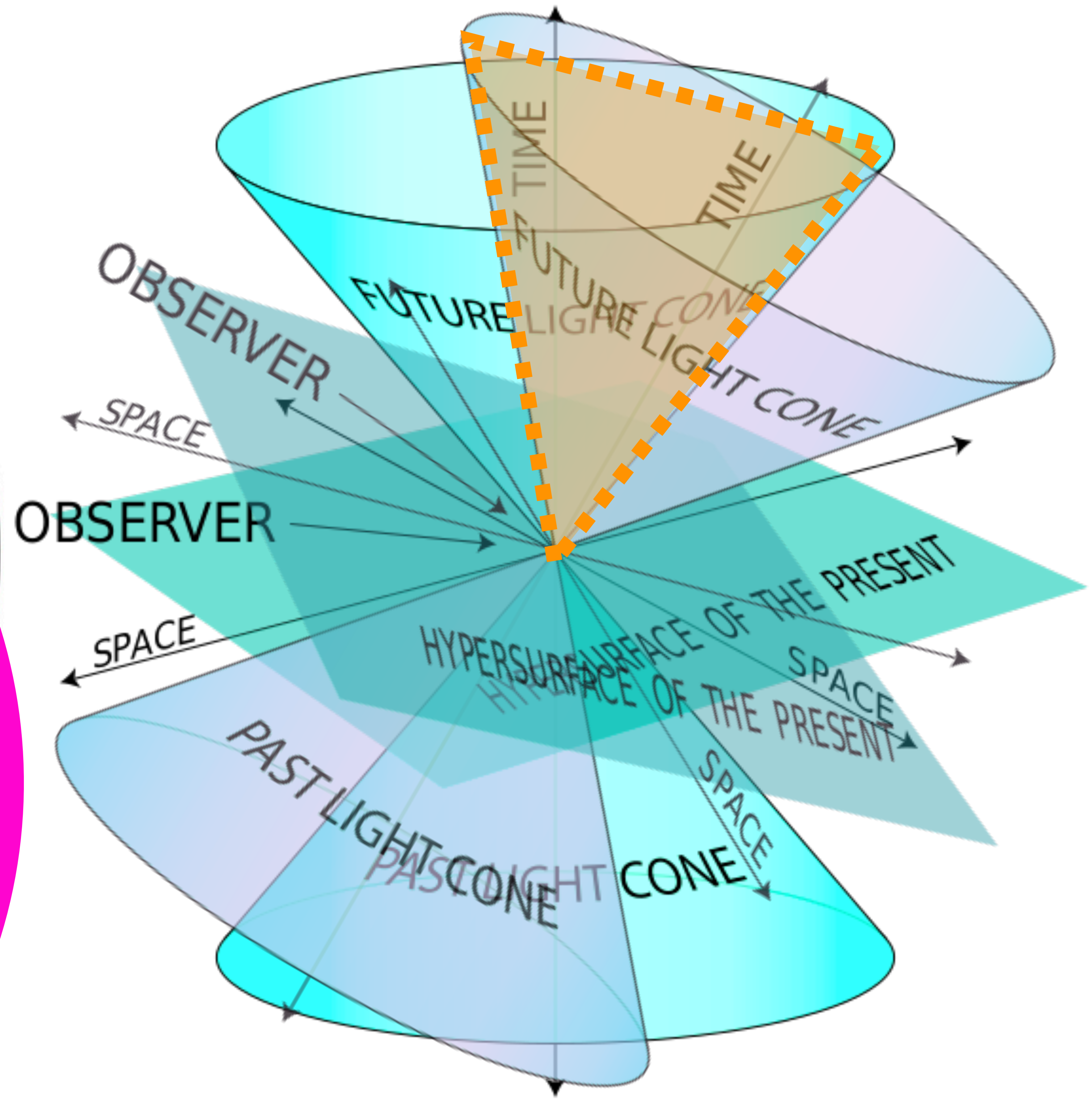


Mental Framework 🏗️🧠

Causal Islands 🏖️🏝️



"Causal Subjectivity"



Mental Framework 🏗️🧠

Mental Framework 🏗️🧠

What is the family of problems that can be consistently computed in a distributed fashion ***without coordination***, and what problems lie outside that family?

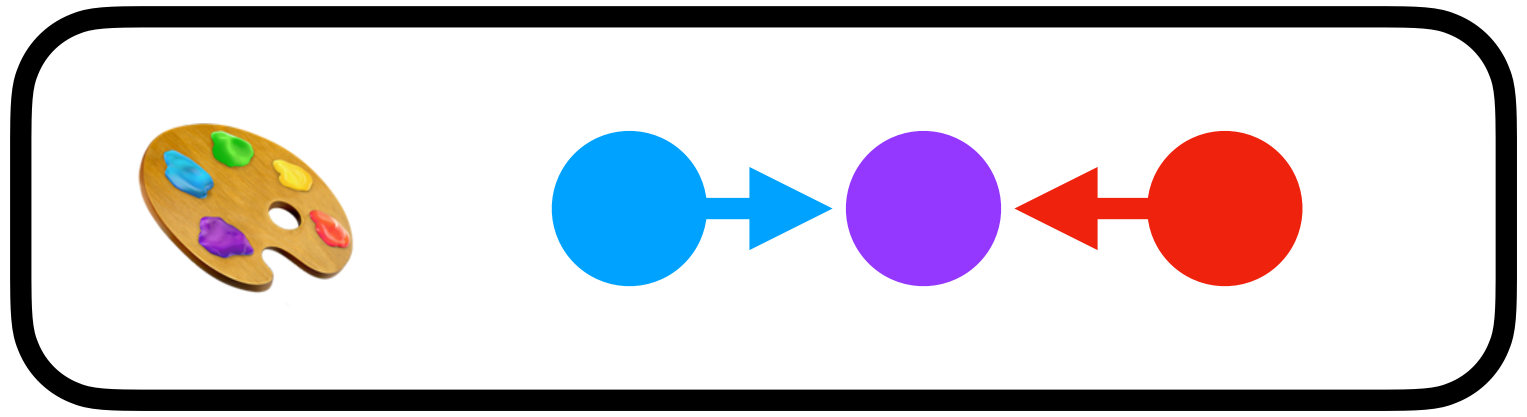
—Hellerstein & Alvaro, Keeping CALM: When Distributed Consistency is Easy

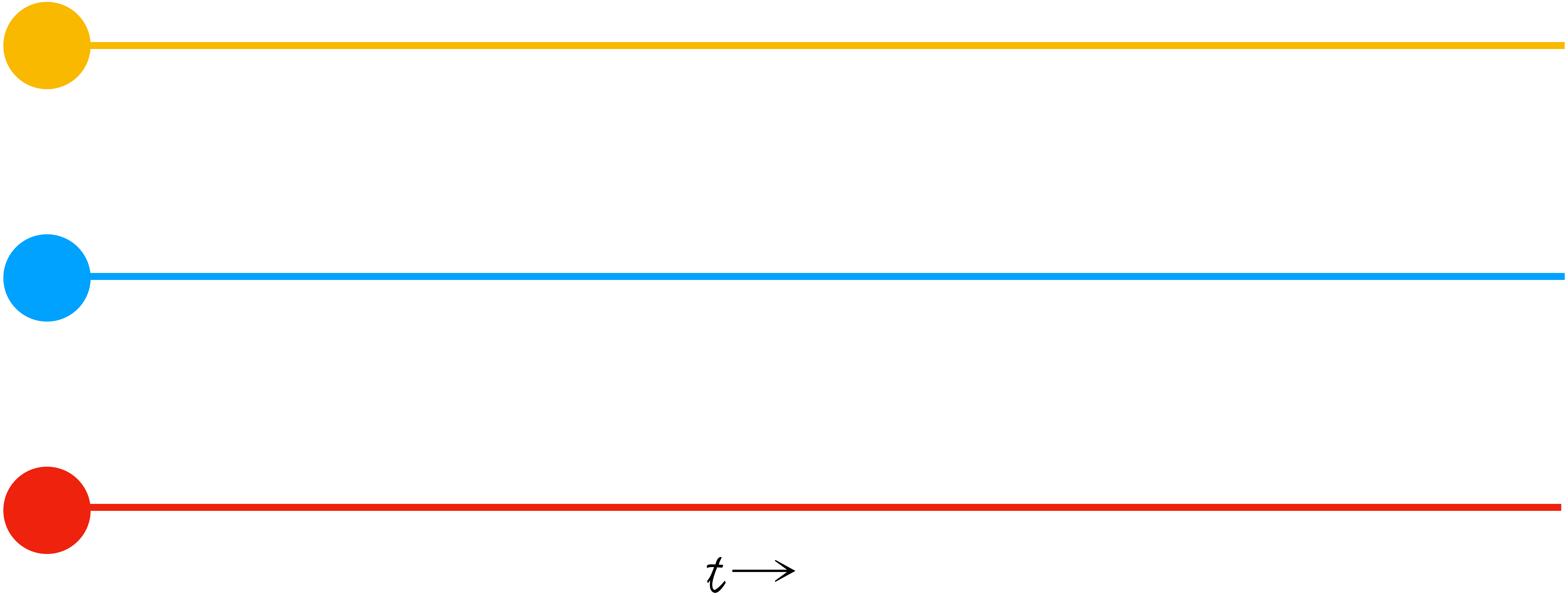
Mental Framework 🏗️🧠

Gossiping Out of Order 🙊

Mental Framework 🏗️🧠

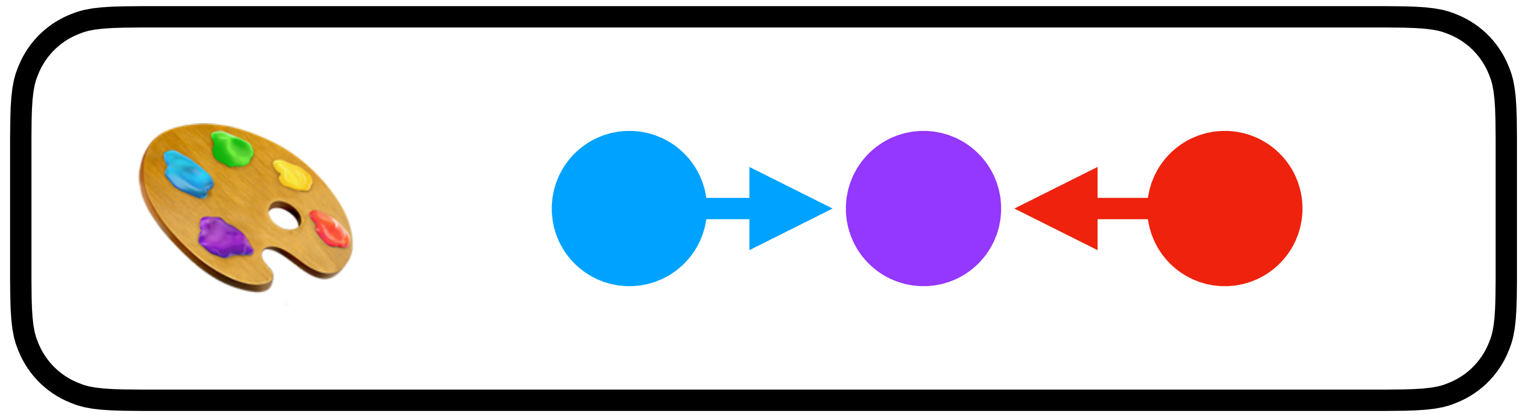
Gossiping Out of Order 🙊

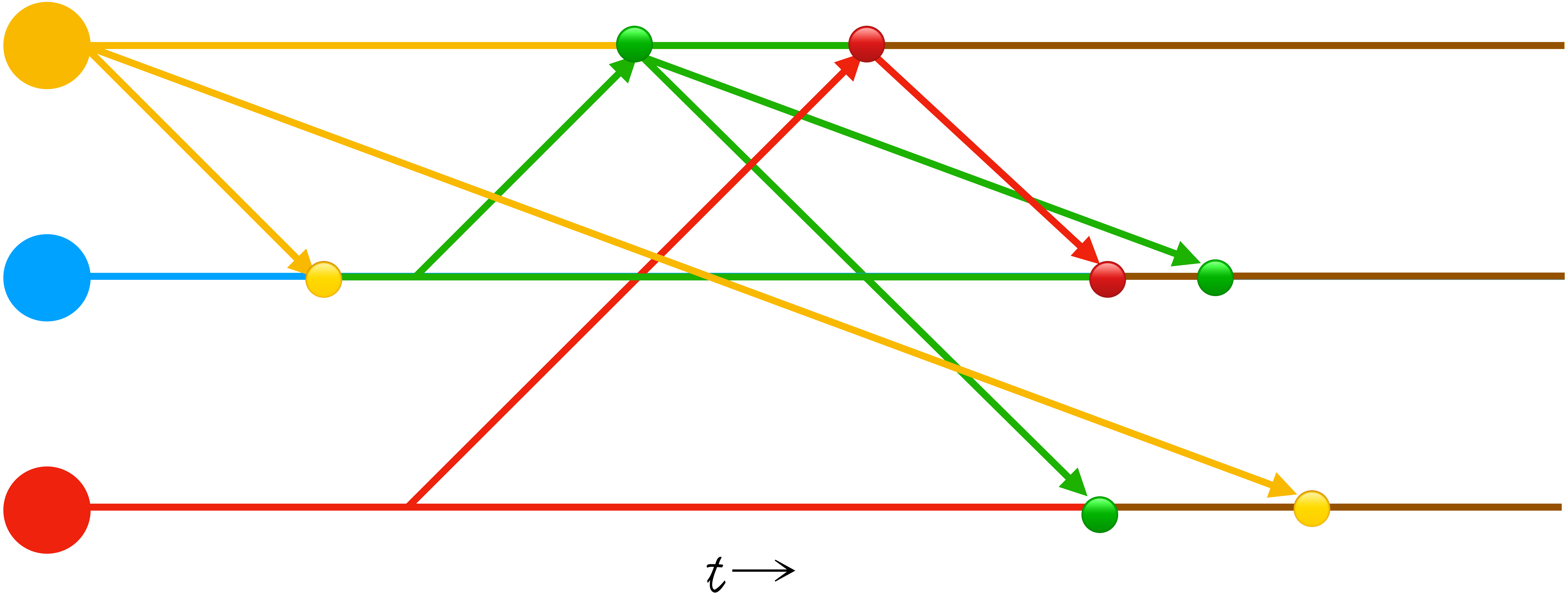




Mental Framework 🏗️🧠

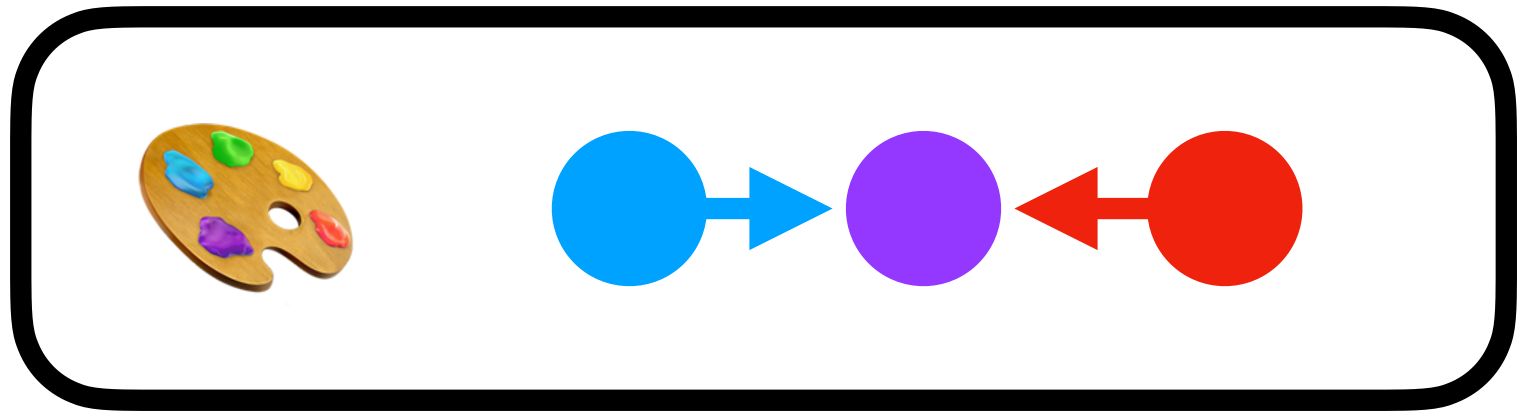
Gossiping Out of Order 🙊

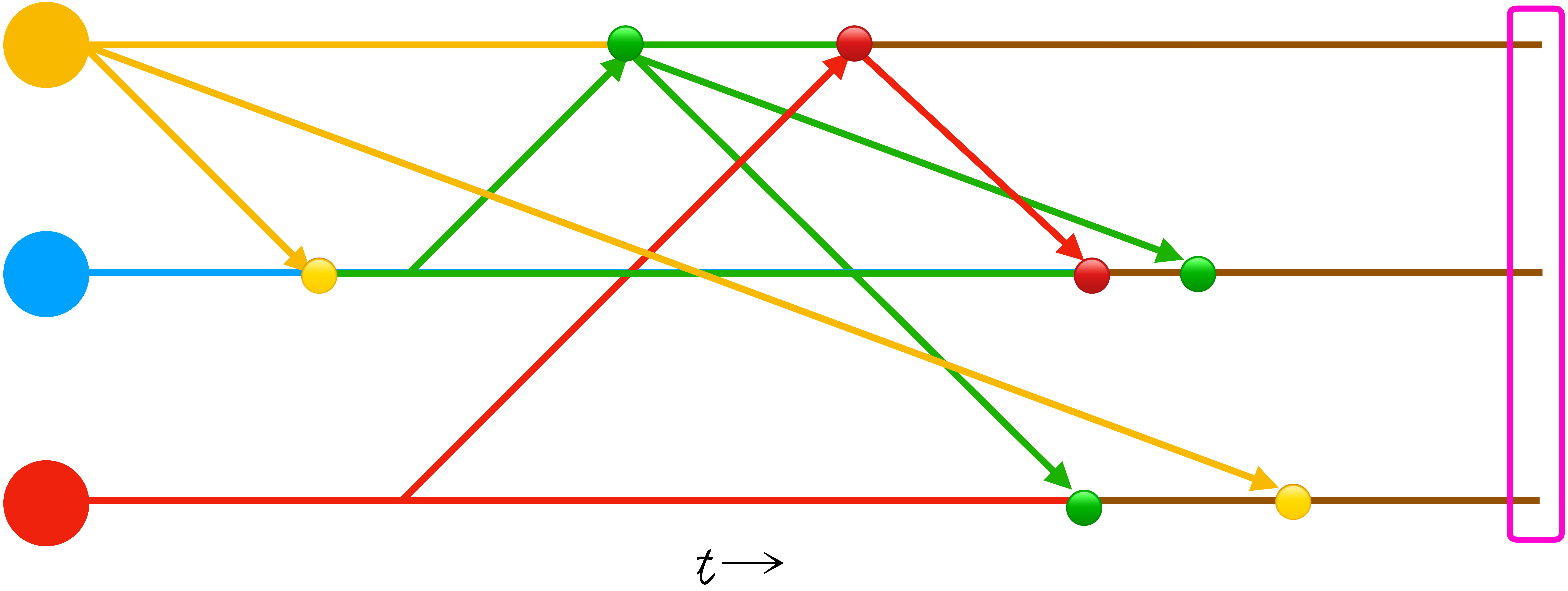




Mental Framework 🏗️🧠

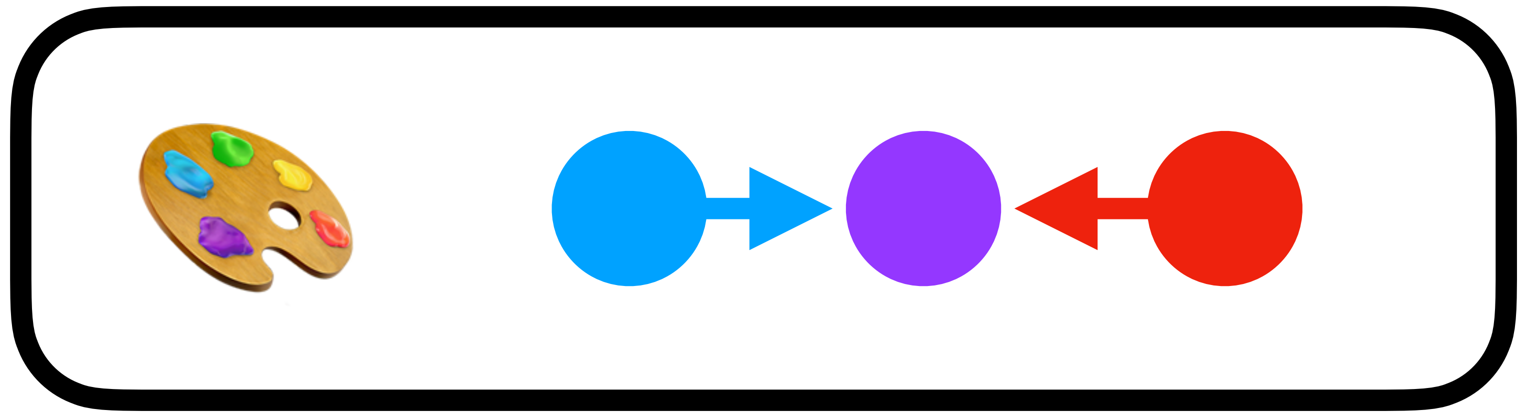
Gossiping Out of Order 🙊

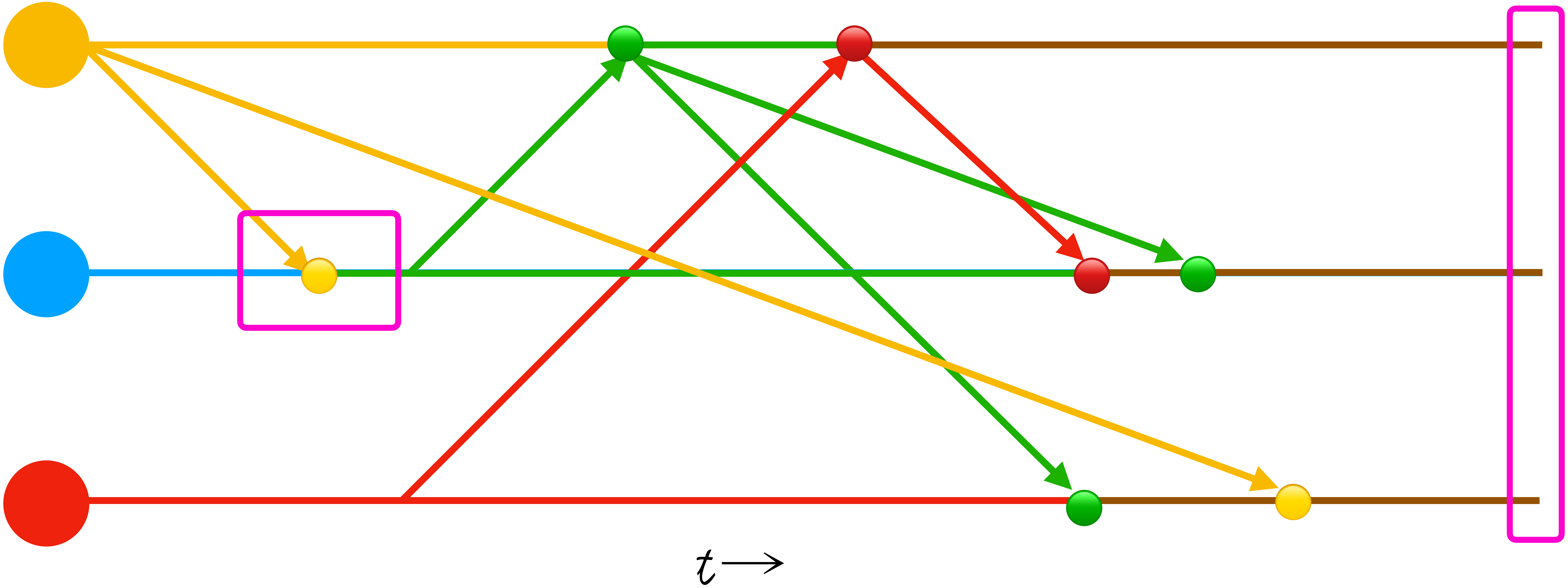




Mental Framework 🏗️🧠

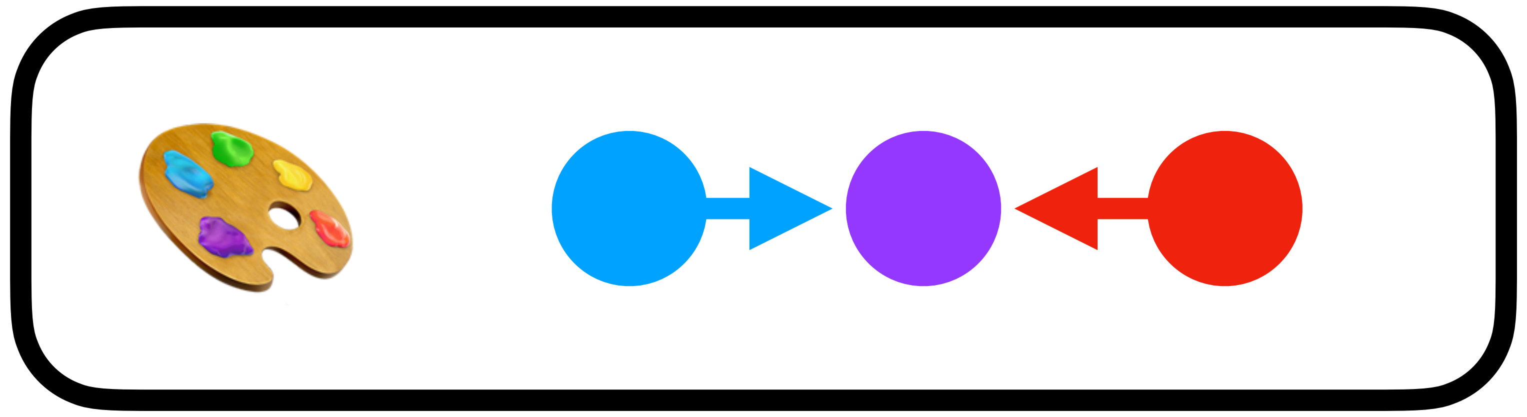
Gossiping Out of Order 🙊

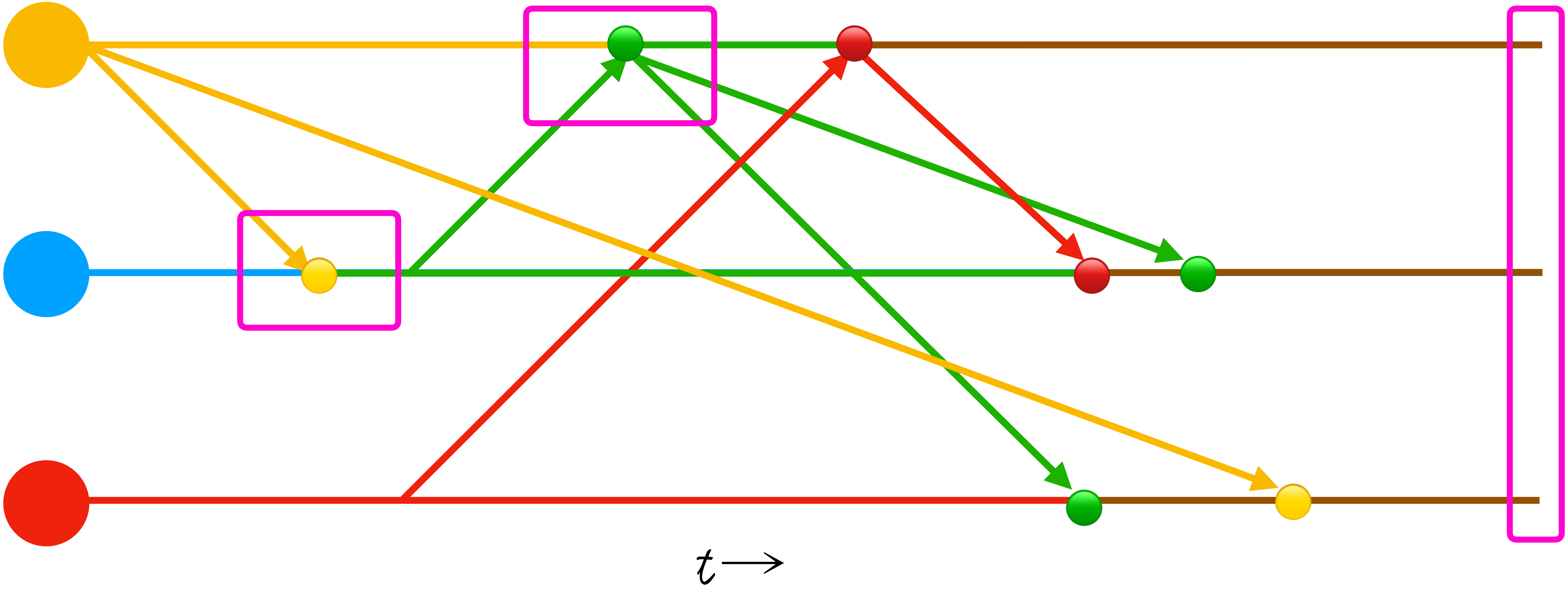




Mental Framework 🏗️🧠

Gossiping Out of Order 🙊

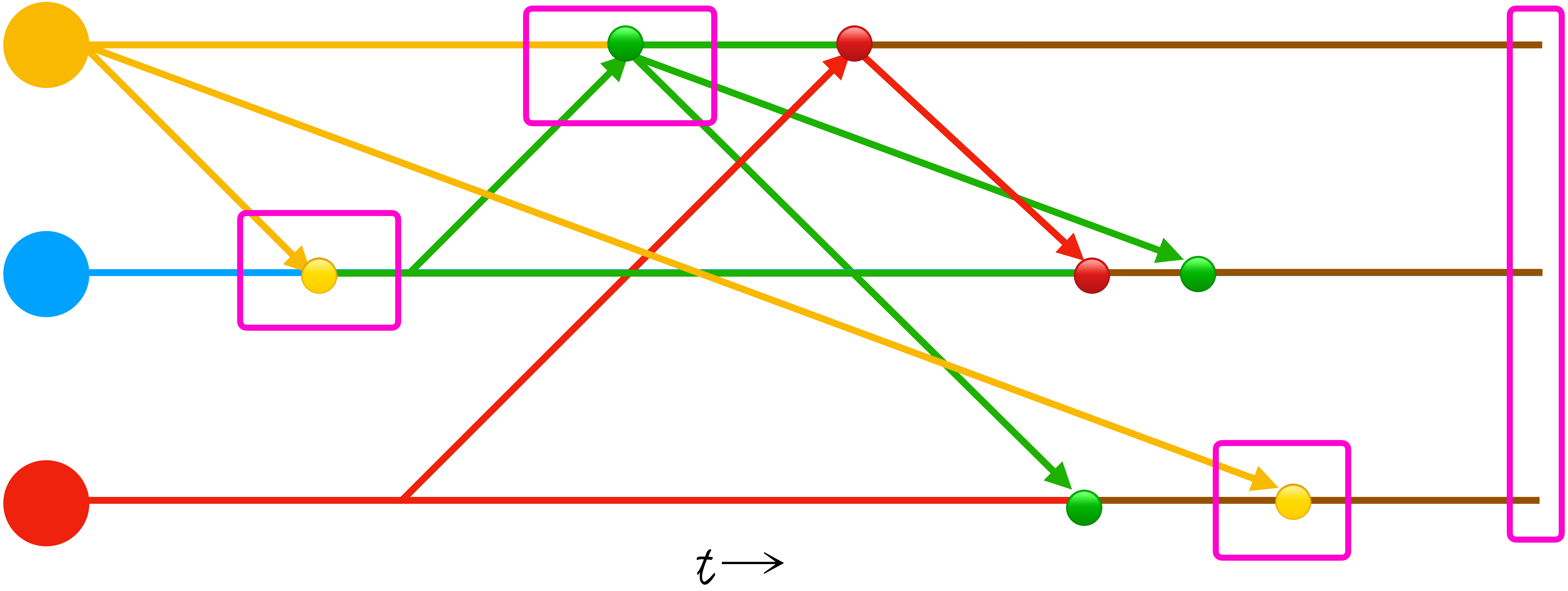




Mental Framework 🏗️🧠

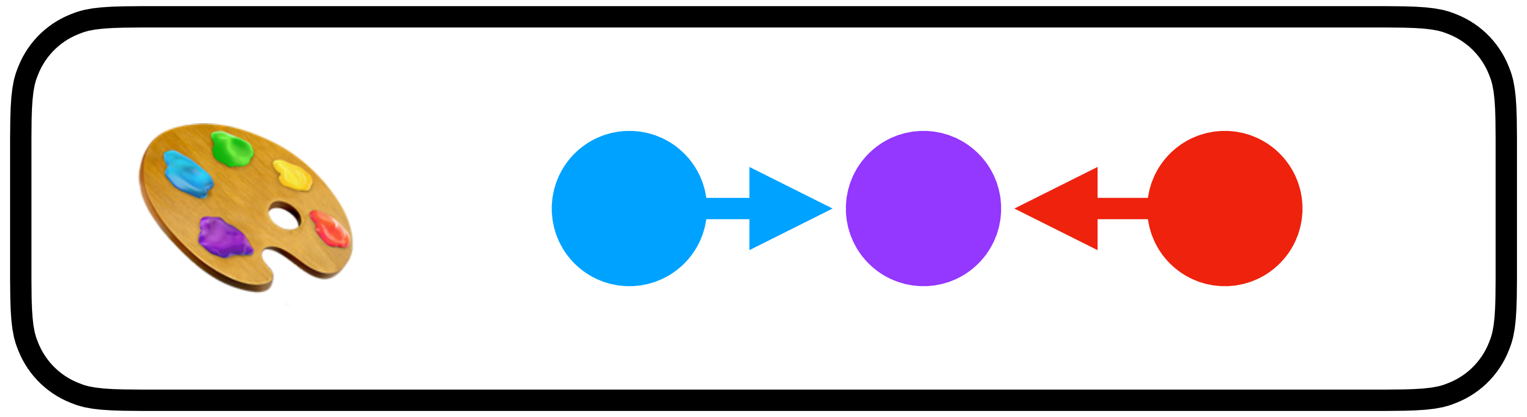
Gossiping Out of Order 🙊

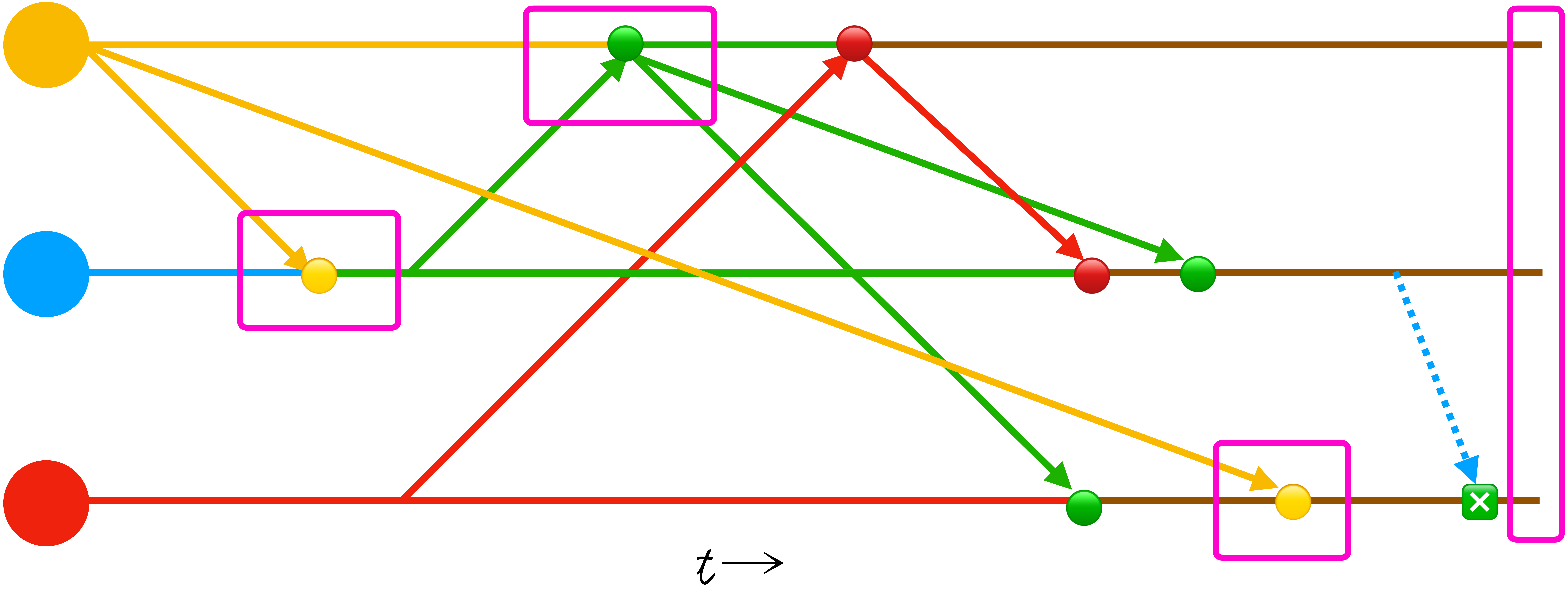
🎨 ● → ● ← ●



Mental Framework 🏗️🧠

Gossiping Out of Order 🙊



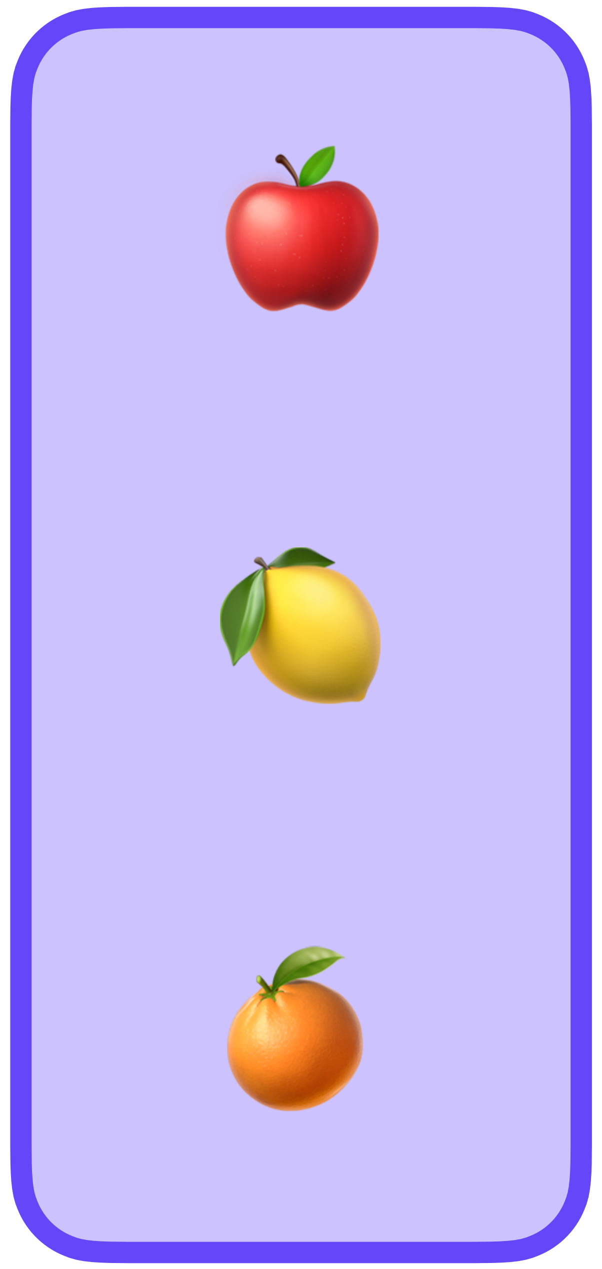


Mental Framework 🏗️🧠

Monotone Functions

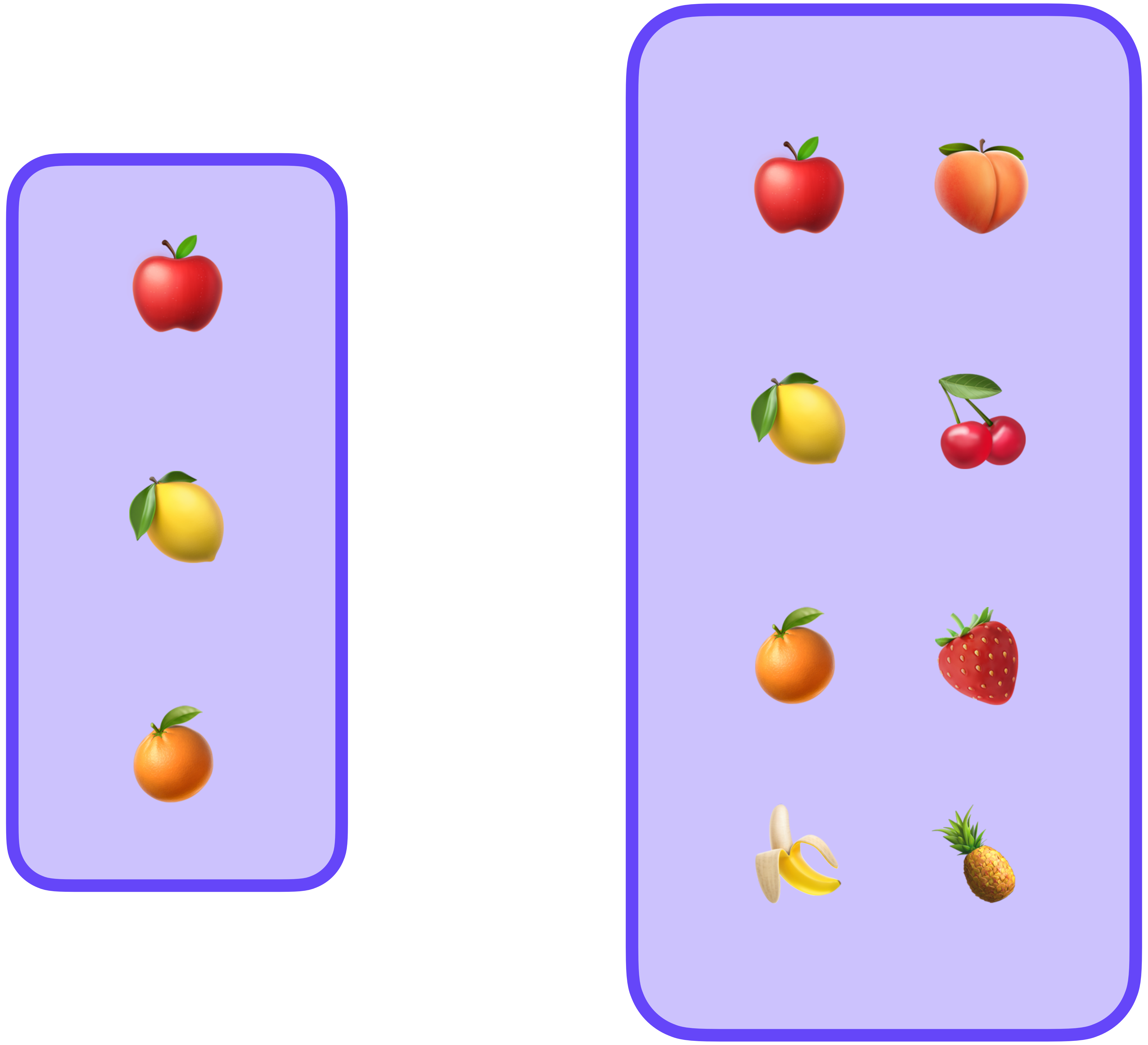
Mental Framework 🏗️🧠

Monotone Functions



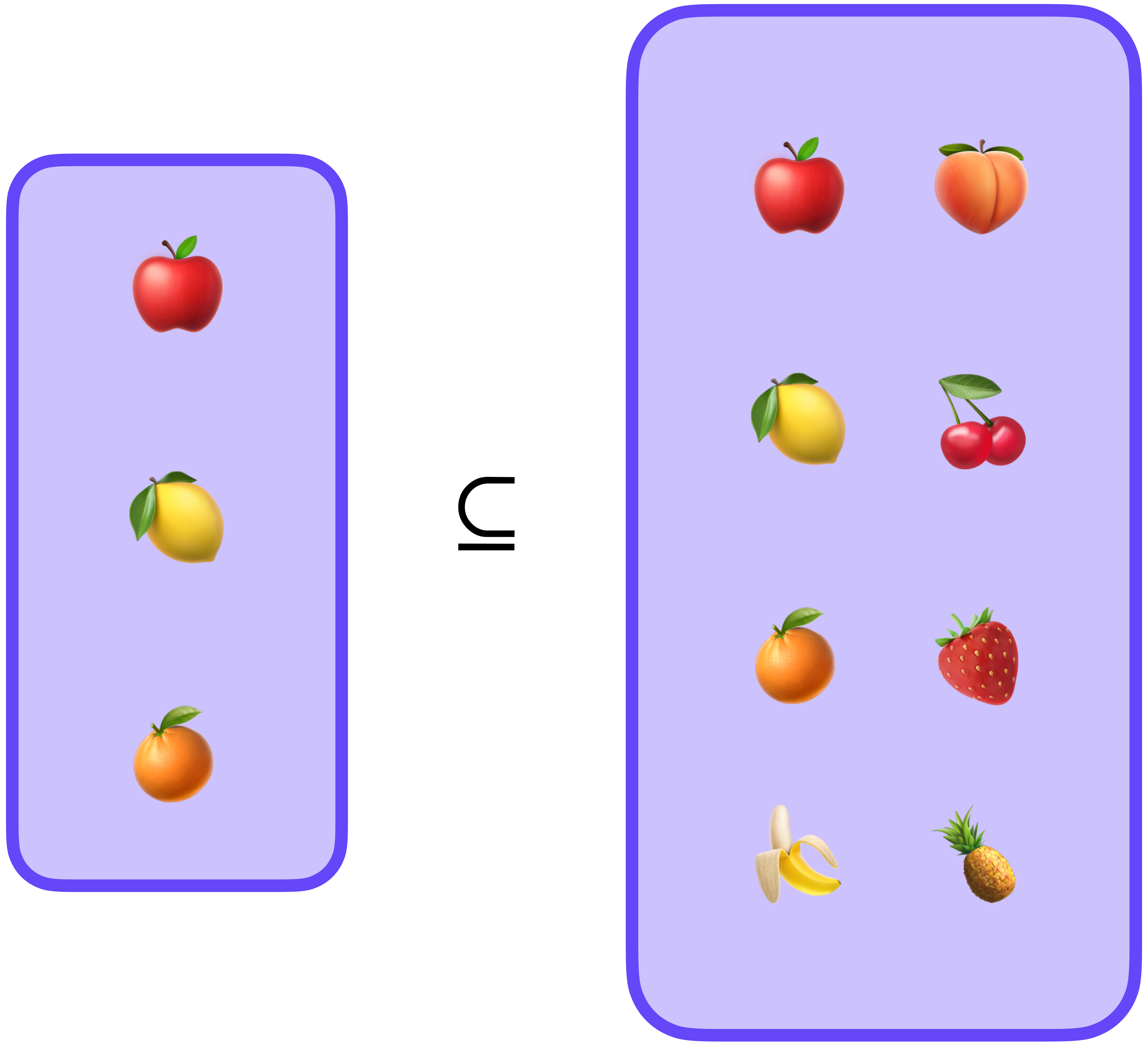
Mental Framework 🏗️🧠

Monotone Functions



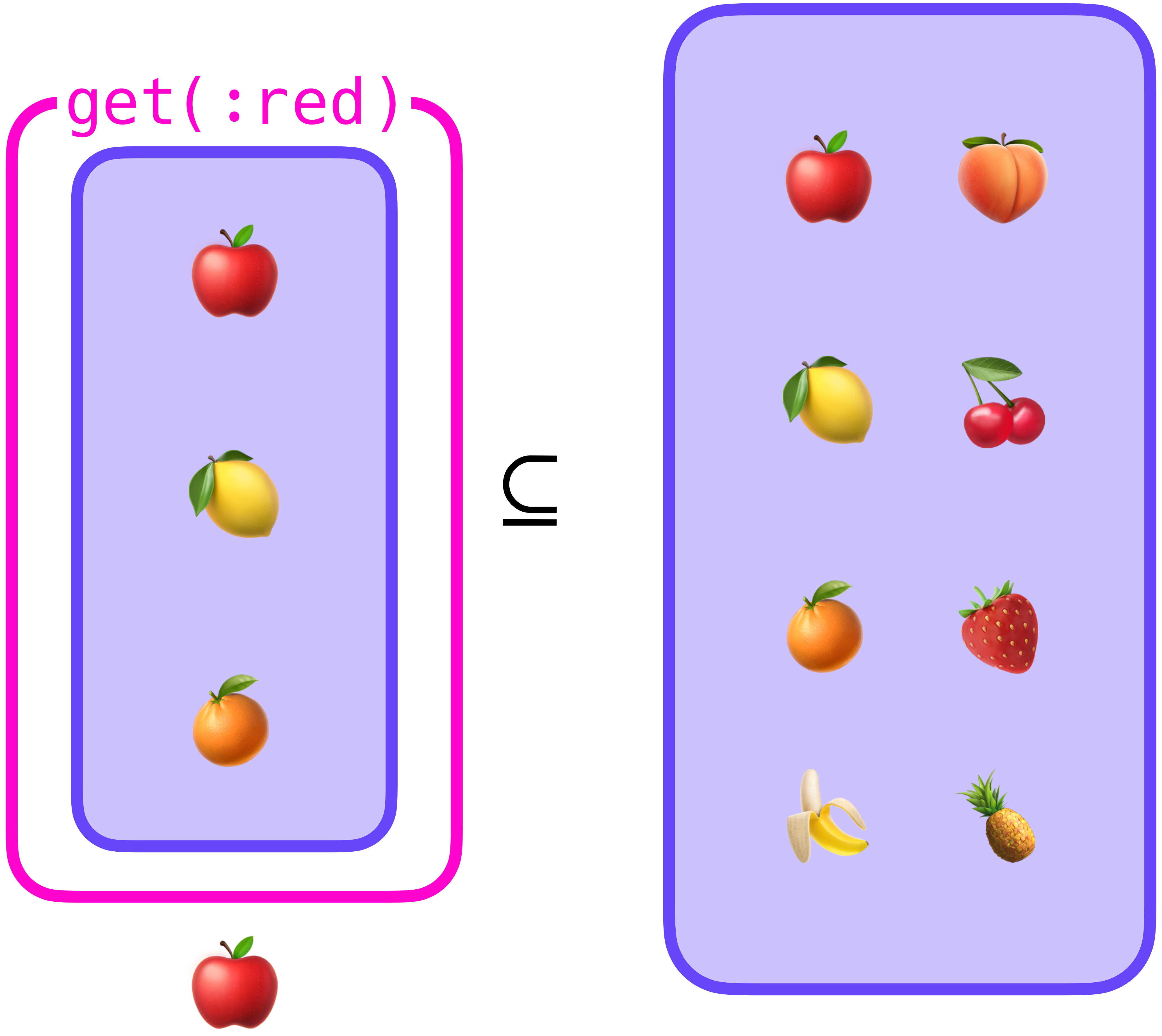
Mental Framework 🏗️🧠

Monotone Functions



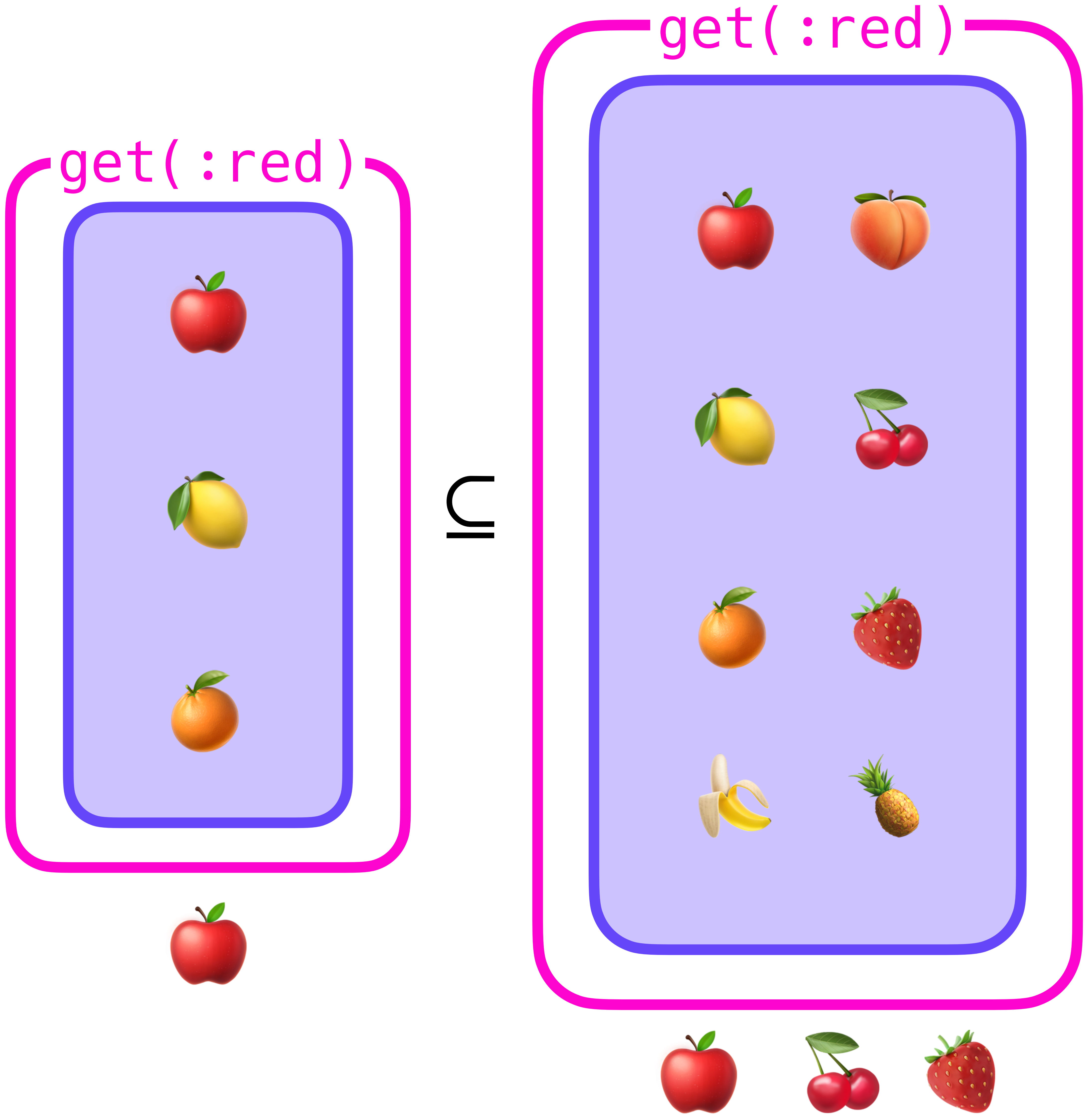
Mental Framework 🏗️🧠

Monotone Functions



Mental Framework 🏗️🧠

Monotone Functions

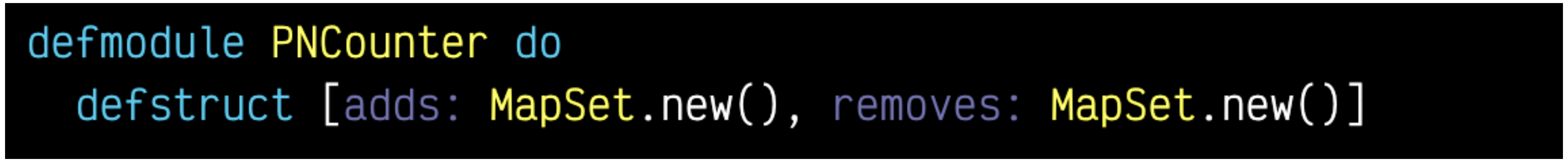


It’s All About that Data 📊

PNCounter

It’s All About that Data 📊

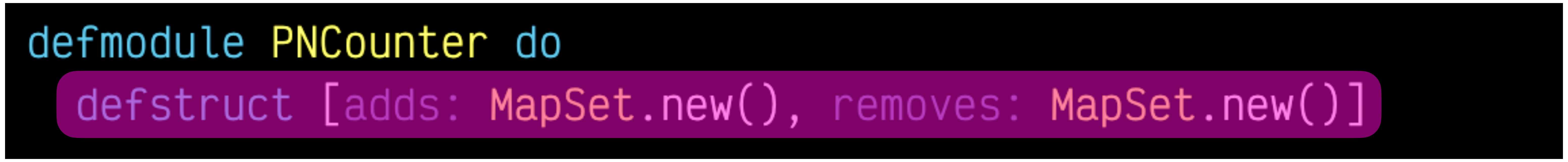
PNCounter



```
defmodule PNCounter do
  defstruct [adds: MapSet.new(), removes: MapSet.new()]
```

It’s All About that Data 📊

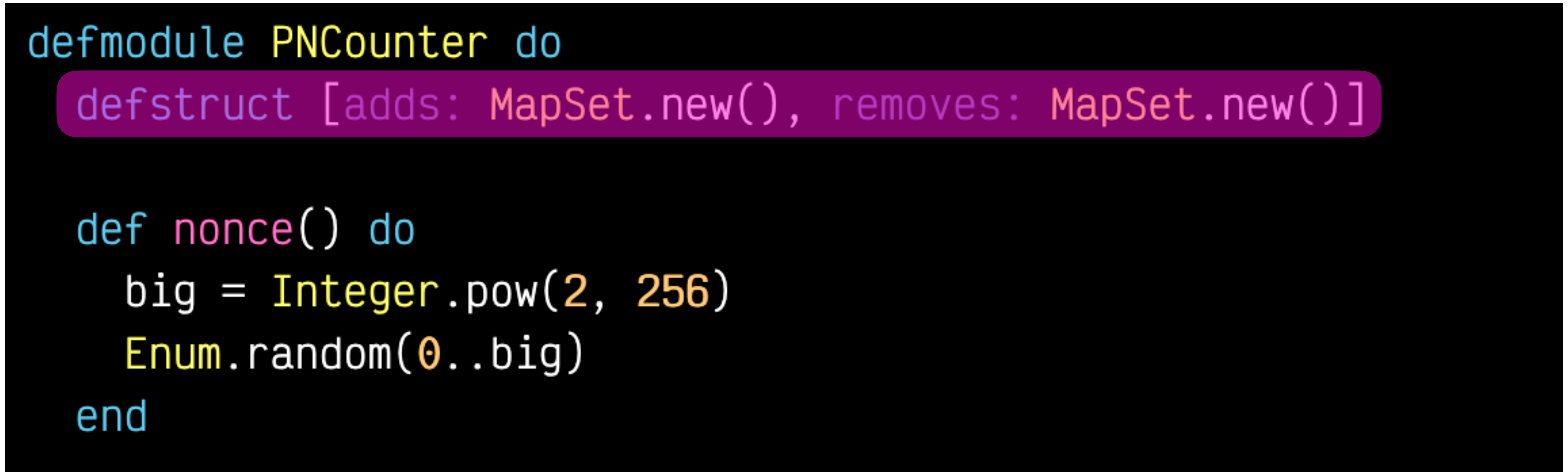
PNCounter



```elixir
defmodule PNCounter do
  defstruct [adds: MapSet.new(), removes: MapSet.new()]
```

It’s All About that Data 📊

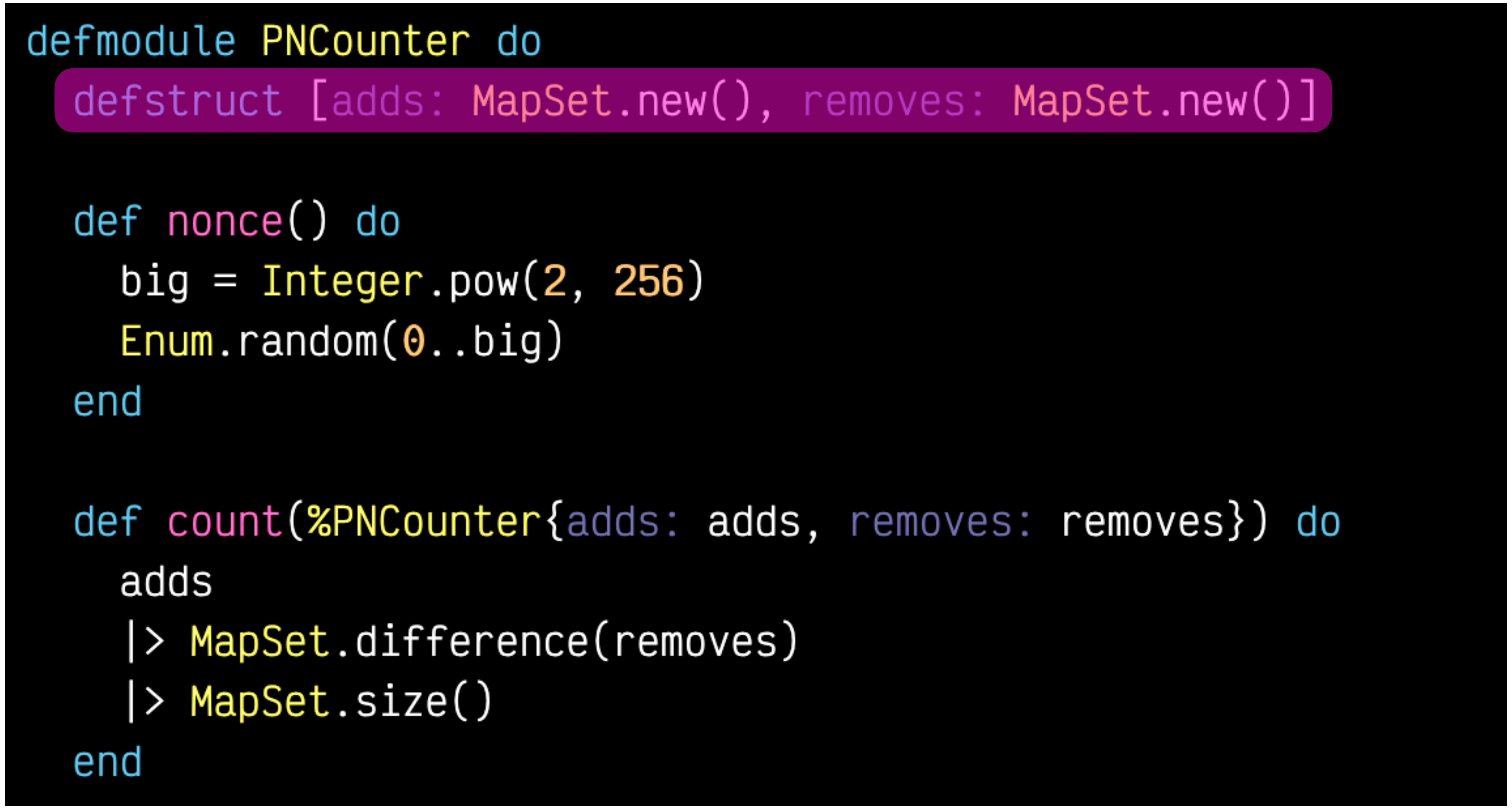
PNCounter



```
defmodule PNCounter do
  defstruct [adds: MapSet.new(), removes: MapSet.new()]

  def nonce() do
    big = Integer.pow(2, 256)
    Enum.random(0..big)
  end
```

It’s All About that Data 📊

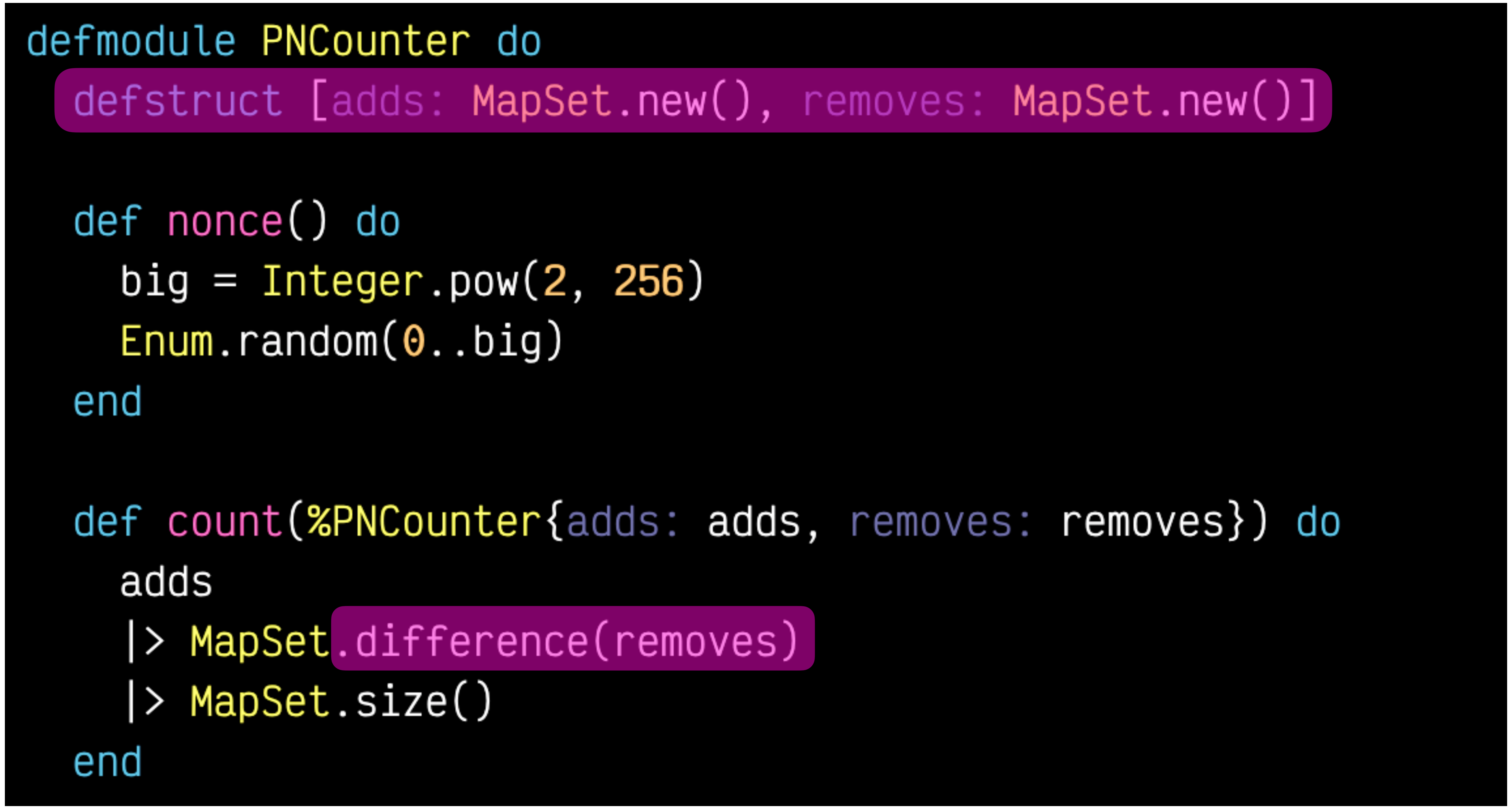
PNCounter



```elixir
defmodule PNCounter do
  defstruct [adds: MapSet.new(), removes: MapSet.new()]

  def nonce() do
    big = Integer.pow(2, 256)
    Enum.random(0..big)
  end

  def count(%PNCounter{adds: adds, removes: removes}) do
    adds
    |> MapSet.difference(removes)
    |> MapSet.size()
  end
```

It’s All About that Data 📊

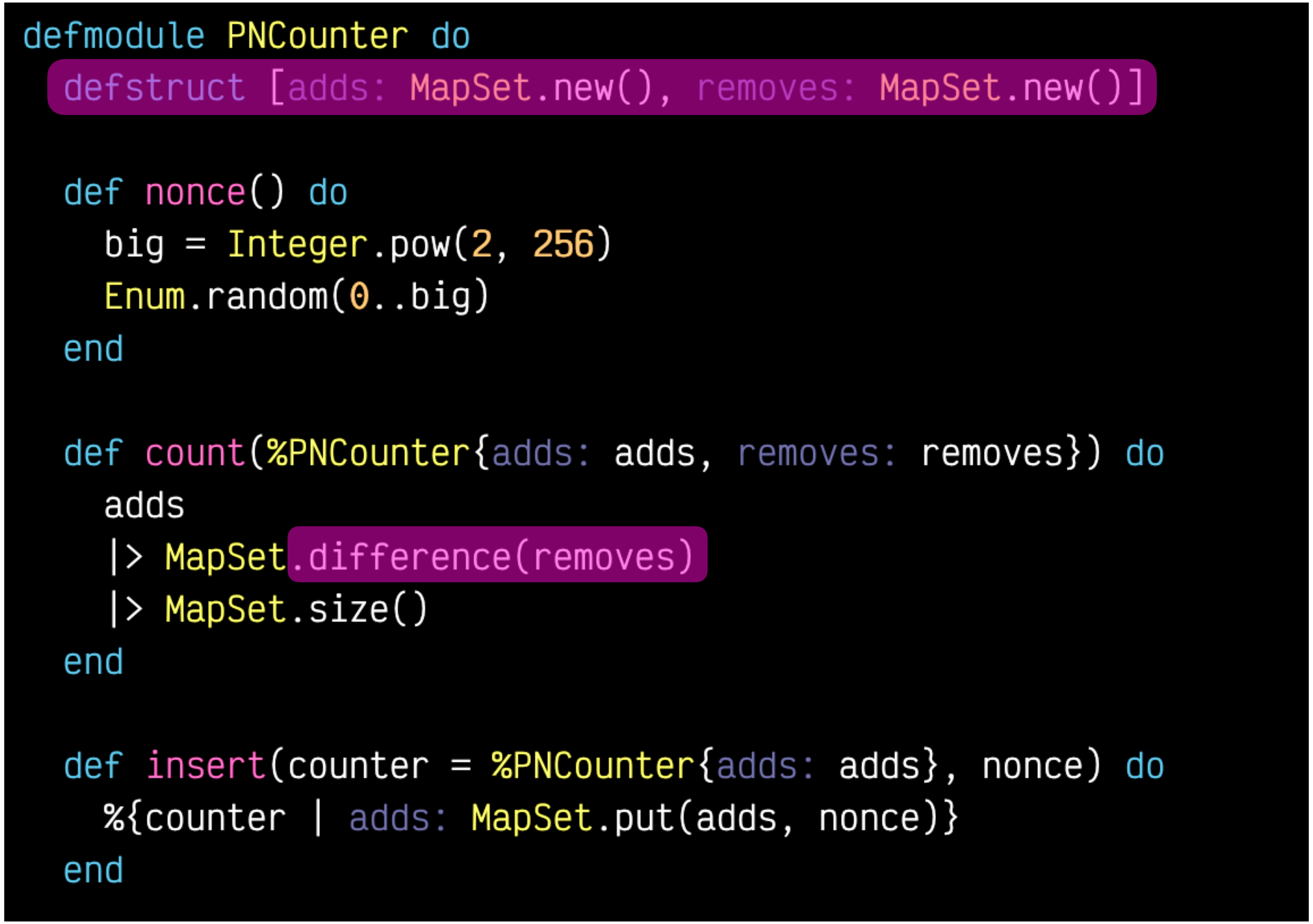
PNCounter



```
defmodule PNCounter do
  defstruct [adds: MapSet.new(), removes: MapSet.new()]

  def nonce() do
    big = Integer.pow(2, 256)
    Enum.random(0..big)
  end

  def count(%PNCounter{adds: adds, removes: removes}) do
    adds
    |> MapSet.difference(removes)
    |> MapSet.size()
  end
```

It’s All About that Data 📊

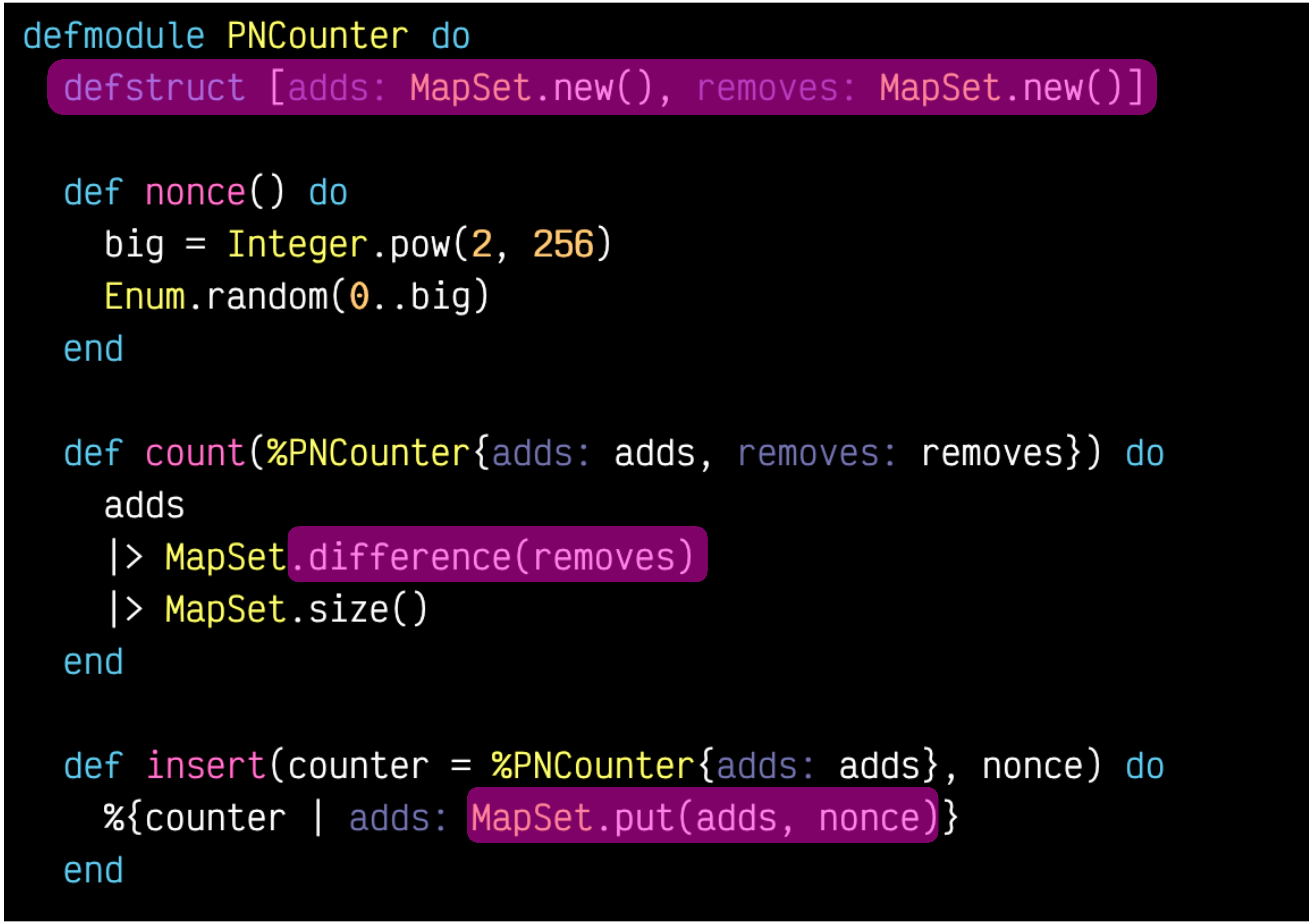
PNCounter



```elixir
defmodule PNCounter do
  defstruct [adds: MapSet.new(), removes: MapSet.new()]

  def nonce() do
    big = Integer.pow(2, 256)
    Enum.random(0..big)
  end

  def count(%PNCounter{adds: adds, removes: removes}) do
    adds
    |> MapSet.difference(removes)
    |> MapSet.size()
  end

  def insert(counter = %PNCounter{adds: adds}, nonce) do
    %{counter | adds: MapSet.put(adds, nonce)}
  end
```

It’s All About that Data 📊

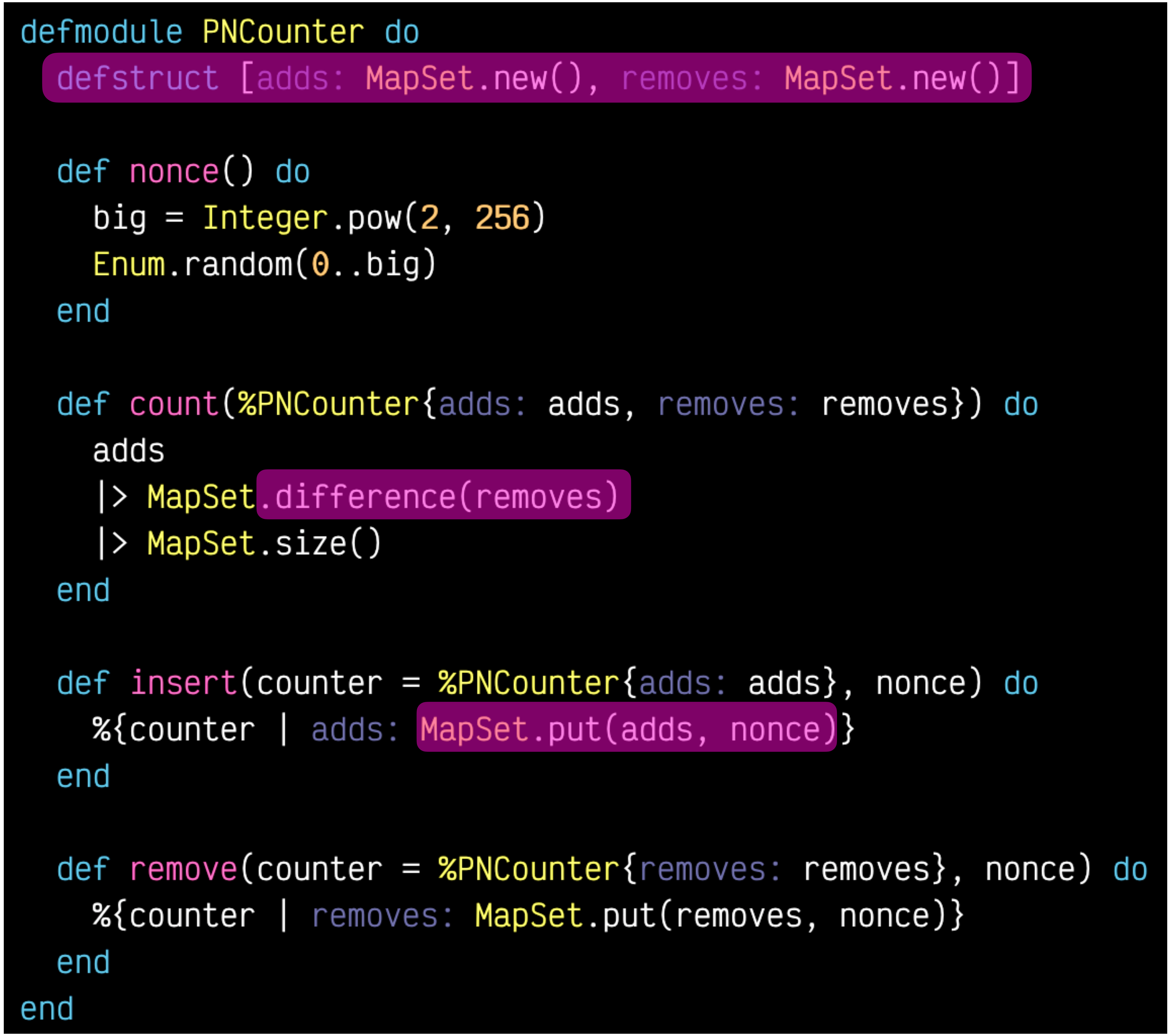
PNCounter



```
defmodule PNCounter do
  defstruct [adds: MapSet.new(), removes: MapSet.new()]

  def nonce() do
    big = Integer.pow(2, 256)
    Enum.random(0..big)
  end

  def count(%PNCounter{adds: adds, removes: removes}) do
    adds
    |> MapSet.difference(removes)
    |> MapSet.size()
  end

  def insert(counter = %PNCounter{adds: adds}, nonce) do
    %{counter | adds: MapSet.put(adds, nonce)}
  end
```

It’s All About that Data 📊

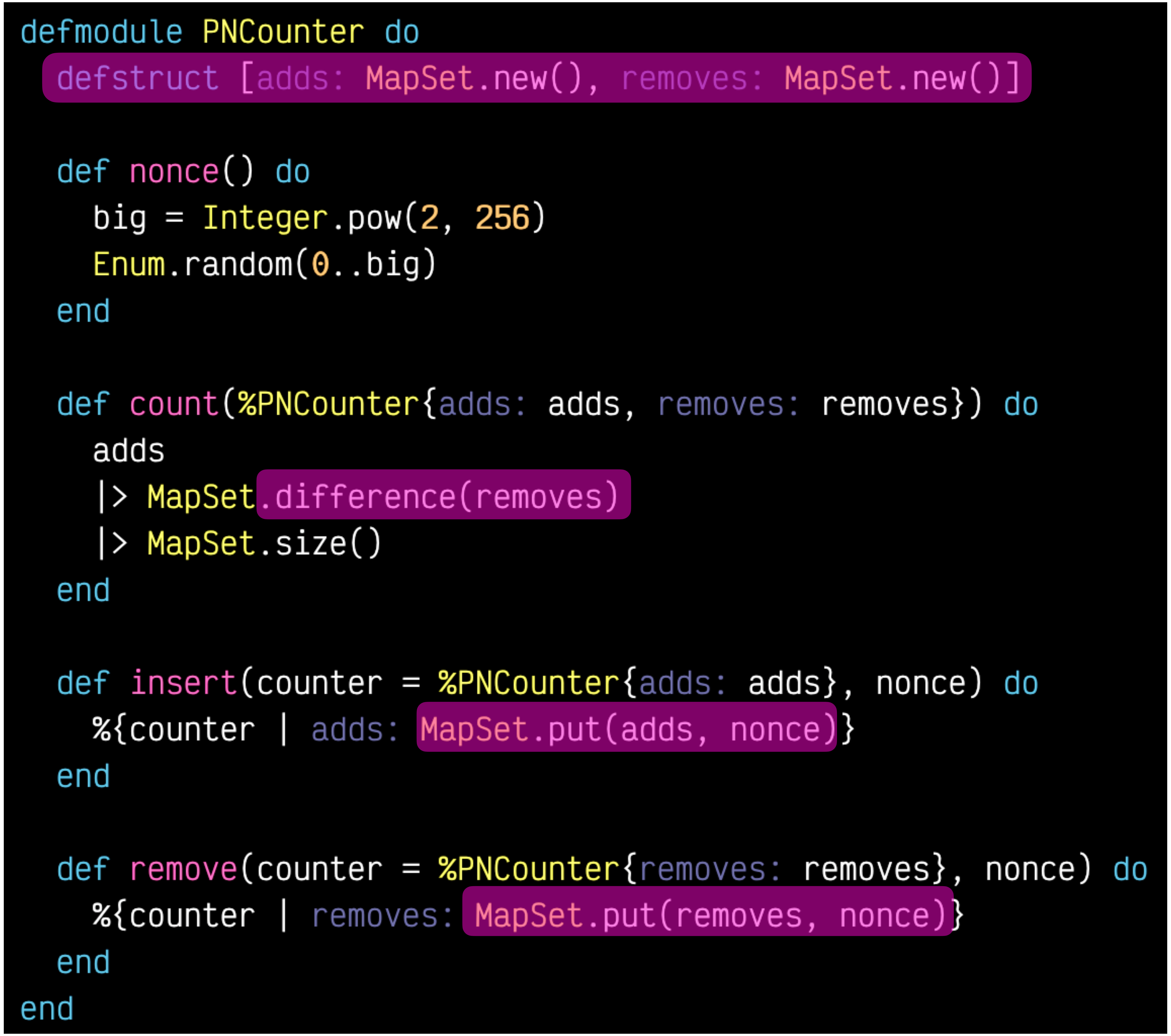
PNCounter



```
defmodule PNCounter do
  defstruct [adds: MapSet.new(), removes: MapSet.new()]

  def nonce() do
    big = Integer.pow(2, 256)
    Enum.random(0..big)
  end

  def count(%PNCounter{adds: adds, removes: removes}) do
    adds
    |> MapSet.difference(removes)
    |> MapSet.size()
  end

  def insert(counter = %PNCounter{adds: adds}, nonce) do
    %{counter | adds: MapSet.put(adds, nonce)}
  end

  def remove(counter = %PNCounter{removes: removes}, nonce) do
    %{counter | removes: MapSet.put(removes, nonce)}
  end
end
```

It’s All About that Data 📊

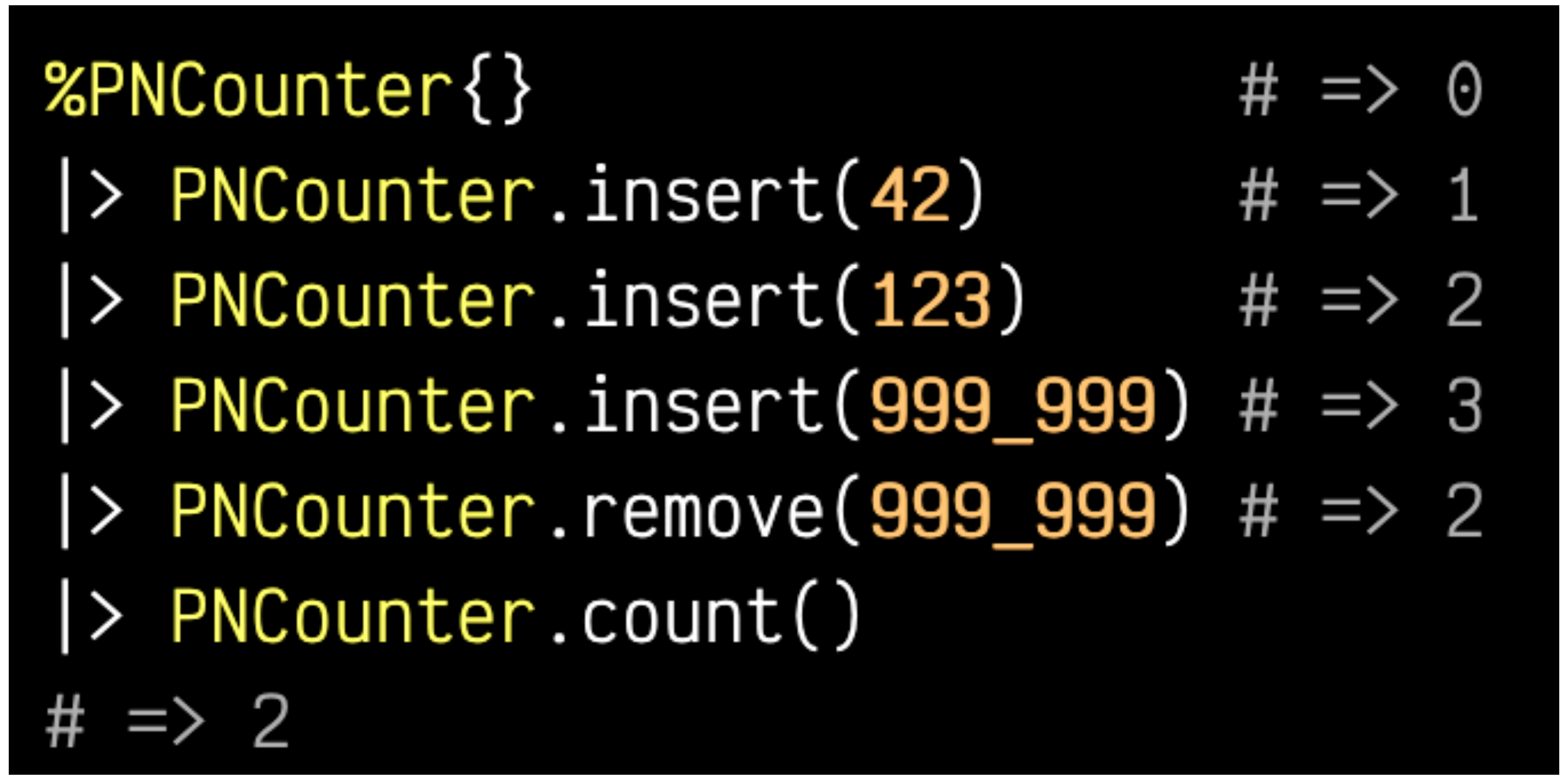
PNCounter



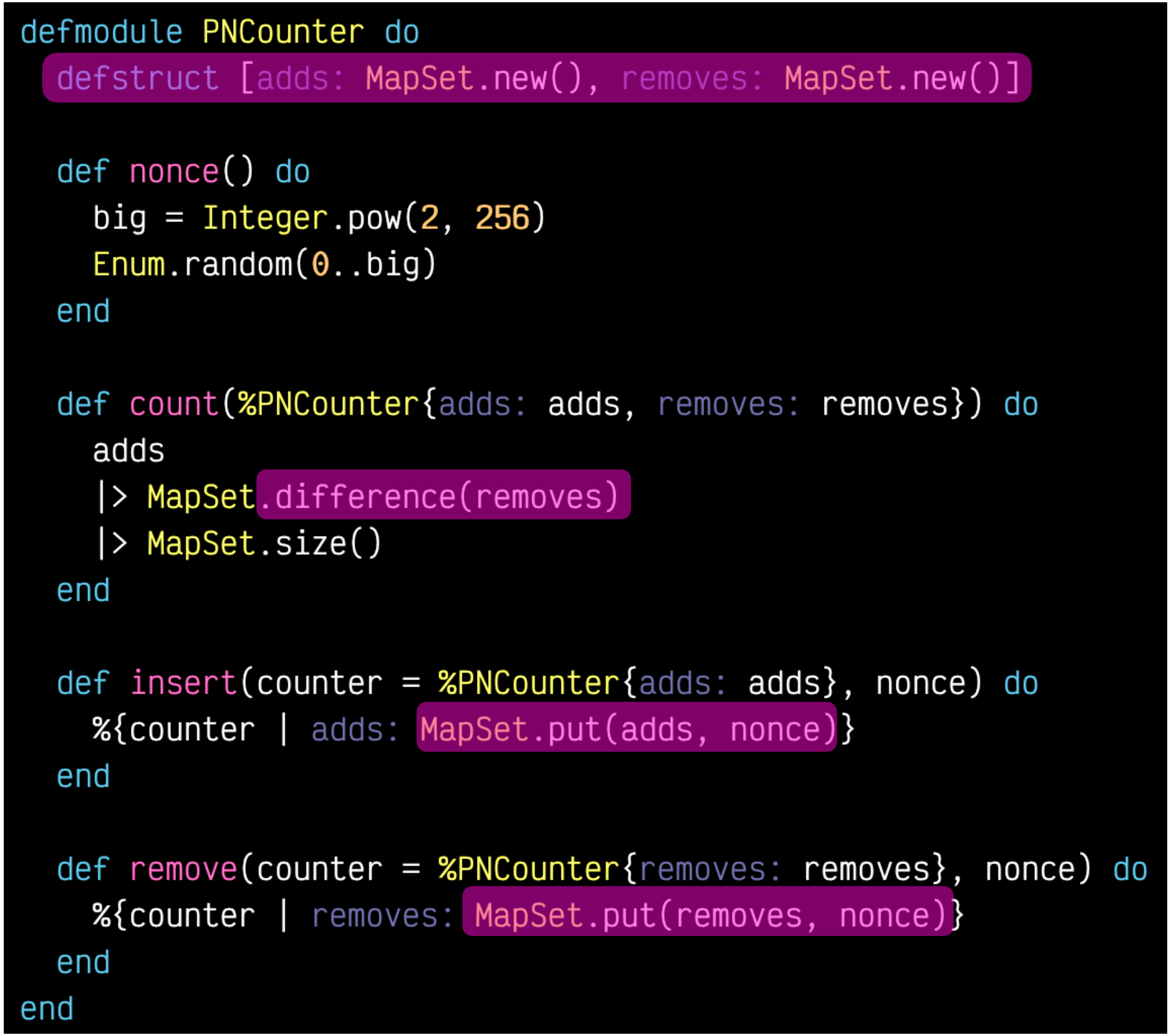
```elixir
defmodule PNCounter do
  defstruct [adds: MapSet.new(), removes: MapSet.new()]

  def nonce() do
    big = Integer.pow(2, 256)
    Enum.random(0..big)
  end

  def count(%PNCounter{adds: adds, removes: removes}) do
    adds
    |> MapSet.difference(removes)
    |> MapSet.size()
  end

  def insert(counter = %PNCounter{adds: adds}, nonce) do
    %{counter | adds: MapSet.put(adds, nonce)}
  end

  def remove(counter = %PNCounter{removes: removes}, nonce) do
    %{counter | removes: MapSet.put(removes, nonce)}
  end
end
```

It’s All About that Data 📊

PNCounter



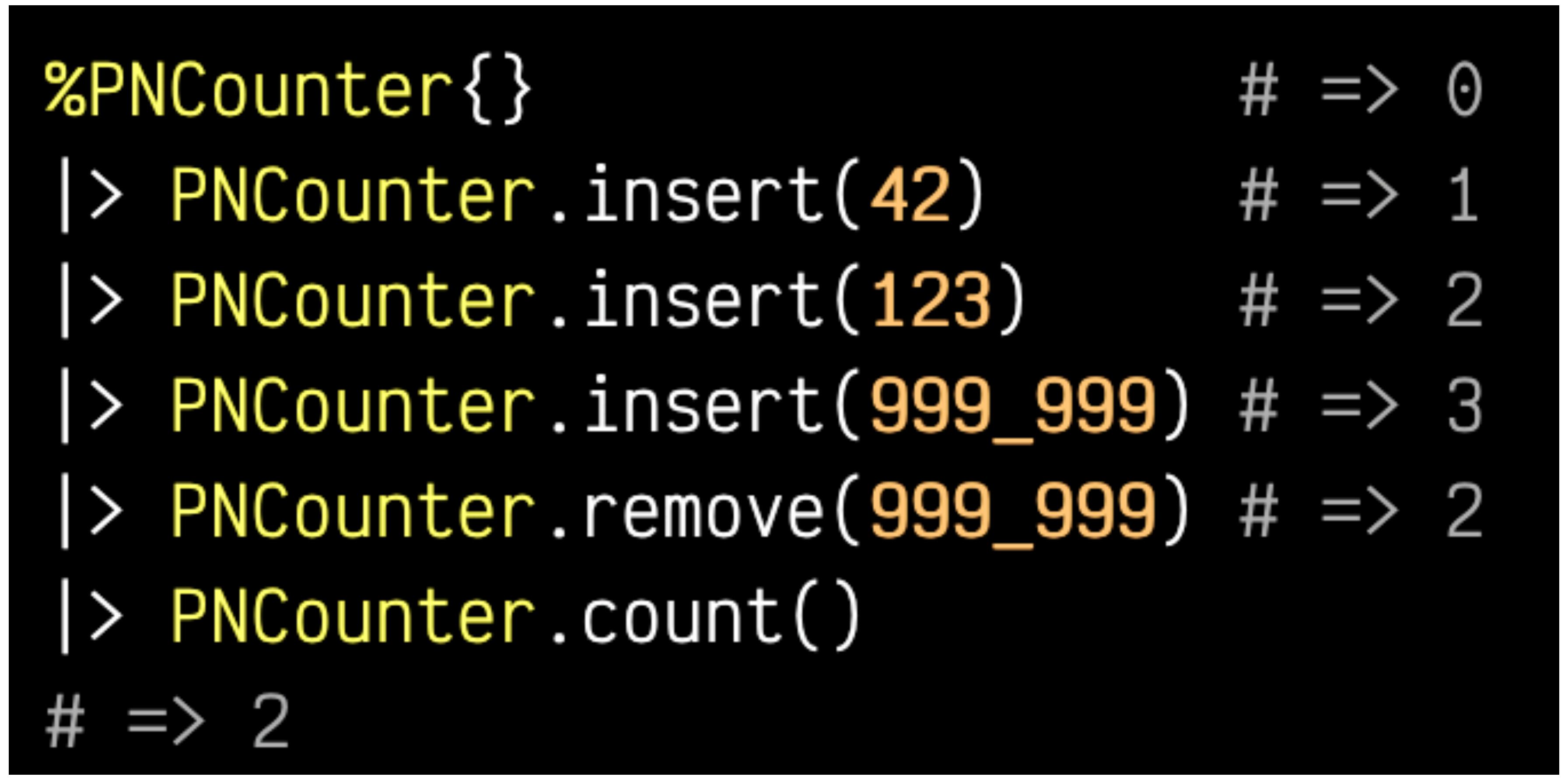
```
%PNCounter{}                    # => 0
|> PNCounter.insert(42)         # => 1
|> PNCounter.insert(123)        # => 2
|> PNCounter.insert(999_999)    # => 3
|> PNCounter.remove(999_999)    # => 2
|> PNCounter.count()
# => 2
```



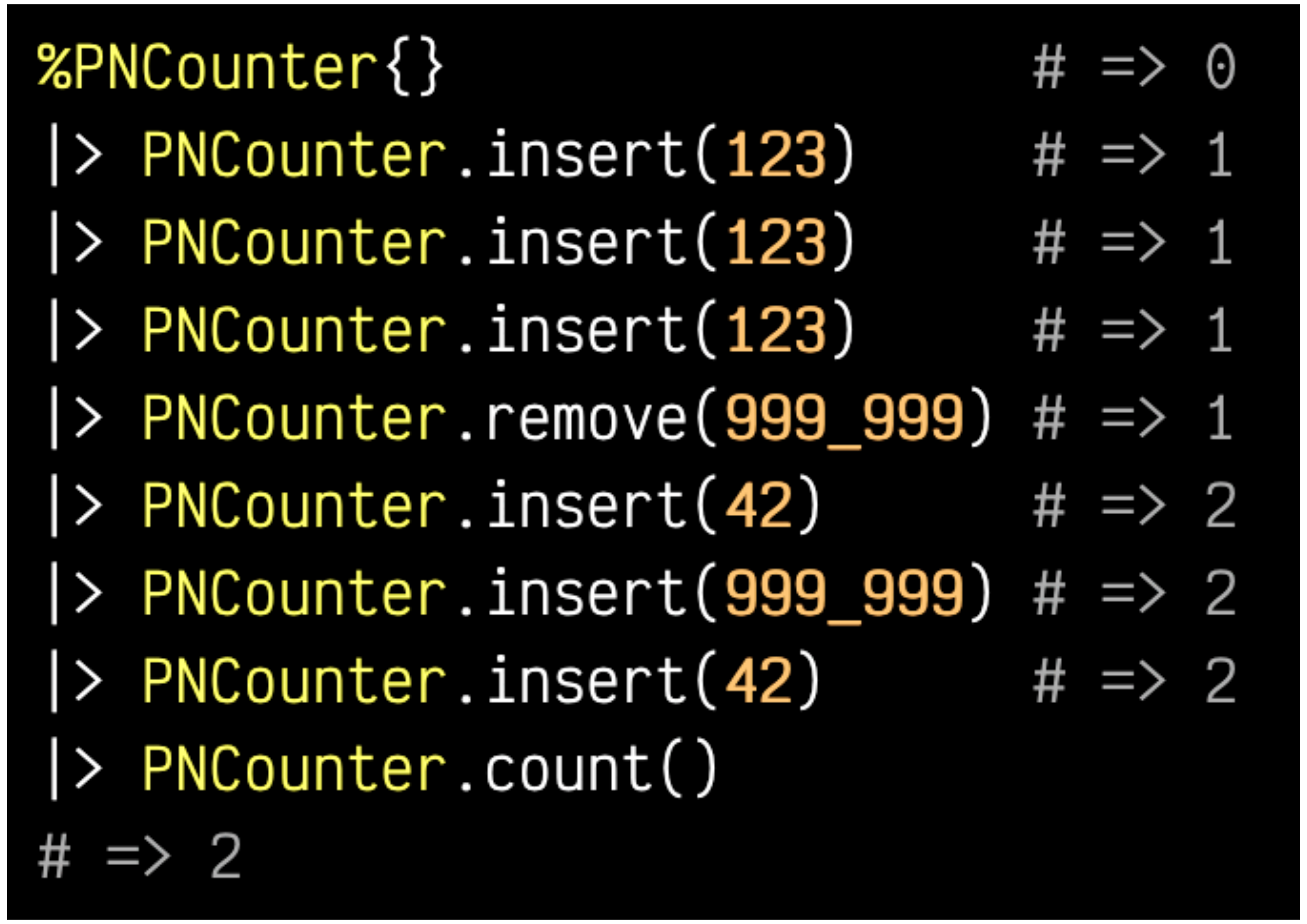
```
defmodule PNCounter do
  defstruct [adds: MapSet.new(), removes: MapSet.new()]

  def nonce() do
    big = Integer.pow(2, 256)
    Enum.random(0..big)
  end

  def count(%PNCounter{adds: adds, removes: removes}) do
    adds
    |> MapSet.difference(removes)
    |> MapSet.size()
  end

  def insert(counter = %PNCounter{adds: adds}, nonce) do
    %{counter | adds: MapSet.put(adds, nonce)}
  end

  def remove(counter = %PNCounter{removes: removes}, nonce) do
    %{counter | removes: MapSet.put(removes, nonce)}
  end
end
```

It’s All About that Data 📊

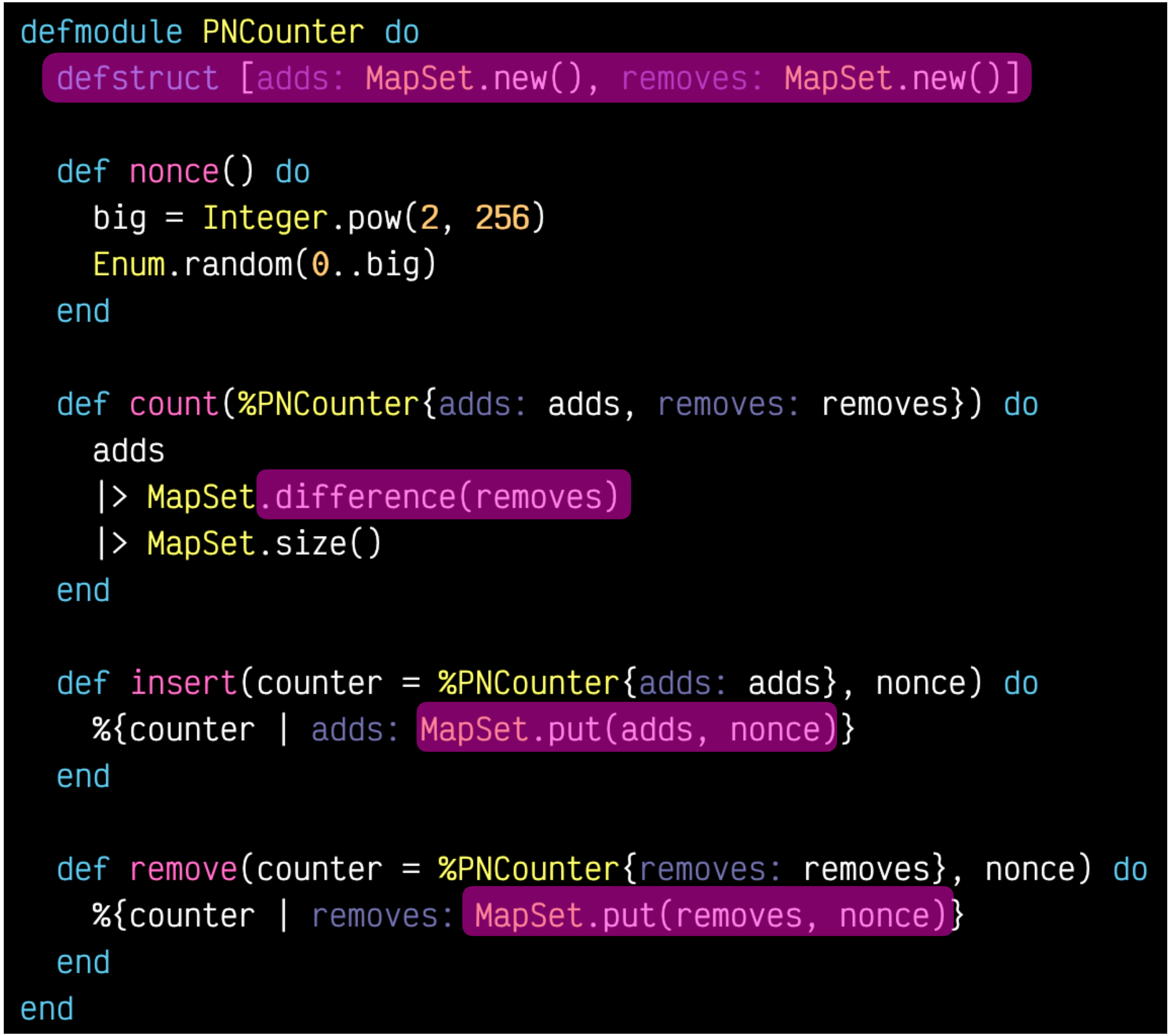
PNCounter



```
%PNCounter{}                 # => 0
|> PNCounter.insert(42)      # => 1
|> PNCounter.insert(123)     # => 2
|> PNCounter.insert(999_999) # => 3
|> PNCounter.remove(999_999) # => 2
|> PNCounter.count()
# => 2
```



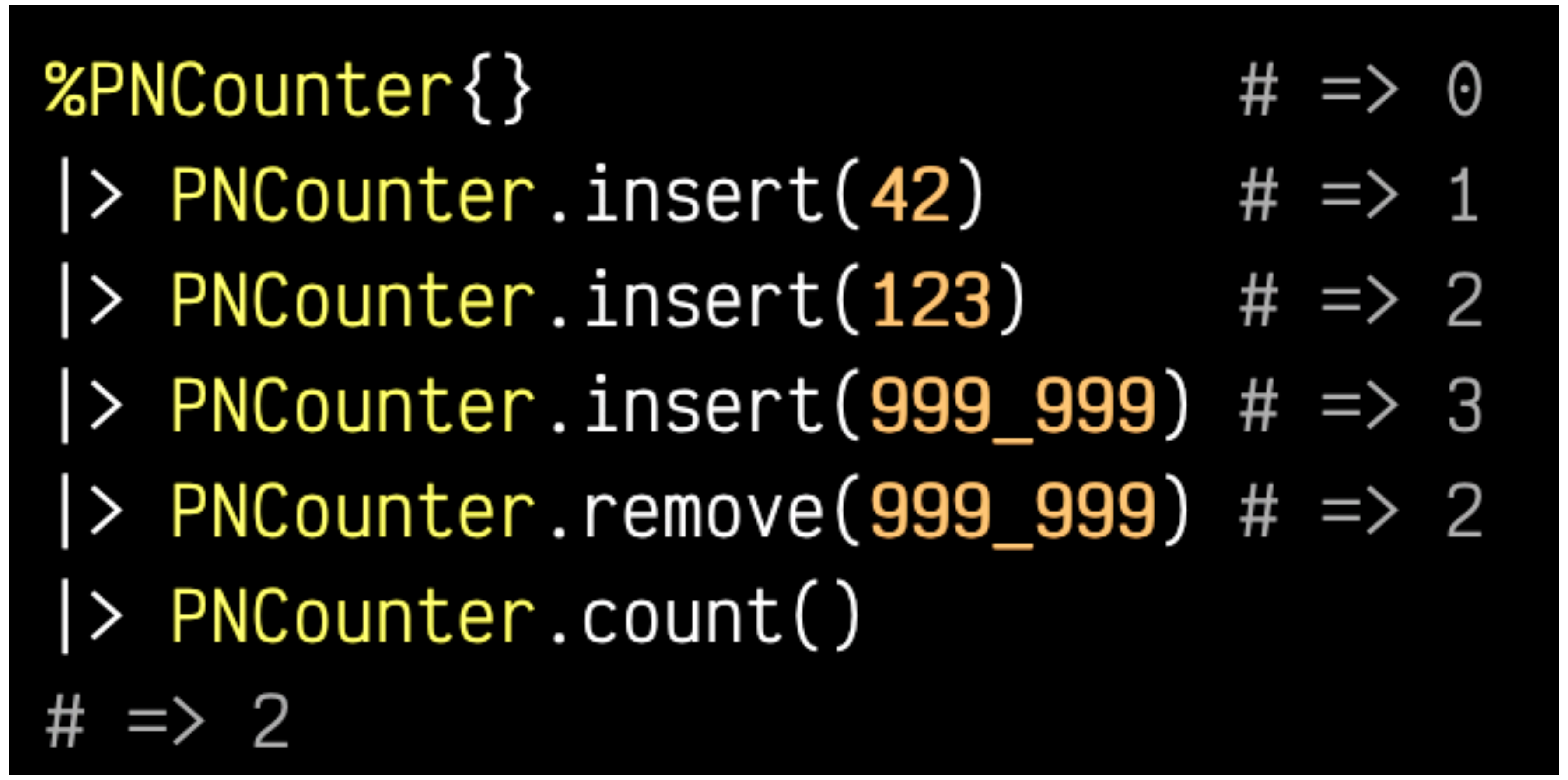
```
%PNCounter{}                 # => 0
|> PNCounter.insert(123)     # => 1
|> PNCounter.insert(123)     # => 1
|> PNCounter.insert(123)     # => 1
|> PNCounter.remove(999_999) # => 1
|> PNCounter.insert(42)      # => 2
|> PNCounter.insert(999_999) # => 2
|> PNCounter.insert(42)      # => 2
|> PNCounter.count()
# => 2
```



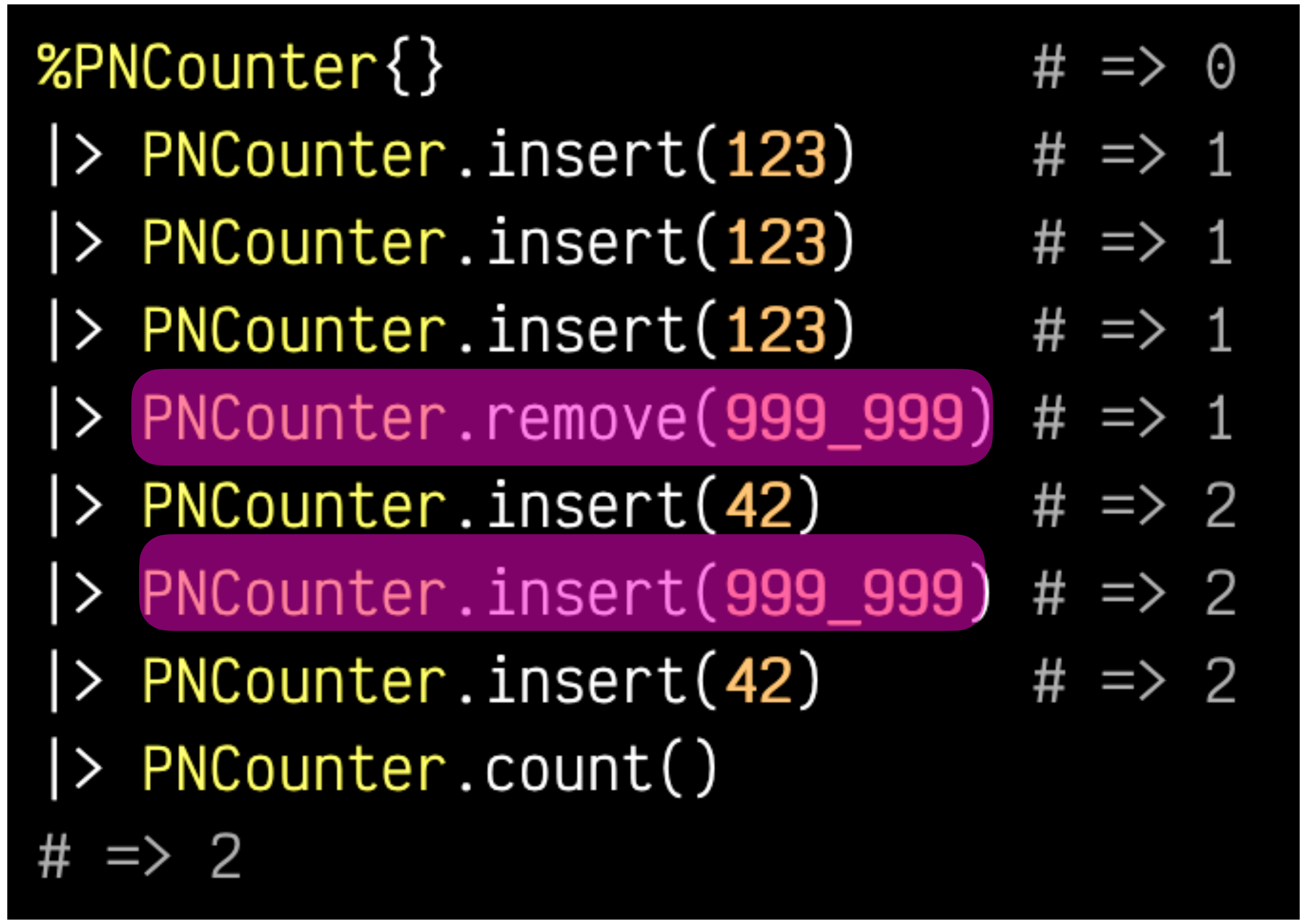
```
defmodule PNCounter do
  defstruct [adds: MapSet.new(), removes: MapSet.new()]

  def nonce() do
    big = Integer.pow(2, 256)
    Enum.random(0..big)
  end

  def count(%PNCounter{adds: adds, removes: removes}) do
    adds
    |> MapSet.difference(removes)
    |> MapSet.size()
  end

  def insert(counter = %PNCounter{adds: adds}, nonce) do
    %{counter | adds: MapSet.put(adds, nonce)}
  end

  def remove(counter = %PNCounter{removes: removes}, nonce) do
    %{counter | removes: MapSet.put(removes, nonce)}
  end
end
```

It’s All About that Data 📊

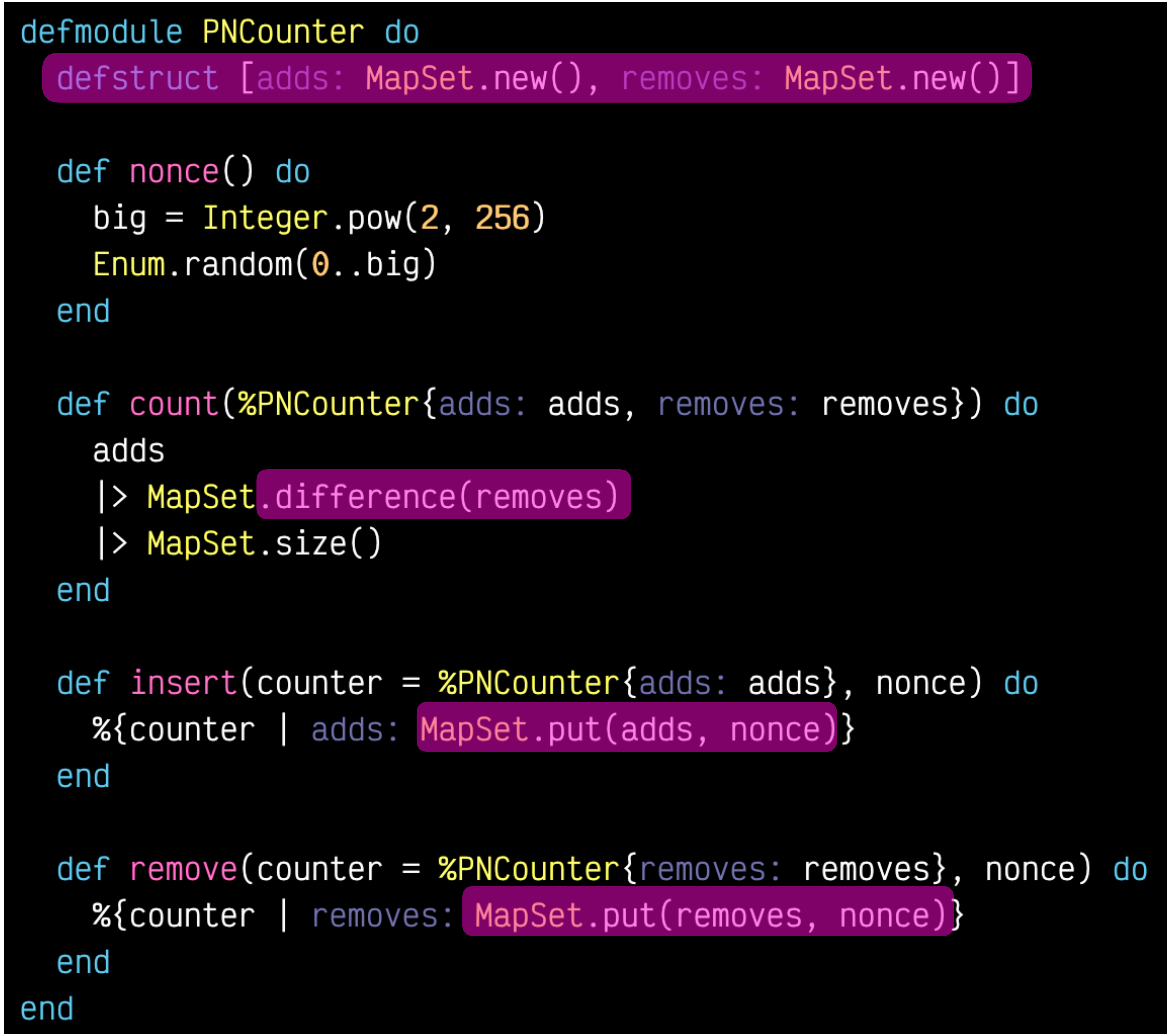
PNCounter



```
%PNCounter{}                   # => 0
|> PNCounter.insert(42)        # => 1
|> PNCounter.insert(123)       # => 2
|> PNCounter.insert(999_999)   # => 3
|> PNCounter.remove(999_999)   # => 2
|> PNCounter.count()
# => 2
```



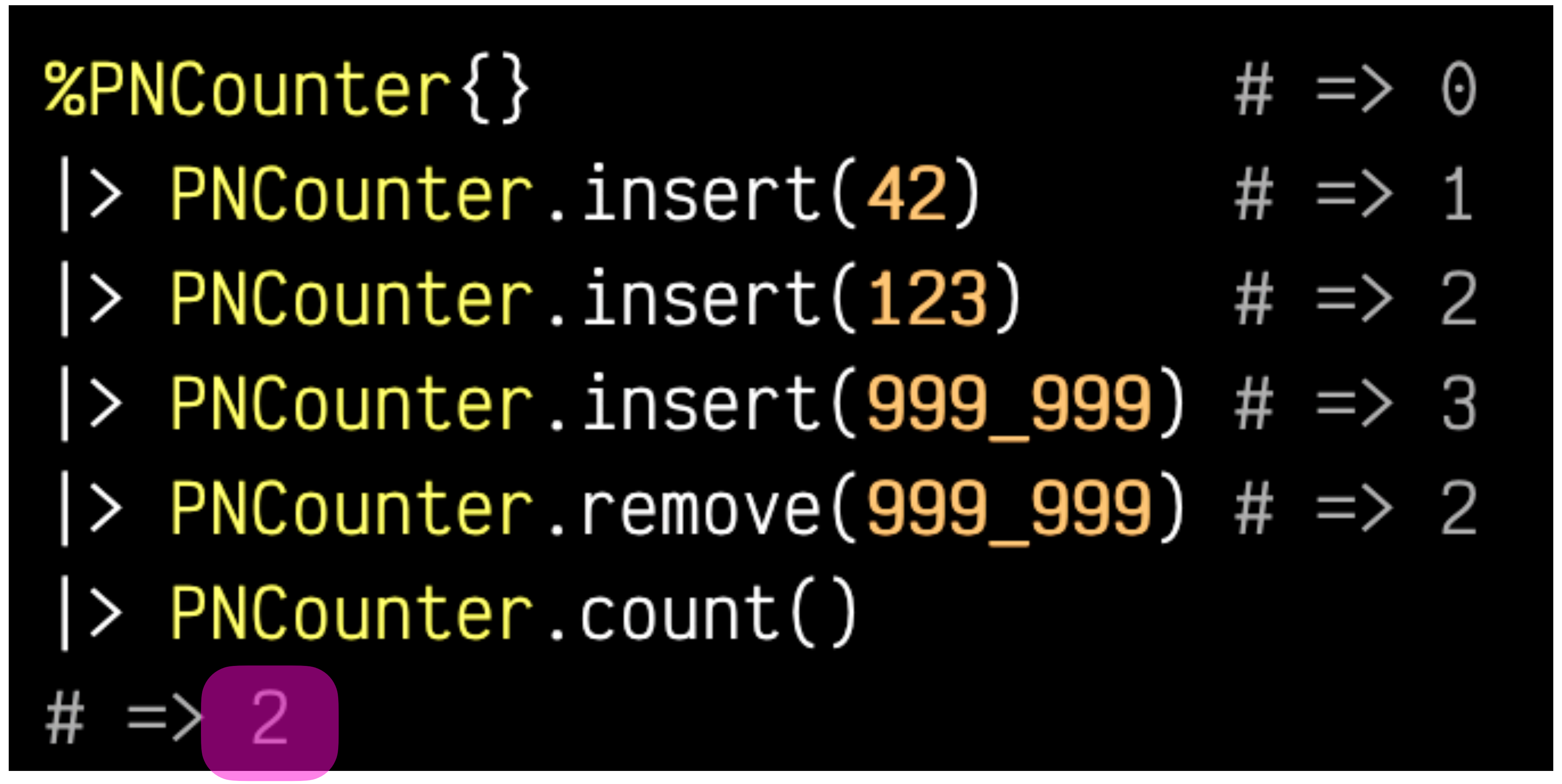
```
%PNCounter{}                   # => 0
|> PNCounter.insert(123)       # => 1
|> PNCounter.insert(123)       # => 1
|> PNCounter.insert(123)       # => 1
|> PNCounter.remove(999_999)   # => 1
|> PNCounter.insert(42)        # => 2
|> PNCounter.insert(999_999)   # => 2
|> PNCounter.insert(42)        # => 2
|> PNCounter.count()
# => 2
```



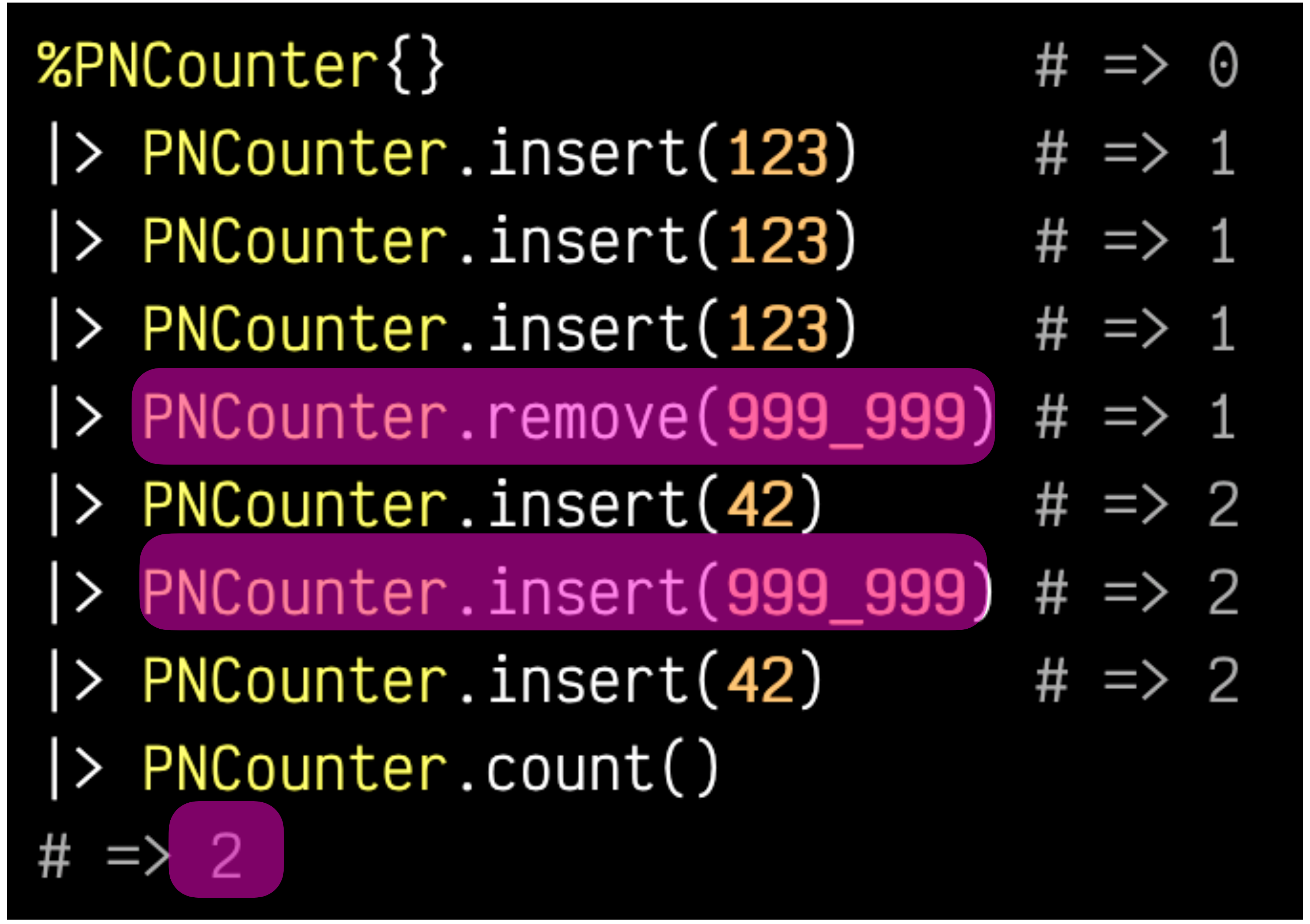
```
defmodule PNCounter do
  defstruct [adds: MapSet.new(), removes: MapSet.new()]

  def nonce() do
    big = Integer.pow(2, 256)
    Enum.random(0..big)
  end

  def count(%PNCounter{adds: adds, removes: removes}) do
    adds
    |> MapSet.difference(removes)
    |> MapSet.size()
  end

  def insert(counter = %PNCounter{adds: adds}, nonce) do
    %{counter | adds: MapSet.put(adds, nonce)}
  end

  def remove(counter = %PNCounter{removes: removes}, nonce) do
    %{counter | removes: MapSet.put(removes, nonce)}
  end
end
```

It’s All About that Data 📊

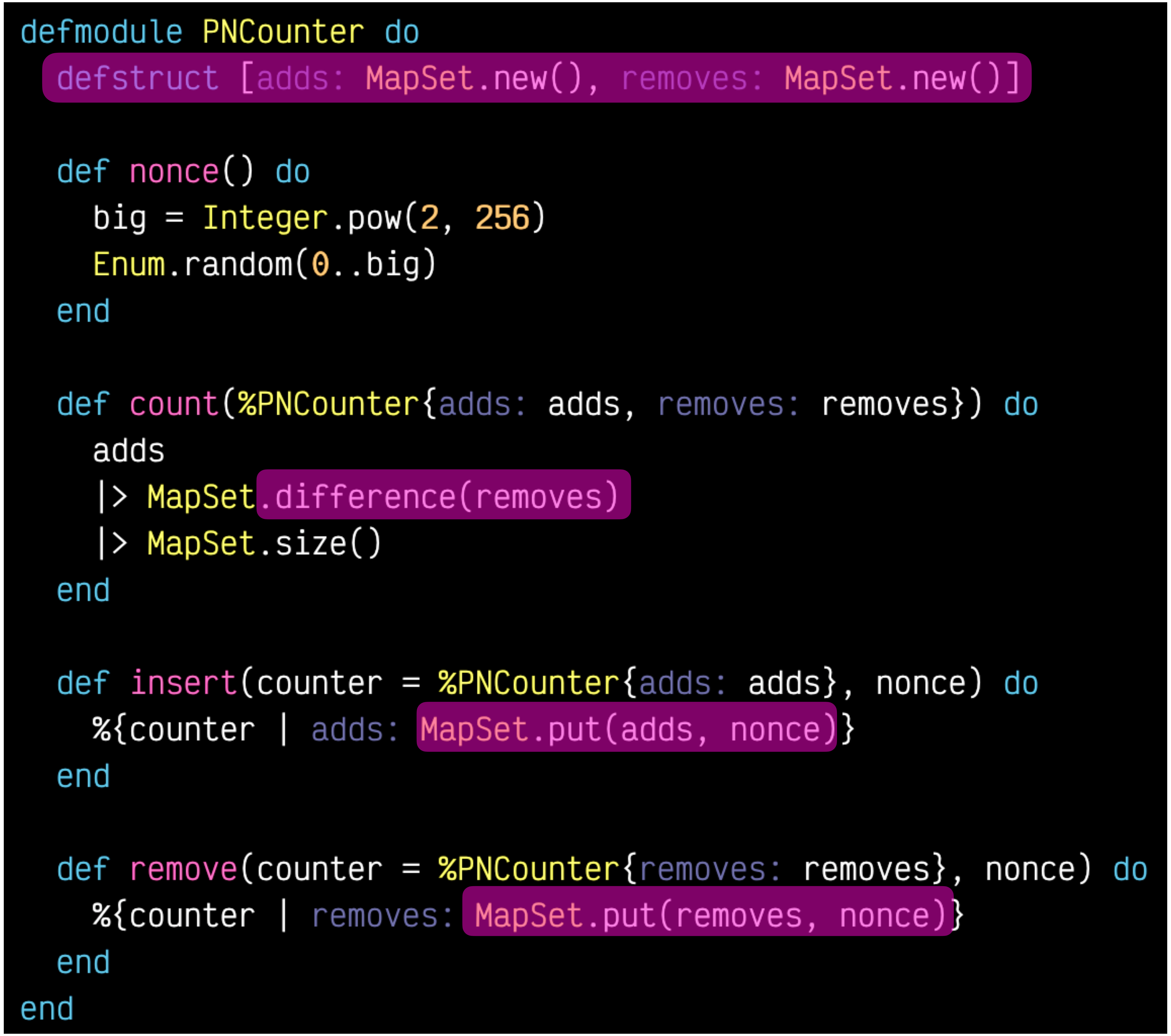
PNCounter



```elixir
%PNCounter{}                    # => 0
|> PNCounter.insert(42)         # => 1
|> PNCounter.insert(123)        # => 2
|> PNCounter.insert(999_999)    # => 3
|> PNCounter.remove(999_999)    # => 2
|> PNCounter.count()
# => 2
```



```elixir
%PNCounter{}                    # => 0
|> PNCounter.insert(123)        # => 1
|> PNCounter.insert(123)        # => 1
|> PNCounter.insert(123)        # => 1
|> PNCounter.remove(999_999)    # => 1
|> PNCounter.insert(42)         # => 2
|> PNCounter.insert(999_999)    # => 2
|> PNCounter.insert(42)         # => 2
|> PNCounter.count()
# => 2
```



```elixir
defmodule PNCounter do
  defstruct [adds: MapSet.new(), removes: MapSet.new()]

  def nonce() do
    big = Integer.pow(2, 256)
    Enum.random(0..big)
  end

  def count(%PNCounter{adds: adds, removes: removes}) do
    adds
    |> MapSet.difference(removes)
    |> MapSet.size()
  end

  def insert(counter = %PNCounter{adds: adds}, nonce) do
    %{counter | adds: MapSet.put(adds, nonce)}
  end

  def remove(counter = %PNCounter{removes: removes}, nonce) do
    %{counter | removes: MapSet.put(removes, nonce)}
  end
end
```

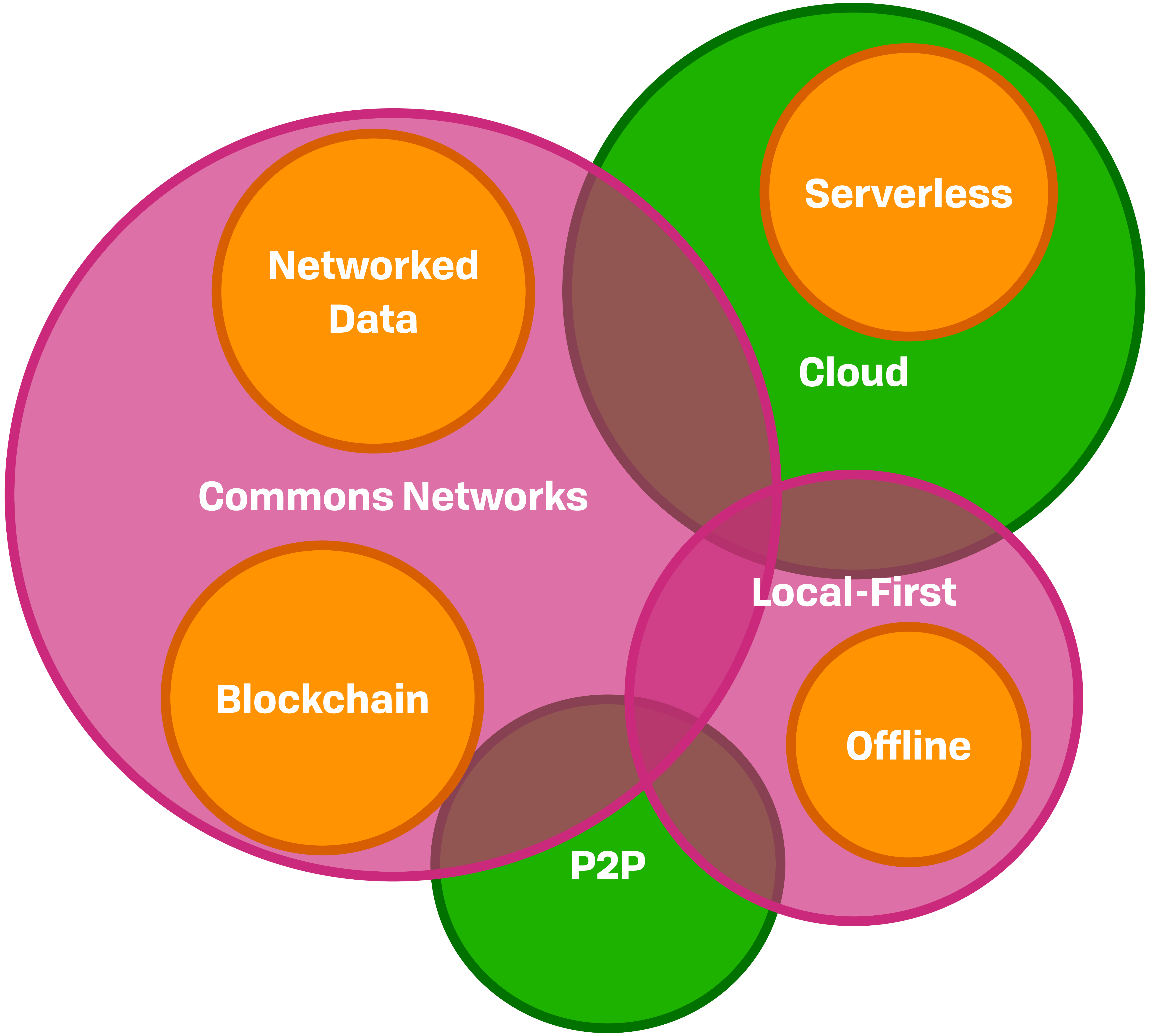
Grappling With Reality

Towards a Solution

⚗️✨

Towards a Solution ⚗️✨

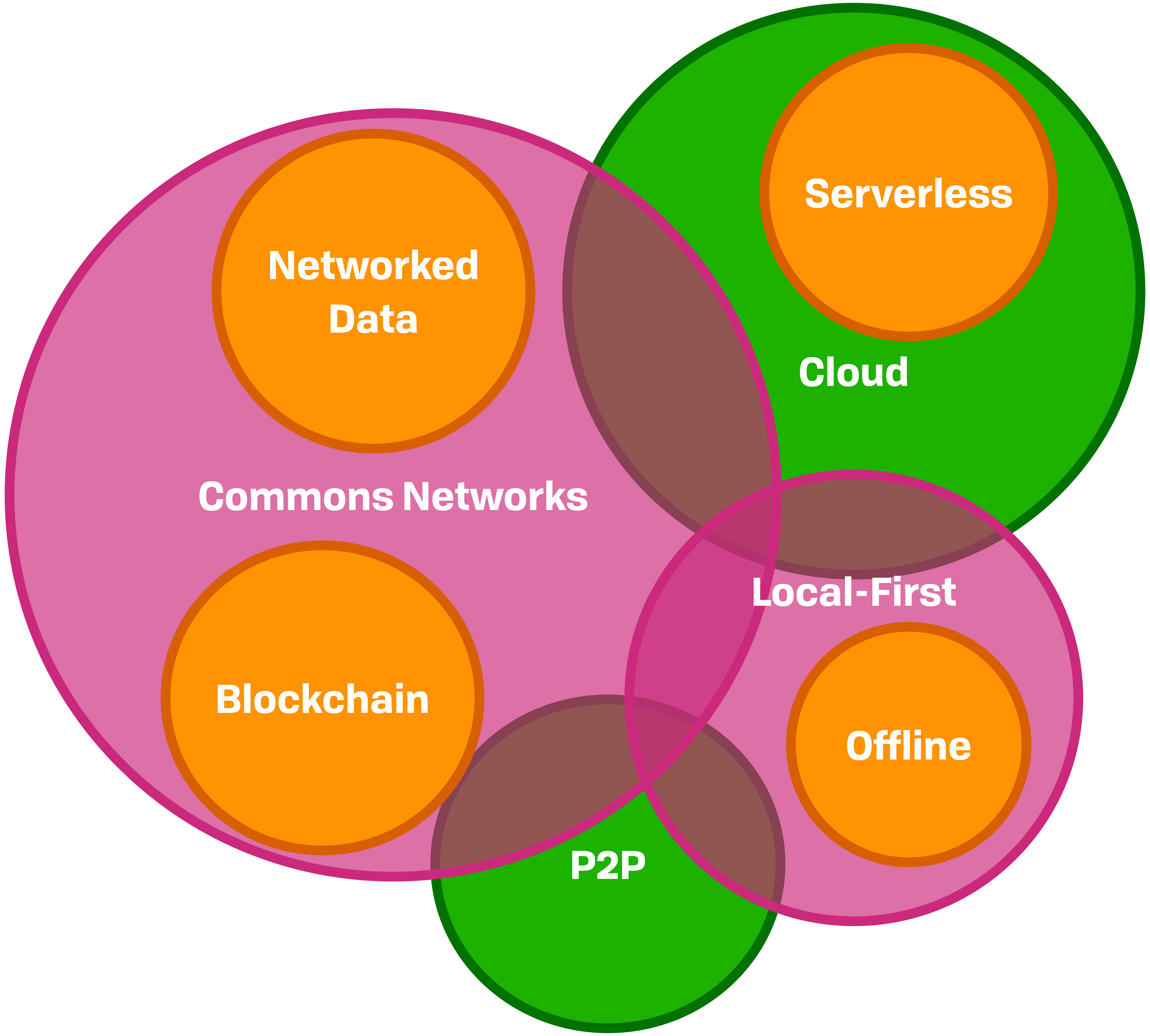
Evolving Toolbox



Towards a Solution ⚗️✨

Evolving Toolbox

Radical shifts how we think about auth, locality of reference, ownership, and reliability



Towards a Solution 🔮✨

Mutable Pointers

```
send(:example@42.123.45.6, :ping)
 %{node_id => %{path => content}}
```

- Single-source server/client
  - DNS: hostname → IP address
  - PIDs: number → address
- Focused: **physical** network
- Referential **opacity** (same PID, different data)

Towards a Solution 🔮✨

Mutable Pointers

```
send(:example@42.123.45.6, :ping)
 %{node_id => %{path => content}}
```

- Single-source server/client
  - DNS: hostname → IP address
  - PIDs: number → address
- Focused: **physical** network
- Referential **opacity** (same PID, different data)

PHYSICAL LOCATION 🗺️

Towards a Solution ⚗️✨

Mutable Pointers

```
send(:example@42.123.45.6, :ping)
 %{node_id => %{path => content}}
```

- Single-source server/client
  - DNS: hostname → IP address
  - PIDs: number → address
- Focused: **physical** network
- Referential **opacity** (same PID, different data)

VIRTUAL ADDRESS 📫

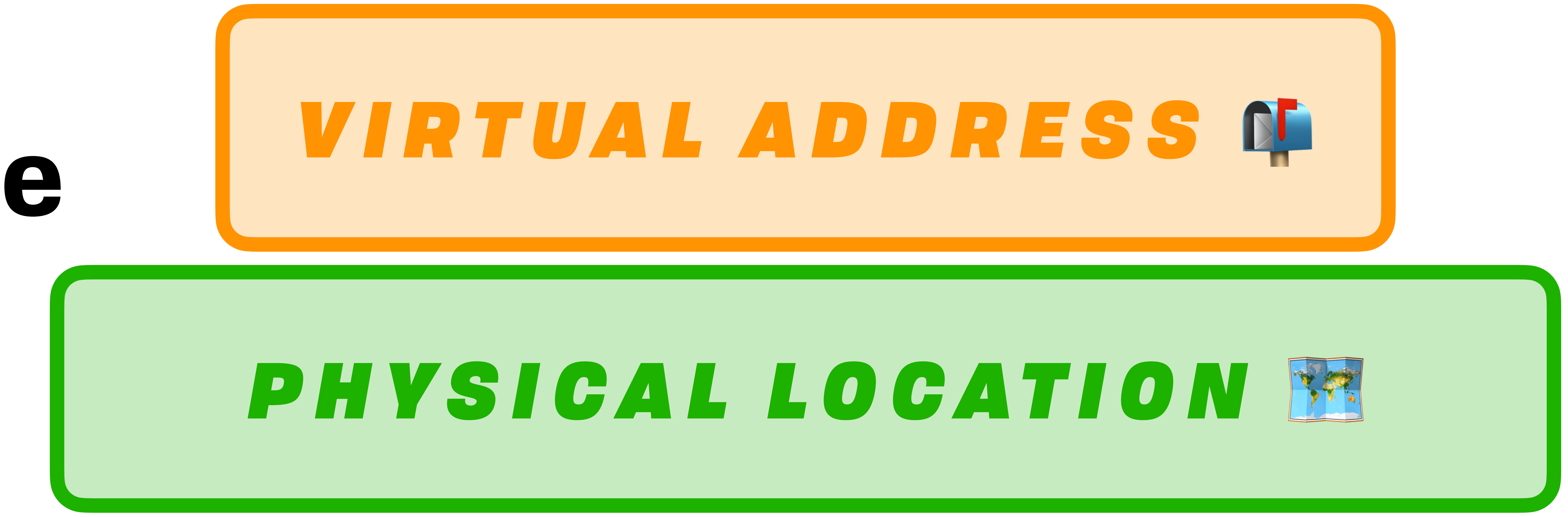
PHYSICAL LOCATION 🗺️

Towards a Solution ⚗️✨

Consistent Keys

```
%{hash(content) => content}
```

- Above virtual address
- Focused: **data itself**
  - Same for **everyone & everywhere**
  - Perfect for caching
- Immutable data++
  - Consistent pointers → consistent data



VIRTUAL ADDRESS 📫

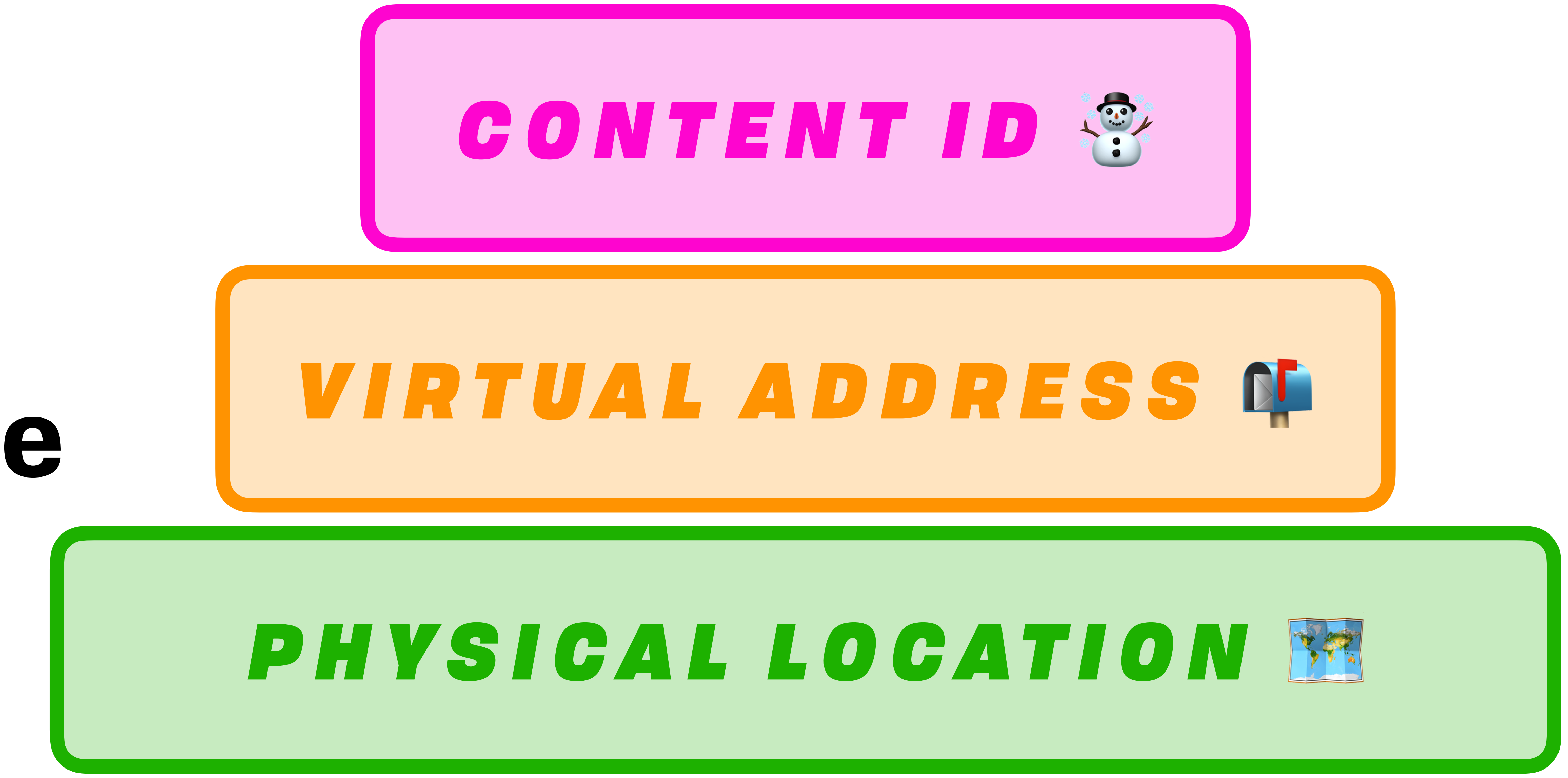
PHYSICAL LOCATION 🗺️

Towards a Solution 🛸✨

Consistent Keys

```
%{hash(content) => content}
```

- Above virtual address
- Focused: **data itself**
  - Same for **everyone & everywhere**
  - Perfect for caching
- Immutable data++
  - Consistent pointers → consistent data



Towards a Solution ⚗️✨

Hash-Based Relationships

Towards a Solution ⚗️✨

Hash-Based Relationships

CID ~ Data PID

```
{
  Qm123456…: {
    data: nil,
    links: [
      {name: “company”, hash: Qmabc…}
      {name: “industry”, hash: Qmzyx…}
    ]
  }
}
```

Towards a Solution 🔮✨

Hash-Based Relationships

```
{
  Qm123456…: {
    data: nil,
    links: [
      {name: “company”, hash: Qmabc…}
      {name: “industry”, hash: Qmzyx…}
    ]
  }
}
```

CID ~ Data PID

```
{
  Qmabcdef…: {
    data: “Fission”,
    links: [
      {name: “city”,  hash:
Qm1gb…},
      {name: “about”, hash: Qm0eN…}
    ]
  }
}
```

Towards a Solution 🔮✨

Hash-Based Relationships

```
{
  Qm123456…: {
    data: nil,
    links: [
      {name: “company”, hash: Qmabc…}
      {name: “industry”, hash: Qmzyx…}
    ]
  }
}
```

CID ~ Data PID

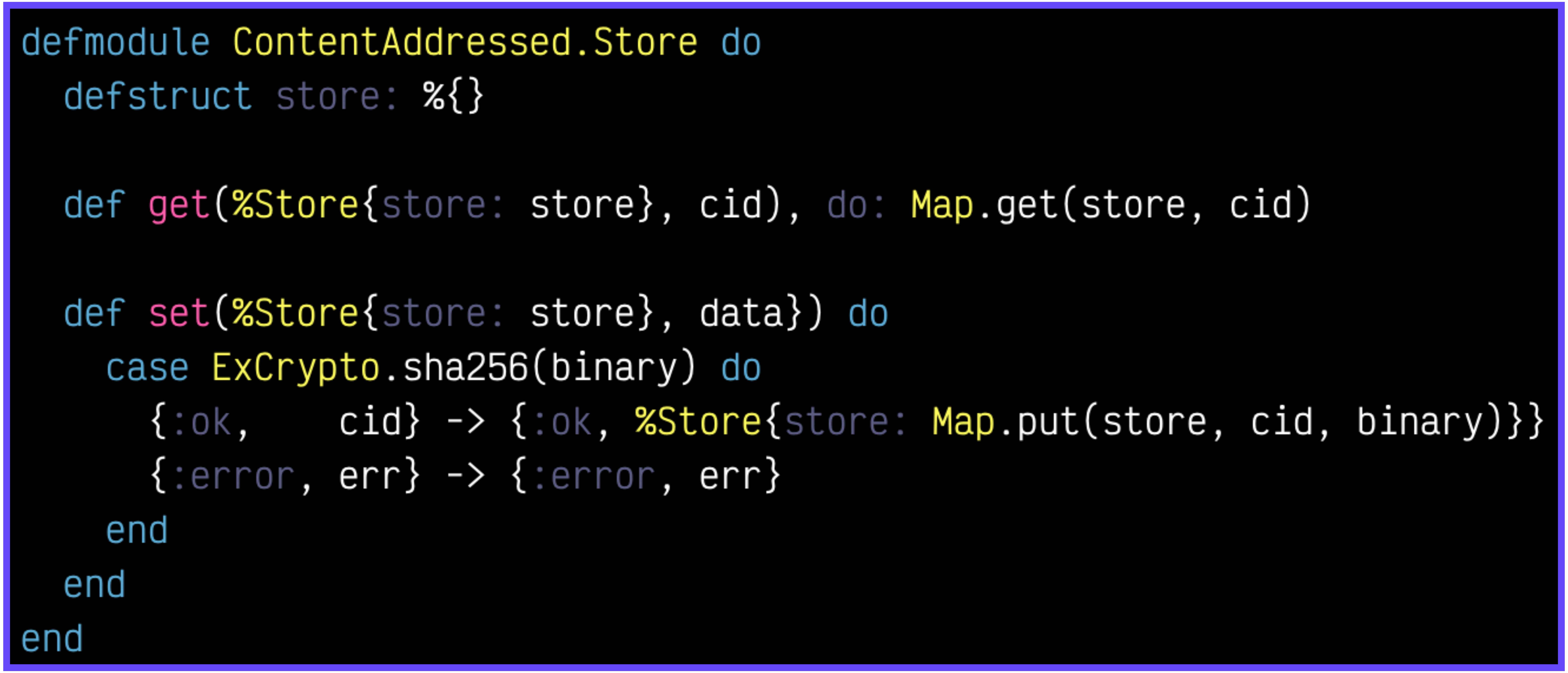
```
{
  Qmabcdef…: {
    data: “Fission”,
    links: [
      {name: “city”,  hash:
Qm1gb…},
      {name: “about”, hash: Qm0eN…}
    ]
  }
}
```

Qm123456…/company/about/ceo

⇒ “Boris Mann”

Towards a Solution 🔮✨

Content IDs Are Easy



```
defmodule ContentAddressed.Store do
  defstruct store: %{}

  def get(%Store{store: store}, cid), do: Map.get(store, cid)

  def set(%Store{store: store}, data}) do
    case ExCrypto.sha256(binary) do
      {:ok,    cid} -> {:ok, %Store{store: Map.put(store, cid, binary)}}
      {:error, err} -> {:error, err}
    end
  end
end
```

Towards a Solution ⚗️✨

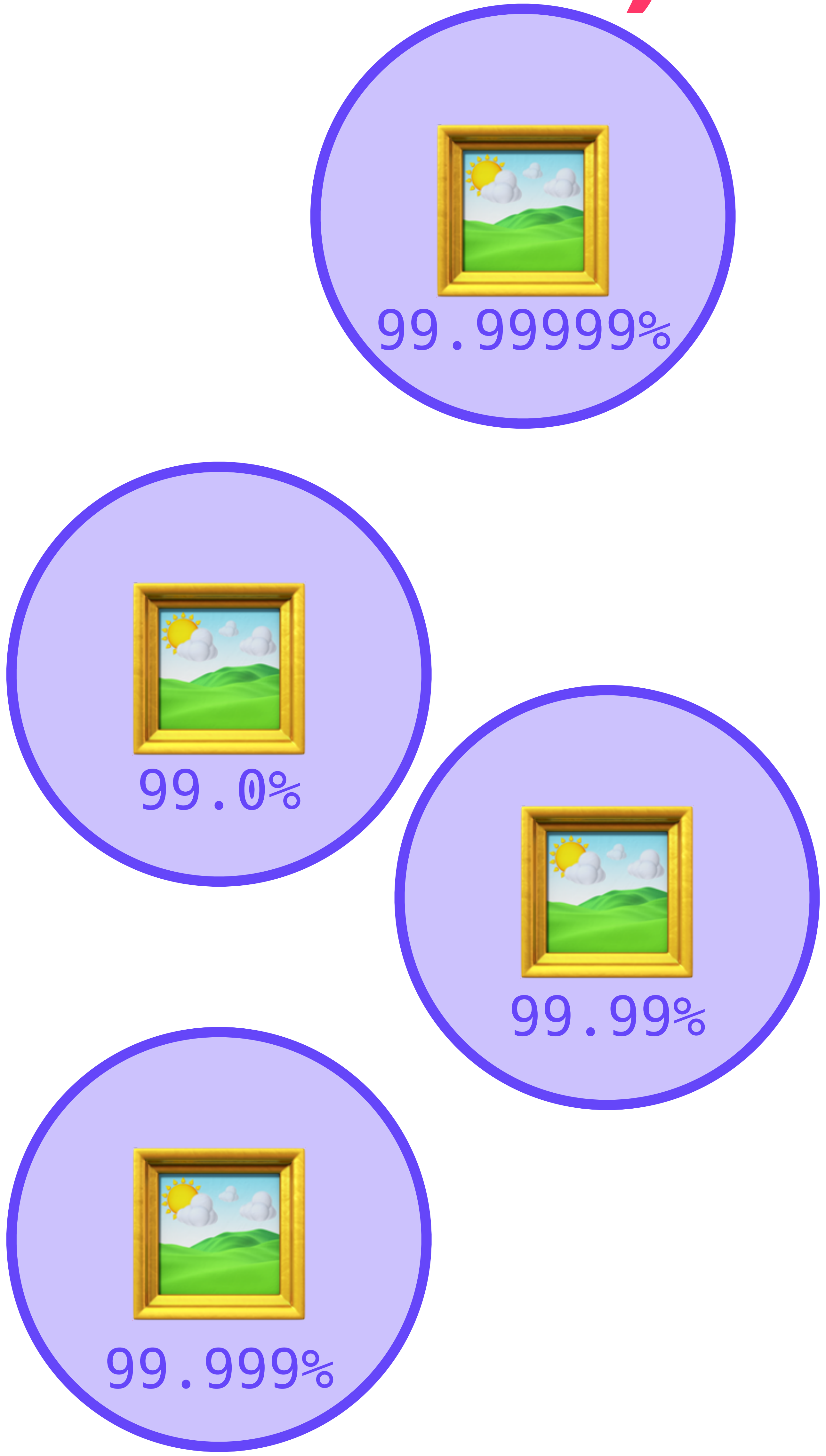
Decoupling, Abundance, Redundancy

Towards a Solution 🔮✨

Decoupling, Abundance, Redundancy

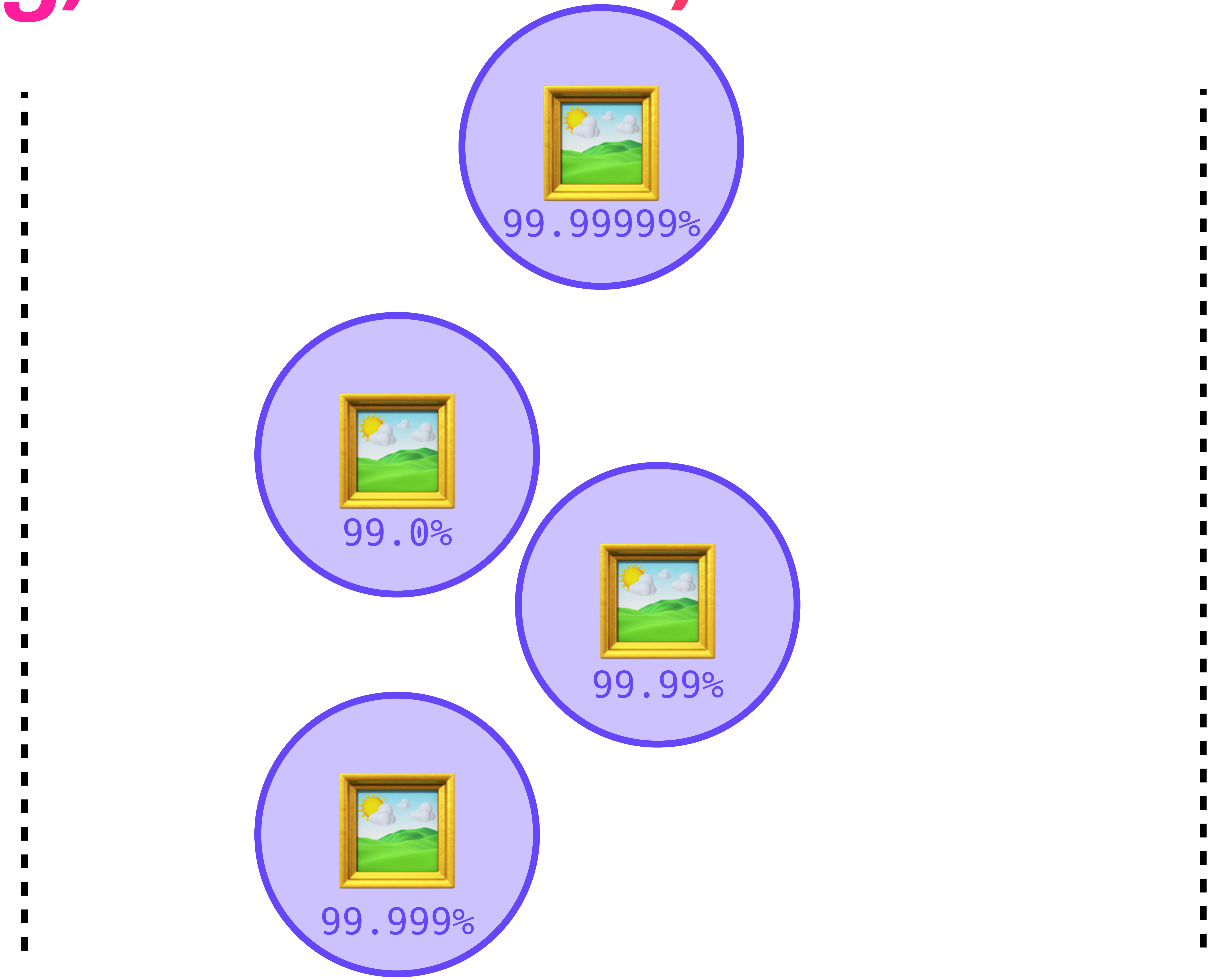
Towards a Solution 🔮✨

Decoupling, Abundance, Redundancy



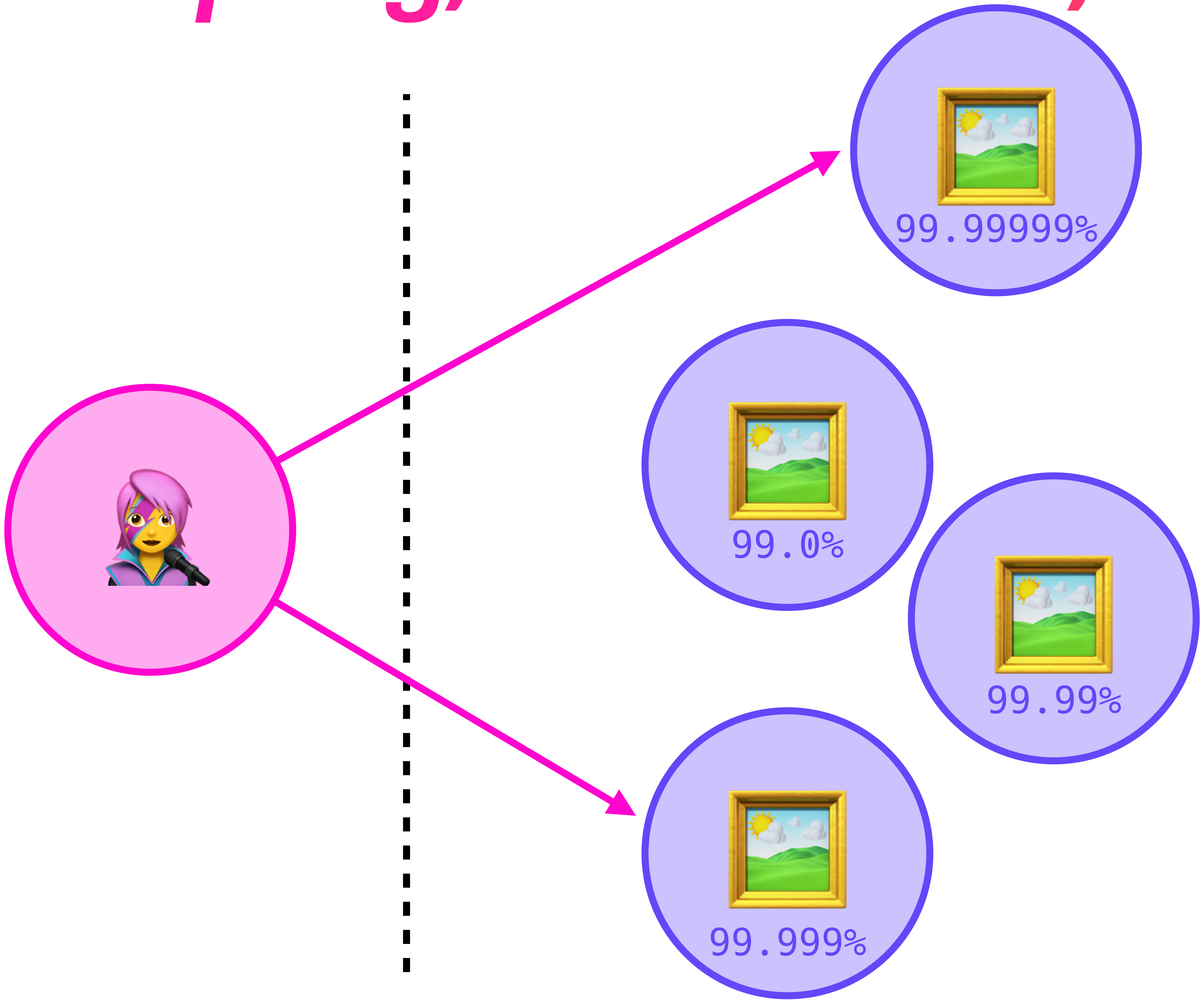
Towards a Solution 🔮✨

Decoupling, Abundance, Redundancy



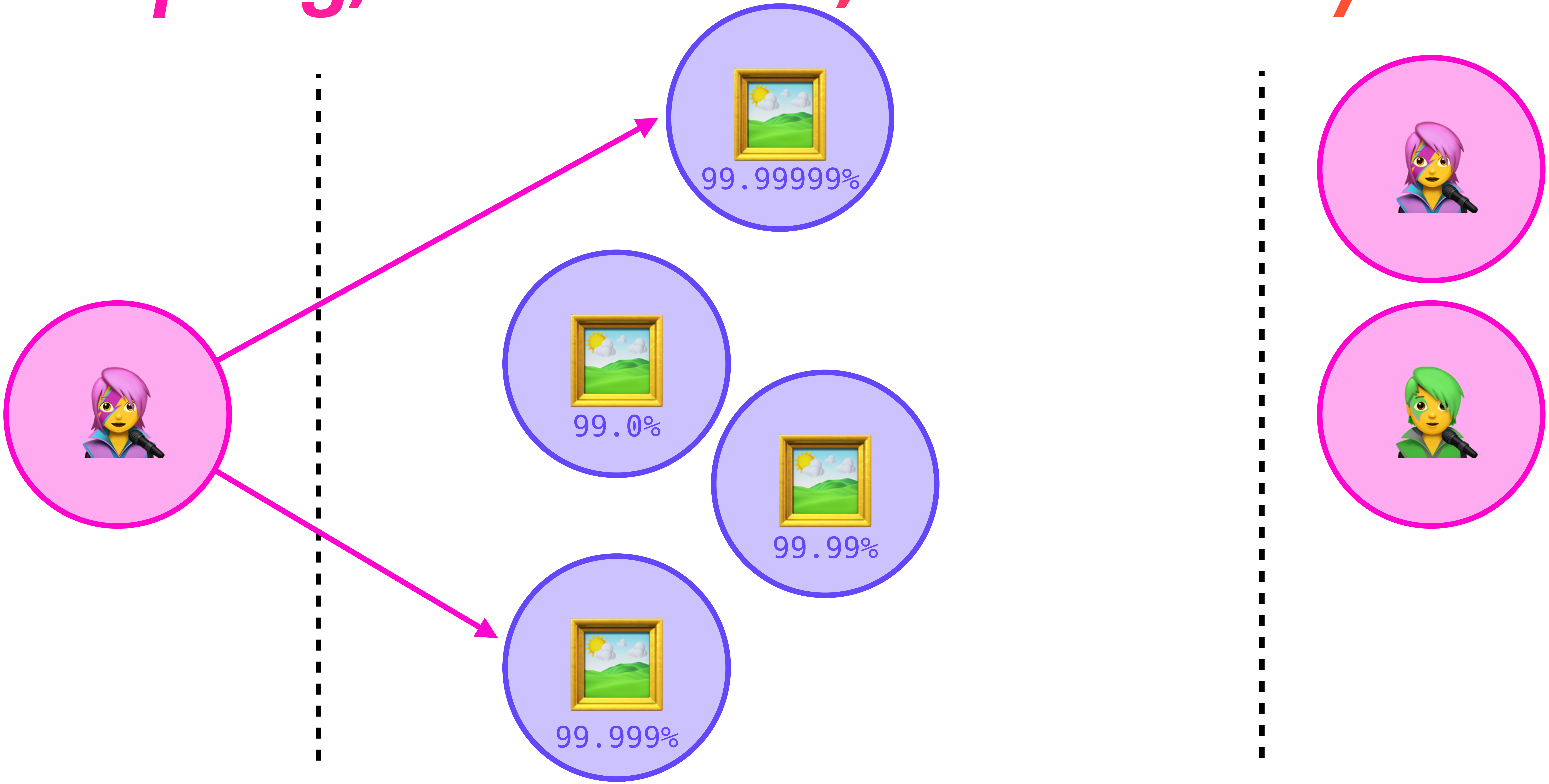
Towards a Solution ⚗️✨

Decoupling, Abundance, Redundancy



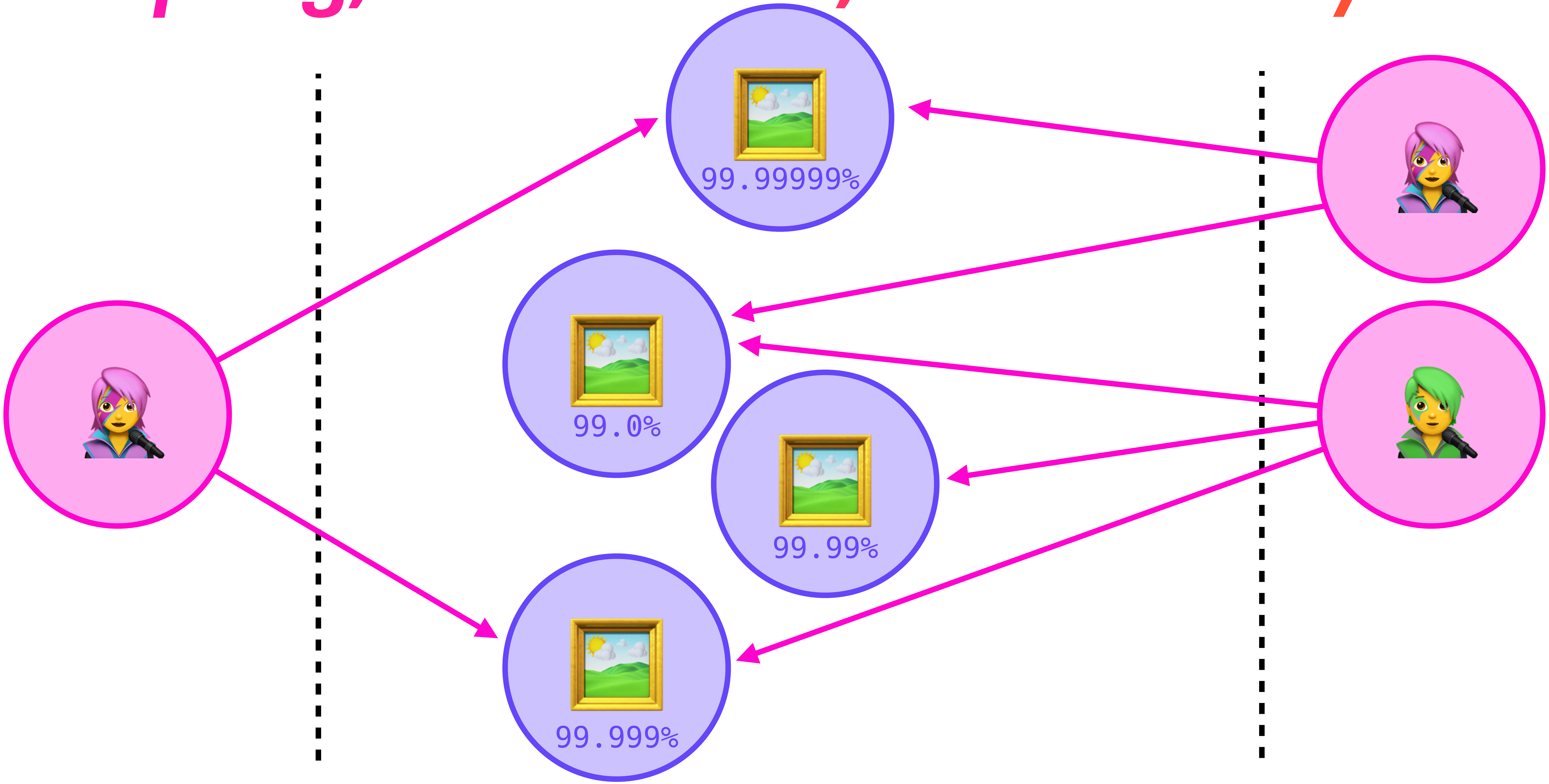
Towards a Solution 🔮✨

Decoupling, Abundance, Redundancy



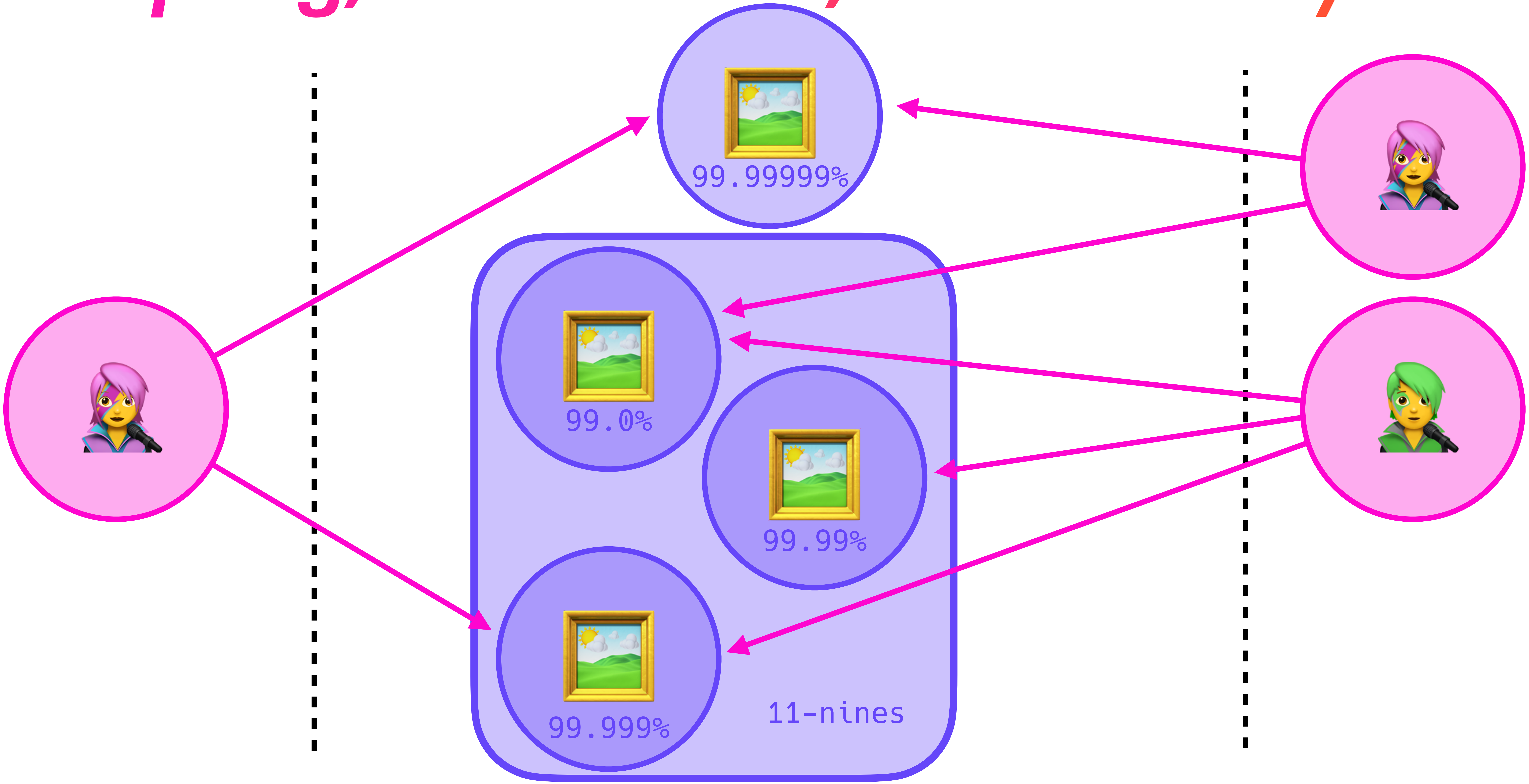
Towards a Solution 🔮✨

Decoupling, Abundance, Redundancy



Towards a Solution 🔮✨

Decoupling, Abundance, Redundancy

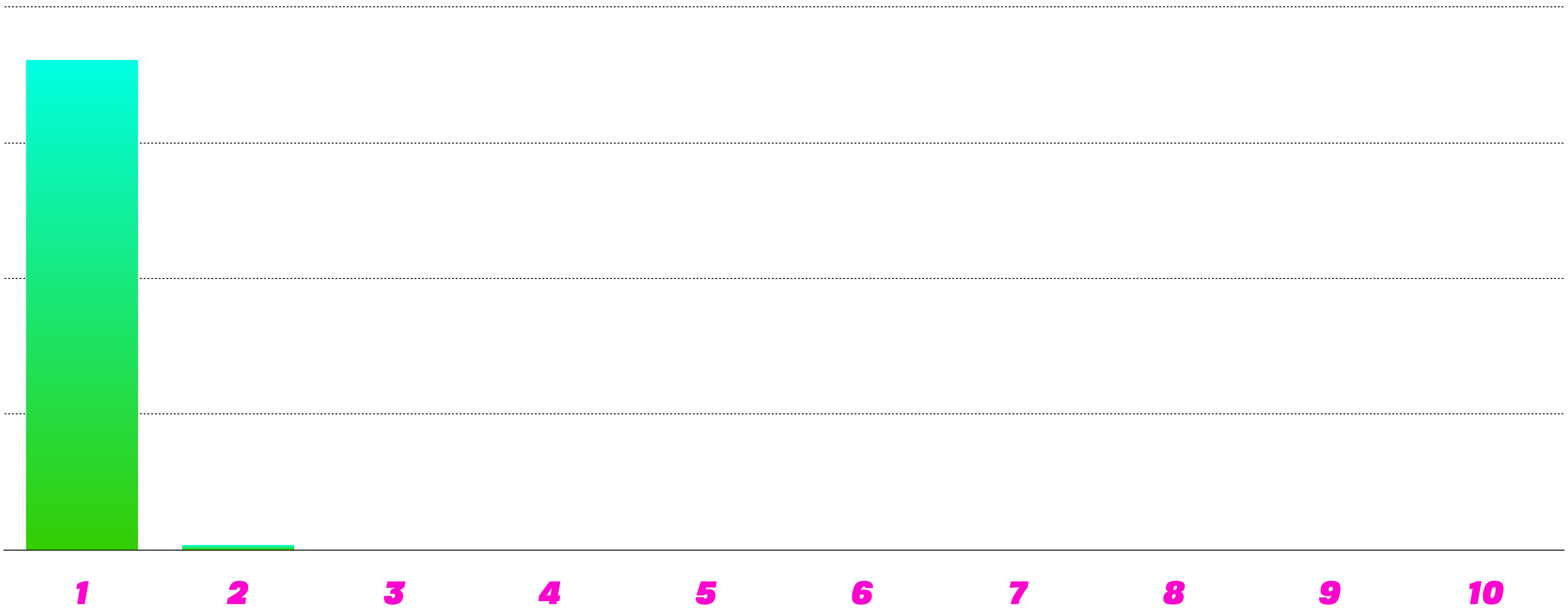


Towards a Solution 🍵✨

Reliability from Unreliable Components

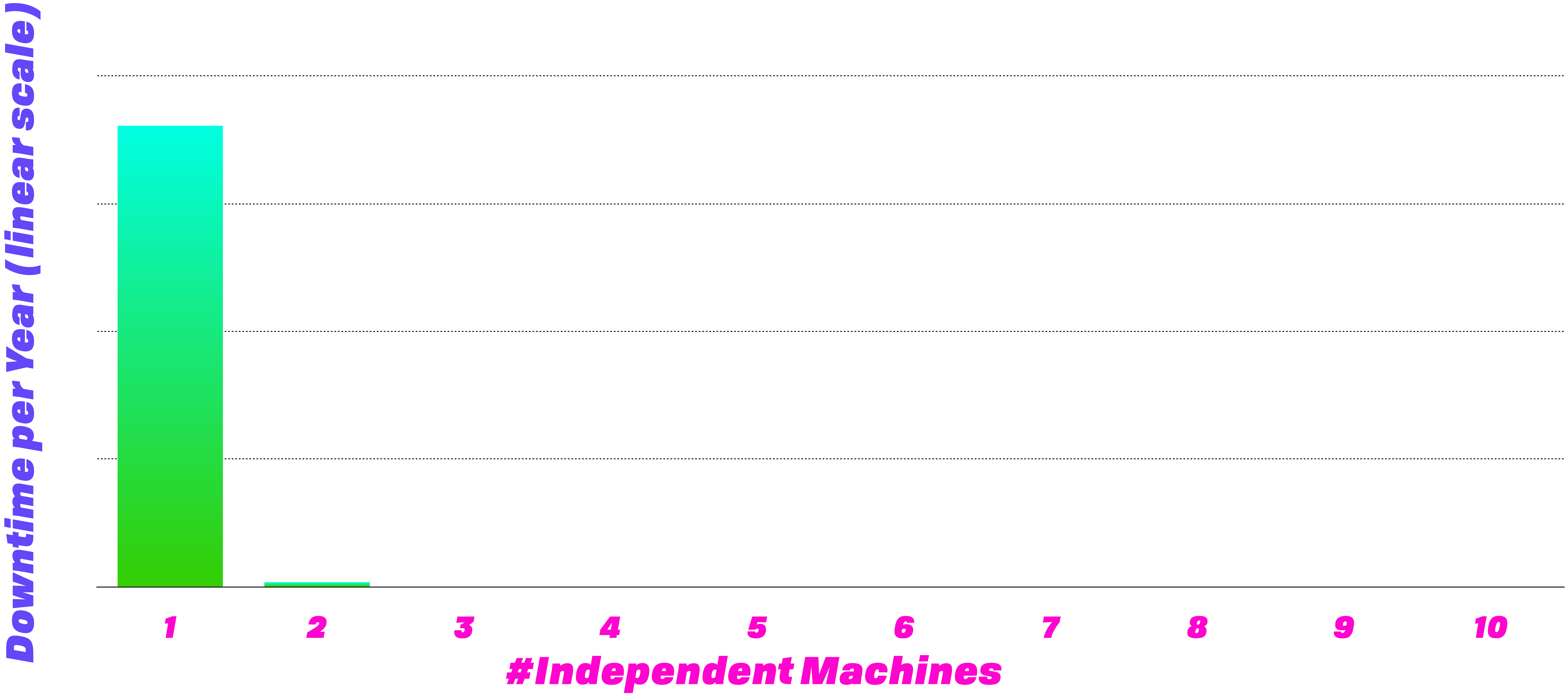
Towards a Solution 🏺✨

Reliability from Unreliable Components



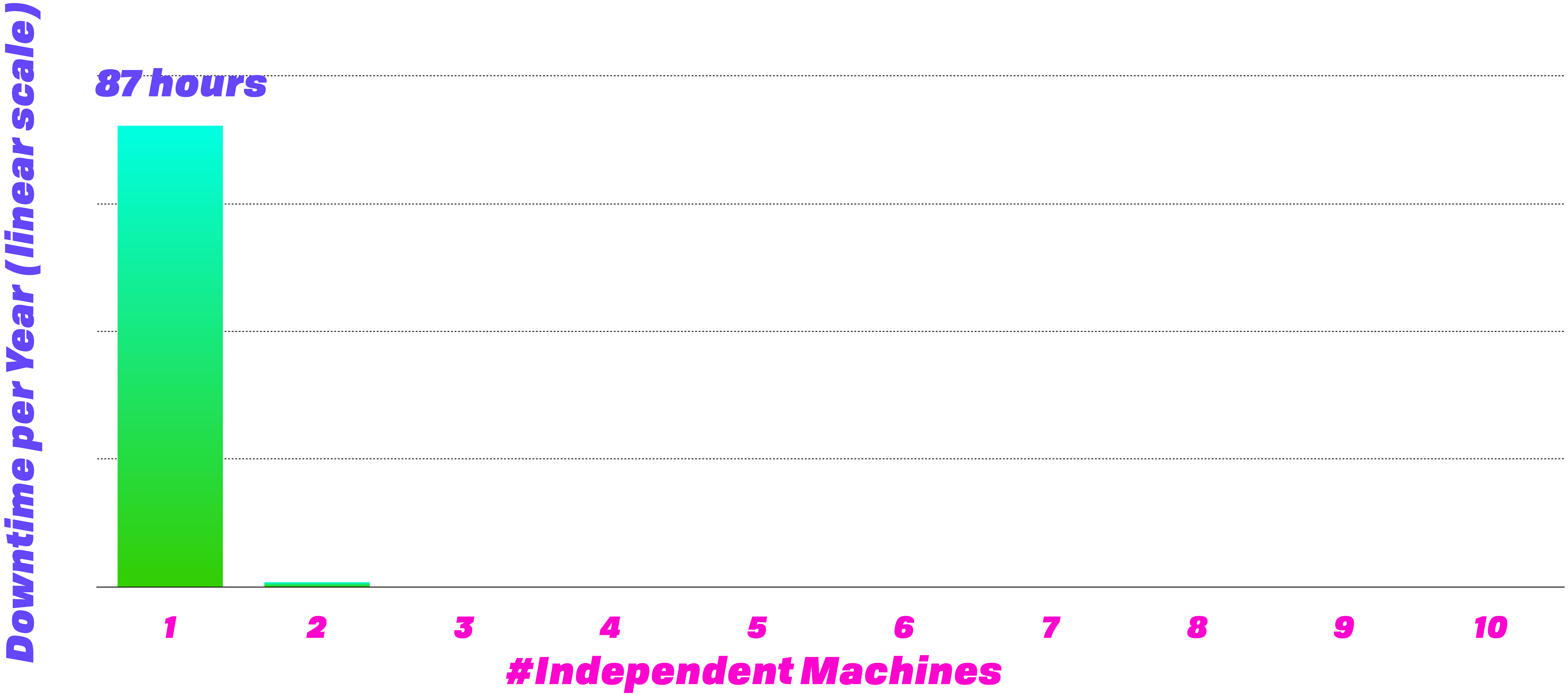
Towards a Solution ⚗️✨

Reliability from Unreliable Components



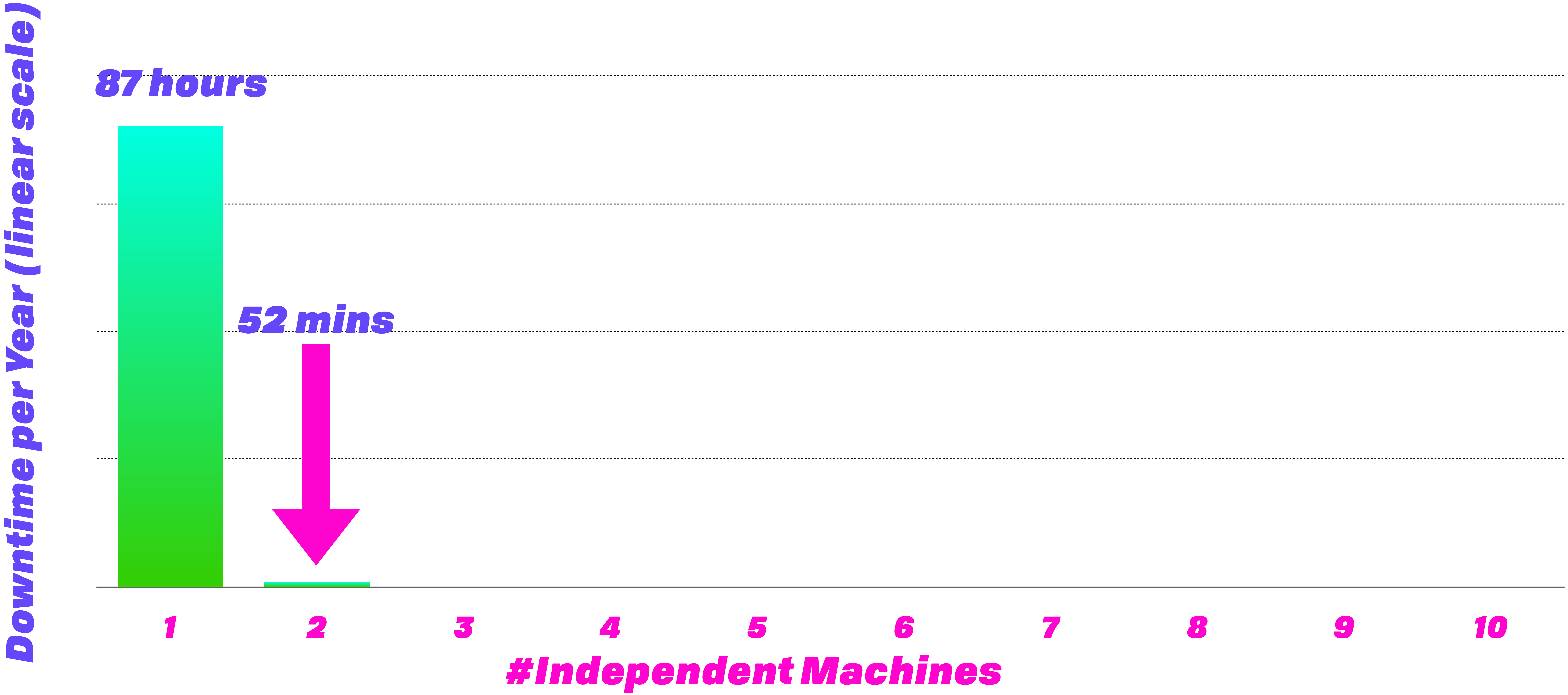
Towards a Solution ⚗️✨

Reliability from Unreliable Components



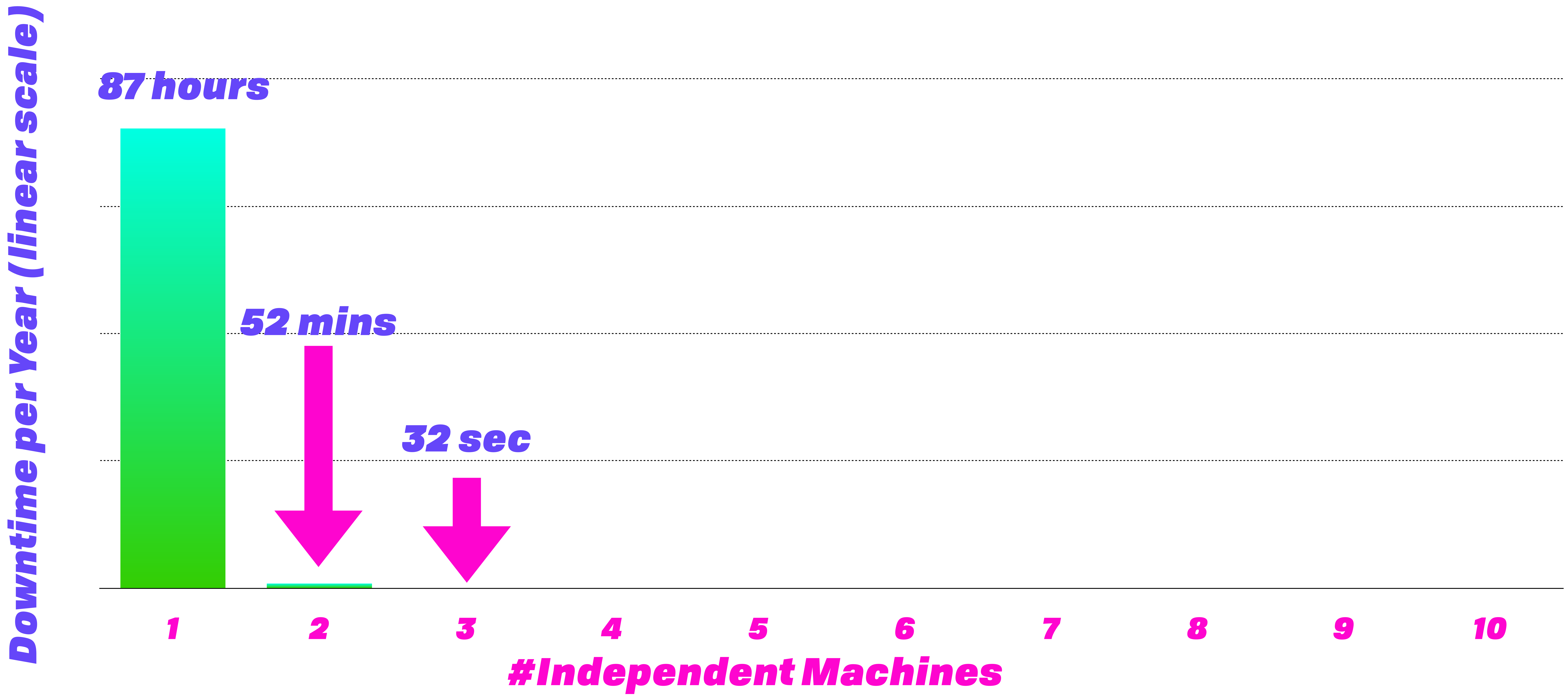
Towards a Solution ⚗️✨

Reliability from Unreliable Components



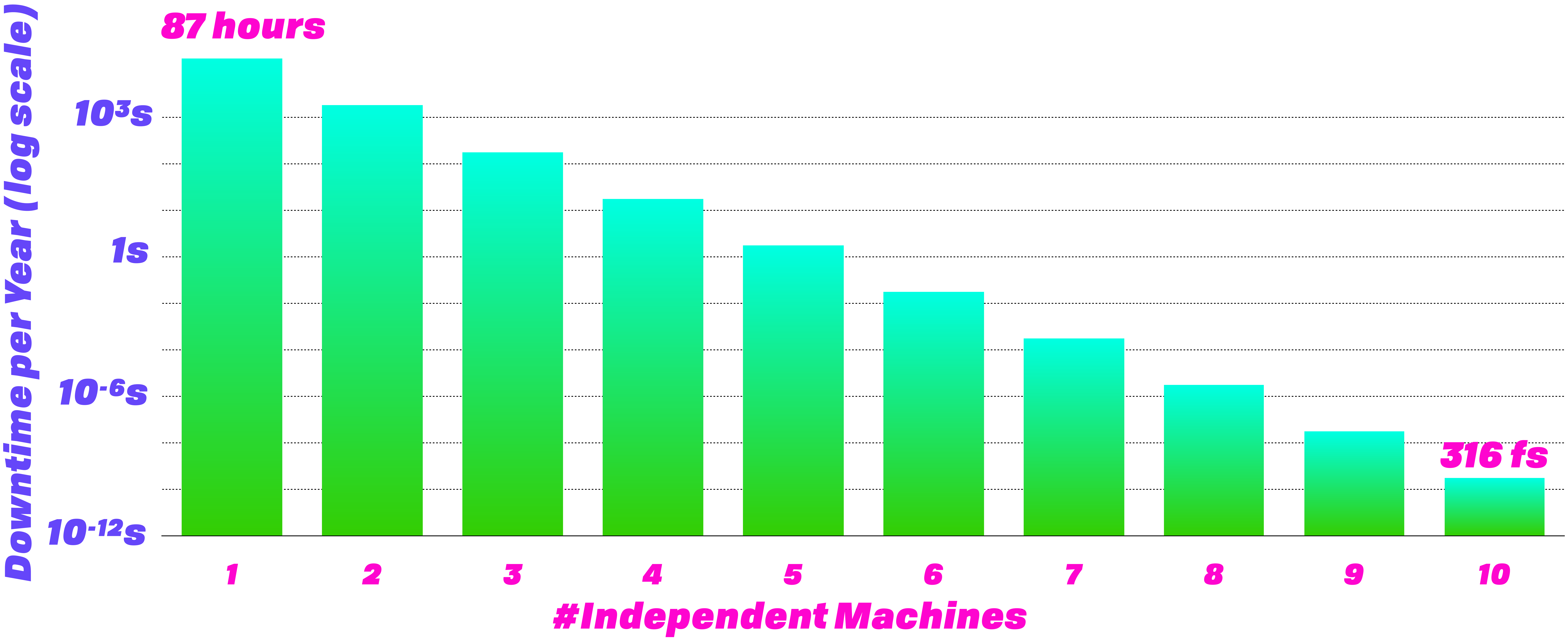
Towards a Solution ⚗️✨

Reliability from Unreliable Components



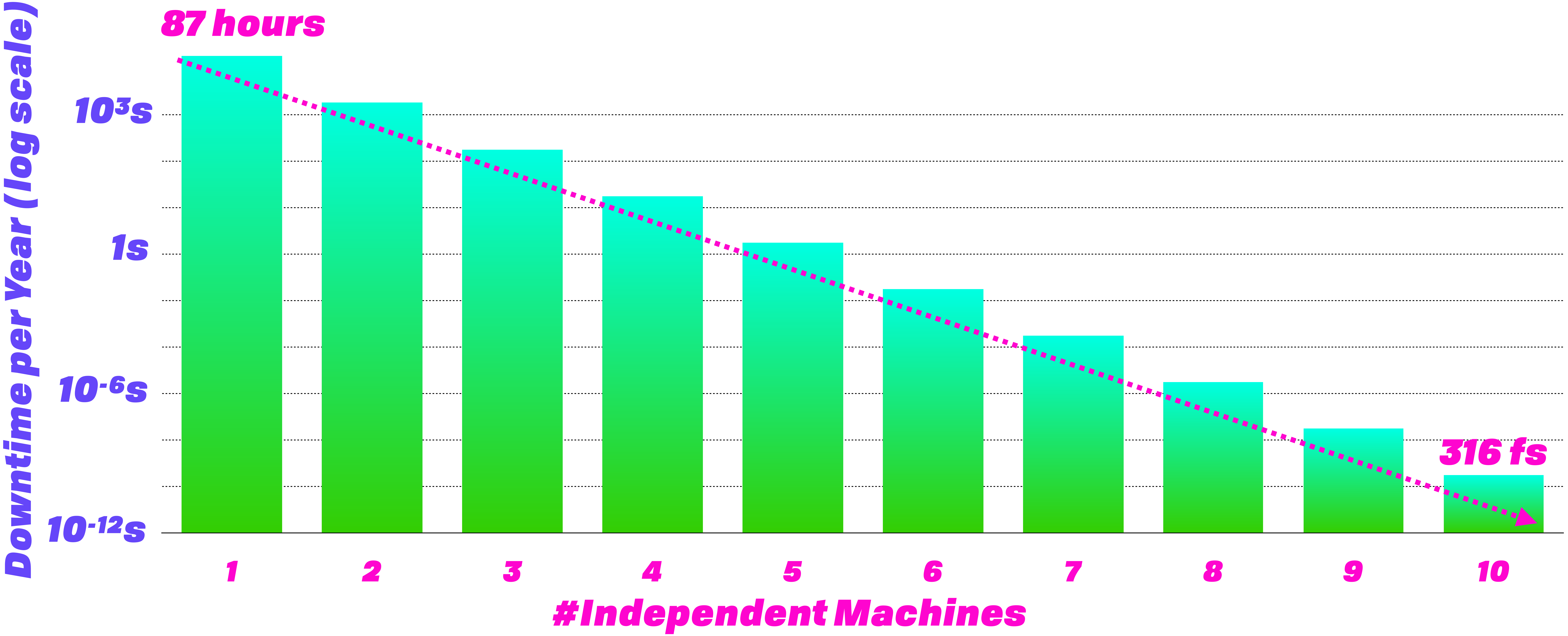
Towards a Solution ⚗️✨

Reliability from Unreliable Components



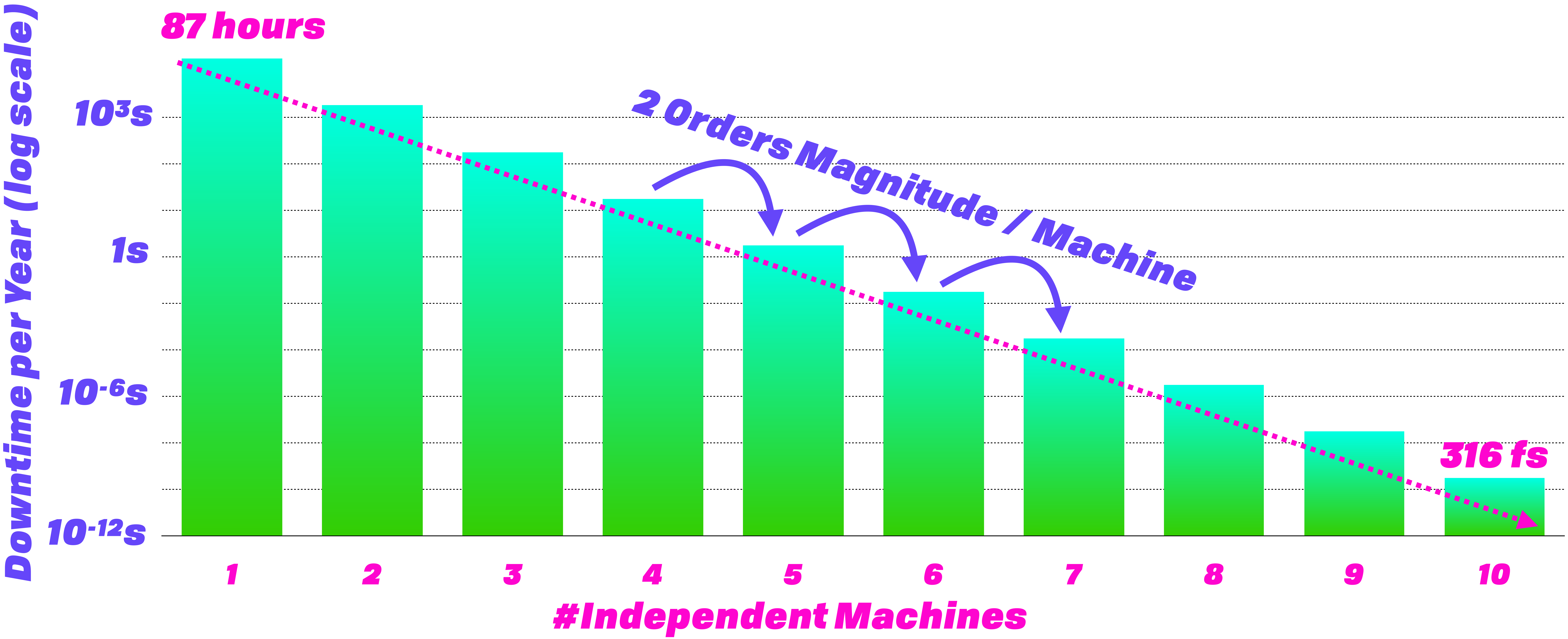
Towards a Solution ⚗️✨

Reliability from Unreliable Components



Towards a Solution 🪩✨

Reliability from Unreliable Components



Towards a Solution ⚗️✨

Why Sync Whole Tables?

Towards a Solution 🔮✨

Why Sync Whole Tables?

user_id	username	company	start_date	inserted_at
1	expede	Fission	AUG-2019	FEB-2020
2	bmann	--	--	OCT-2020

Towards a Solution 🔮✨

Why Sync Whole Tables?

user_id	username	company	start_date	inserted_at
1	expede	Fission	AUG-2019	FEB-2020
2	bmann	--	--	OCT-2020

kb_id	owner_id	mode	switches	inserted_at
42	1	Wireless	Blue	JAN-2020

Towards a Solution 🔮✨

Why Sync Whole Tables?

Who's clock?

Meaningful or coincidence?

user_id	username	company	start_date	inserted_at
1	expede	Fission	AUG-2019	FEB-2020
2	bmann	--	--	OCT-2020

kb_id	owner_id	mode	switches	inserted_at
42	1	Wireless	Blue	JAN-2020

Towards a Solution 🔮✨

Why Sync Whole Tables?

🔒?

Who's clock?

Meaningful or coincidence?

user_id	username	company	start_date	inserted_at
1	expede	Fission	AUG-2019	FEB-2020
2	bmann	--	--	OCT-2020

kb_id	owner_id	mode	switches	inserted_at
42	1	Wireless	Blue	JAN-2020

Towards a Solution 🔮✨

Why Sync Whole Tables?

🔒? 🪵?

Who's clock?

Meaningful or coincidence?

user_id	username	company	start_date	inserted_at
1	expede	Fission	AUG-2019	FEB-2020
2	bmann	--	--	OCT-2020

kb_id	owner_id	mode	switches	inserted_at
42	1	Wireless	Blue	JAN-2020

Towards a Solution 🔮✨

Why Sync Whole Tables?

🔒? 🪵? 😤?

Who's clock?

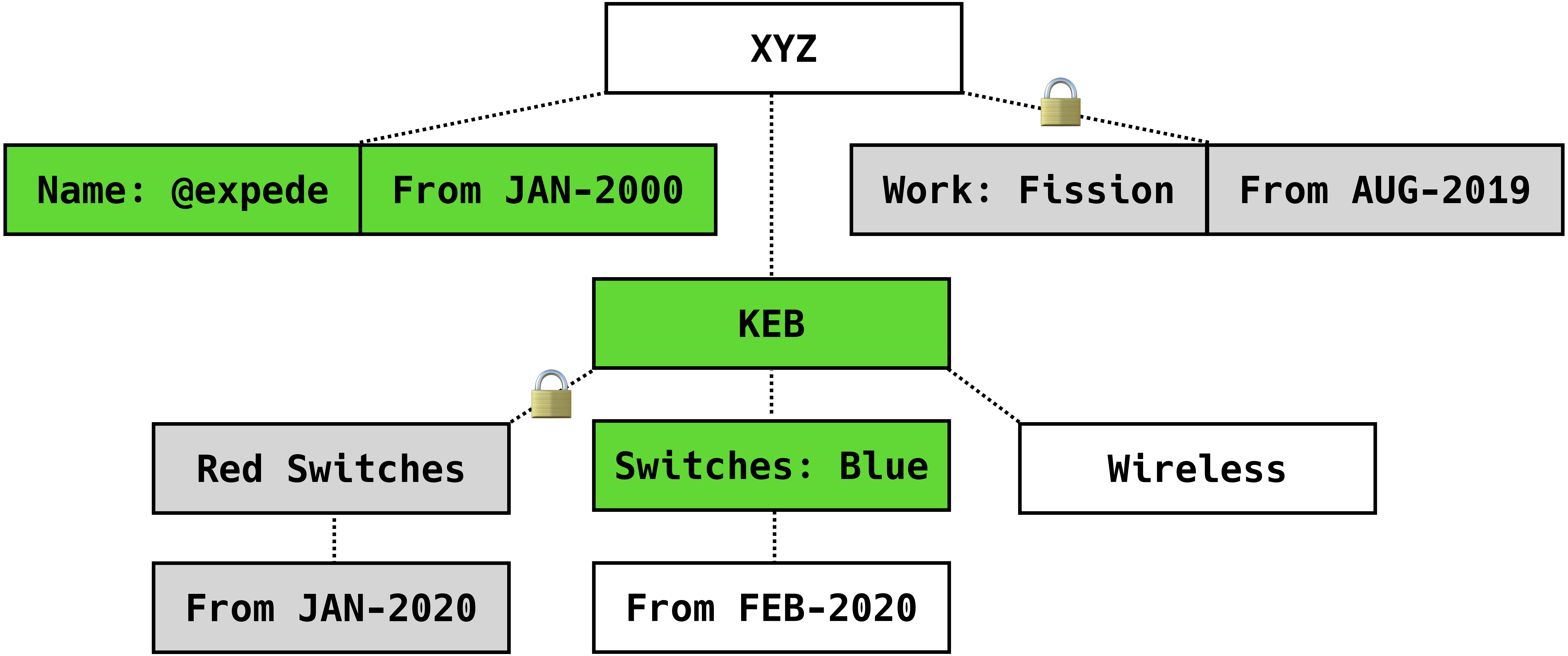
Meaningful or coincidence?

user_id	username	company	start_date	inserted_at
1	expede	Fission	AUG-2019	FEB-2020
2	bmann	--	--	OCT-2020

kb_id	owner_id	mode	switches	inserted_at
42	1	Wireless	Blue	JAN-2020

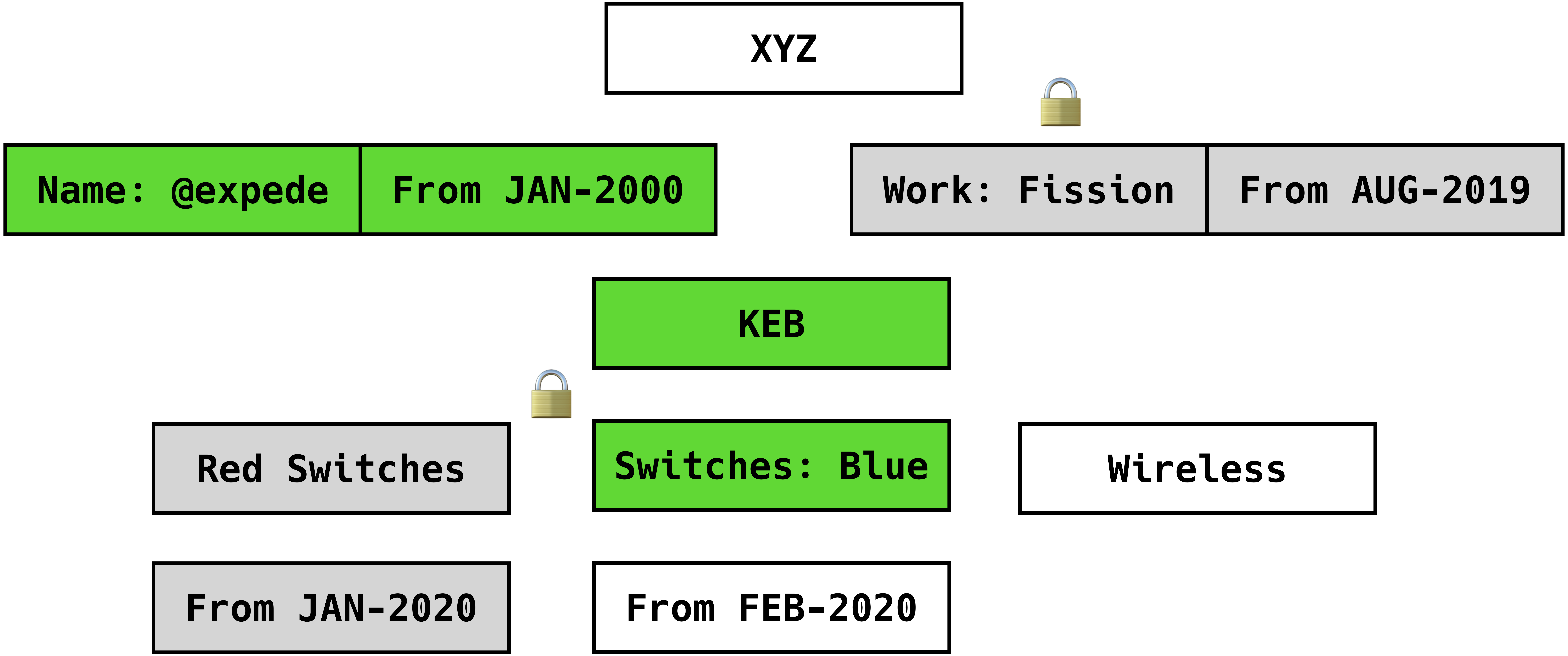
Towards a Solution 🔮✨

Relationships



Towards a Solution 🔮✨

Relationships



XYZ

Name: @expede

From JAN-2000

🔒

Work: Fission

From AUG-2019

KEB

🔒

Red Switches

Switches: Blue

Wireless

From JAN-2020

From FEB-2020

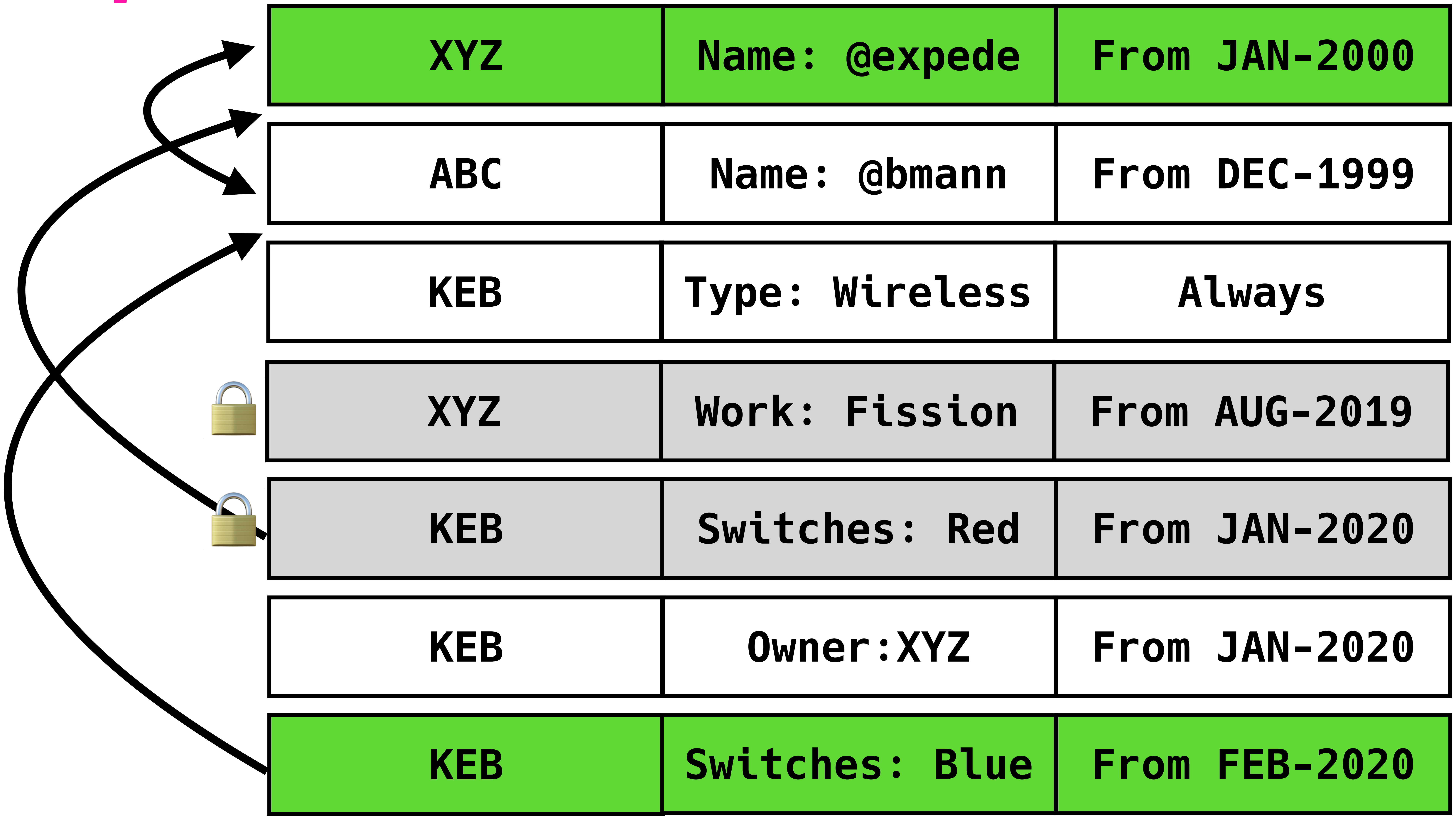
Towards a Solution 🔮✨

A Sequel to SQL: Nonlinear DBs

	XYZ	Name: @expede	From JAN-2000
	ABC	Name: @bmann	From DEC-1999
	KEB	Type: Wireless	Always
🔒	XYZ	Work: Fission	From AUG-2019
🔒	KEB	Switches: Red	From JAN-2020
	KEB	Owner:XYZ	From JAN-2020
	KEB	Switches: Blue	From FEB-2020

Towards a Solution 🔮✨

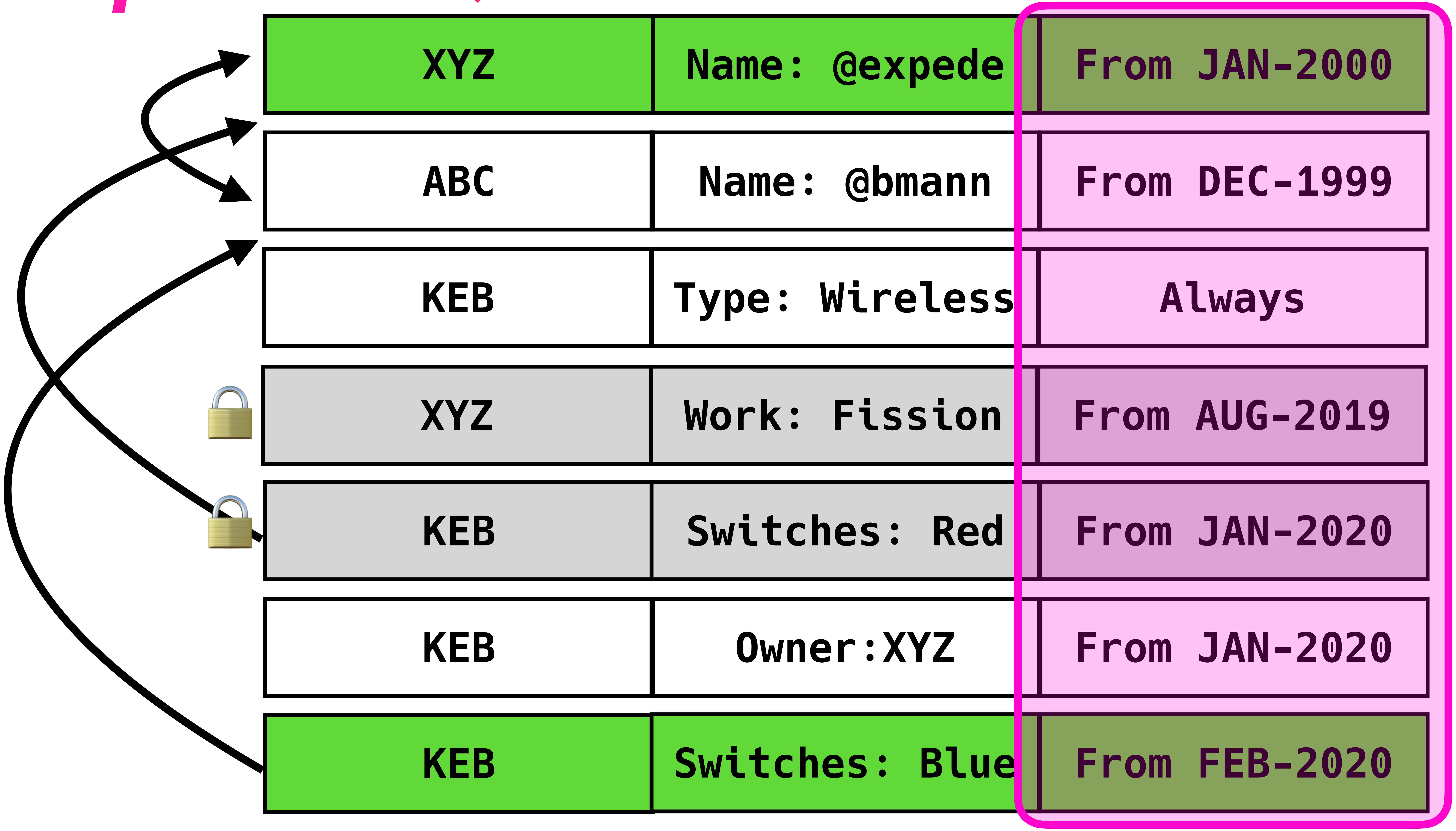
A Sequel to SQL: Nonlinear DBs



	XYZ	Name: @expede	From JAN-2000
	ABC	Name: @bmann	From DEC-1999
	KEB	Type: Wireless	Always
🔒	XYZ	Work: Fission	From AUG-2019
🔒	KEB	Switches: Red	From JAN-2020
	KEB	Owner:XYZ	From JAN-2020
	KEB	Switches: Blue	From FEB-2020

Towards a Solution 🔮✨

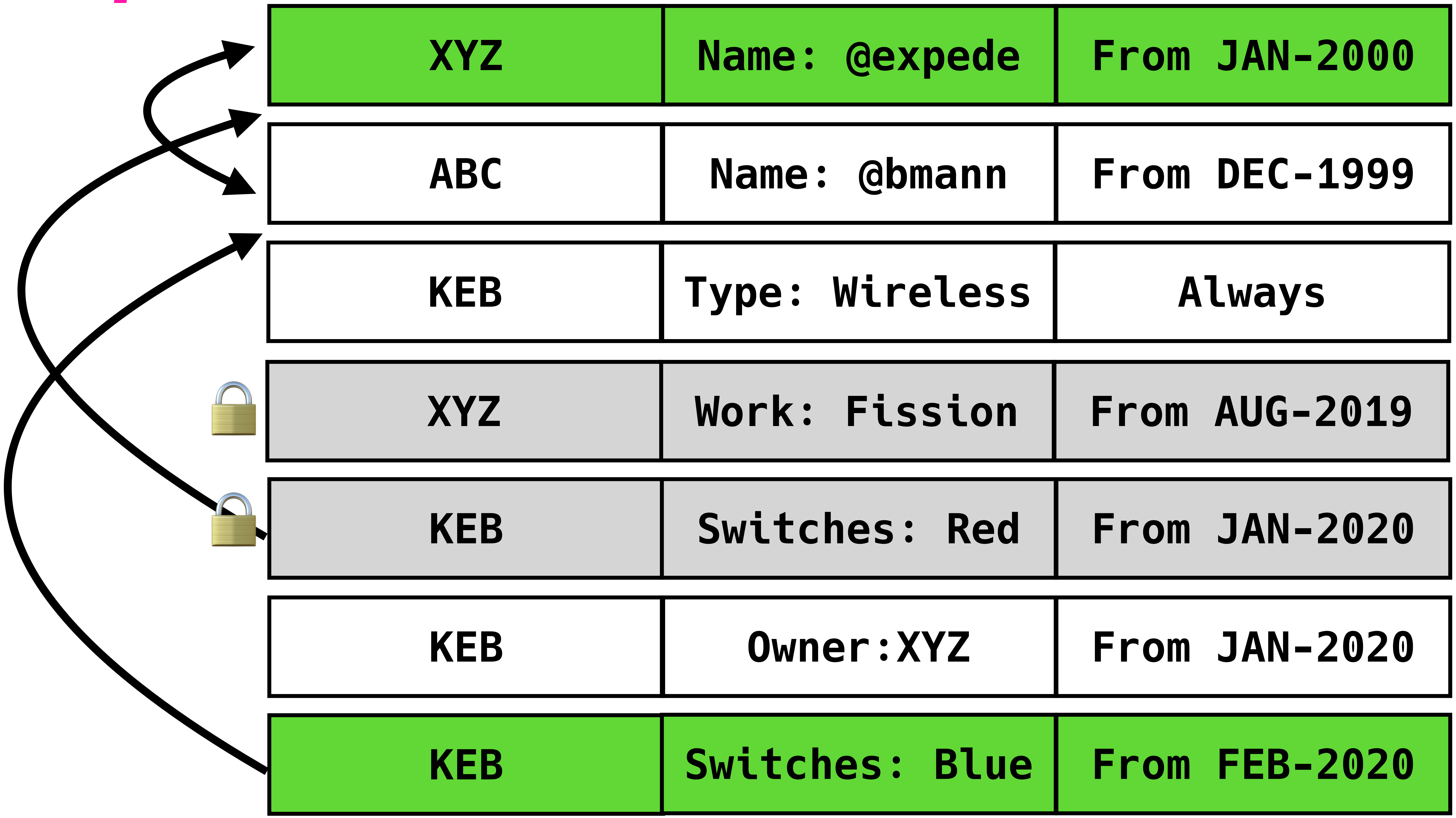
A Sequel to SQL: Nonlinear DBs



	XYZ	Name: @expede	From JAN-2000
	ABC	Name: @bmann	From DEC-1999
	KEB	Type: Wireless	Always
🔒	XYZ	Work: Fission	From AUG-2019
🔒	KEB	Switches: Red	From JAN-2020
	KEB	Owner:XYZ	From JAN-2020
	KEB	Switches: Blue	From FEB-2020

Towards a Solution ⚗️✨

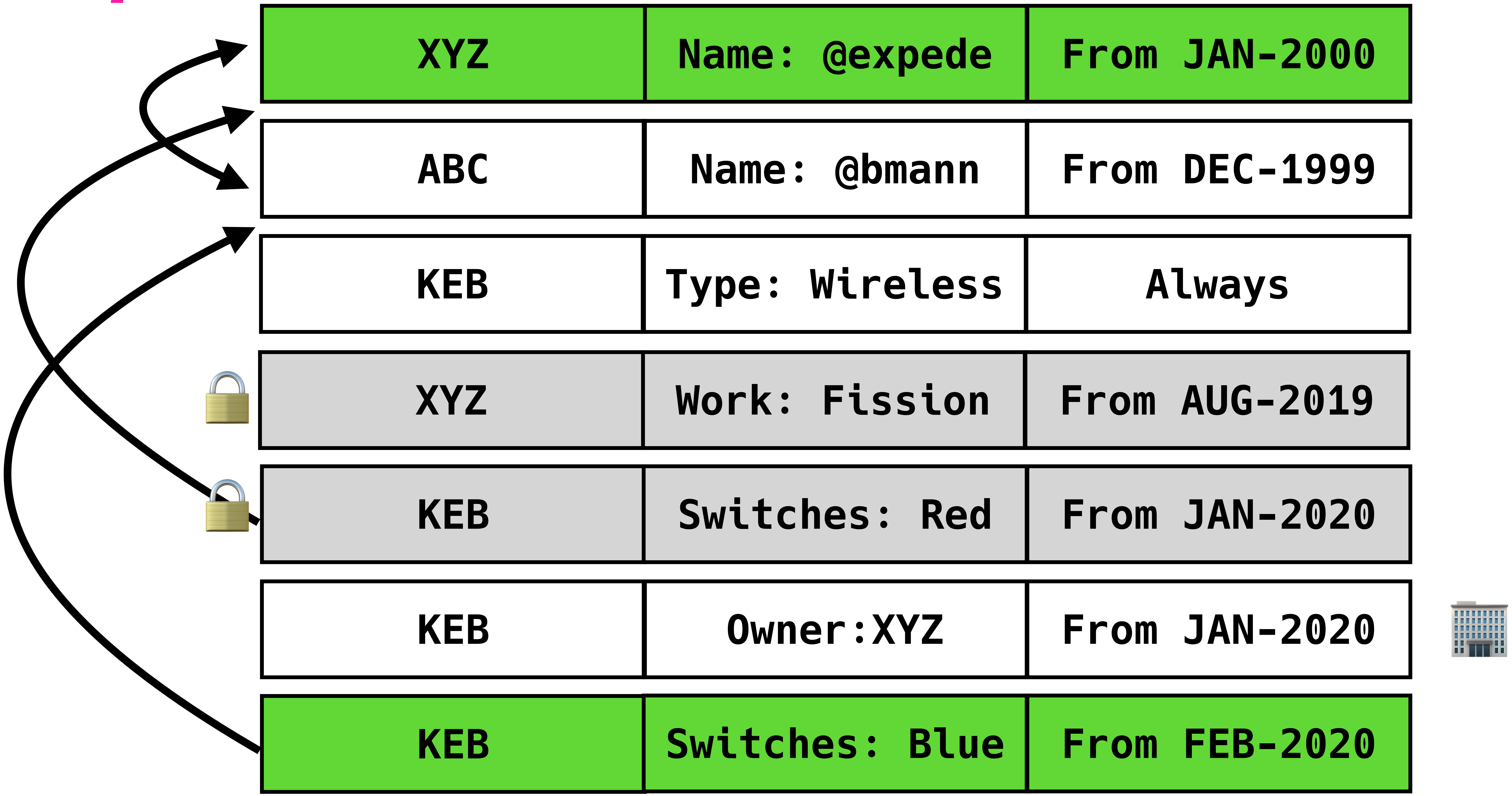
A Sequel to SQL: Nonlinear DBs



	XYZ	Name: @expede	From JAN-2000
	ABC	Name: @bmann	From DEC-1999
	KEB	Type: Wireless	Always
🔒	XYZ	Work: Fission	From AUG-2019
🔒	KEB	Switches: Red	From JAN-2020
	KEB	Owner:XYZ	From JAN-2020
	KEB	Switches: Blue	From FEB-2020

Towards a Solution 🔮✨

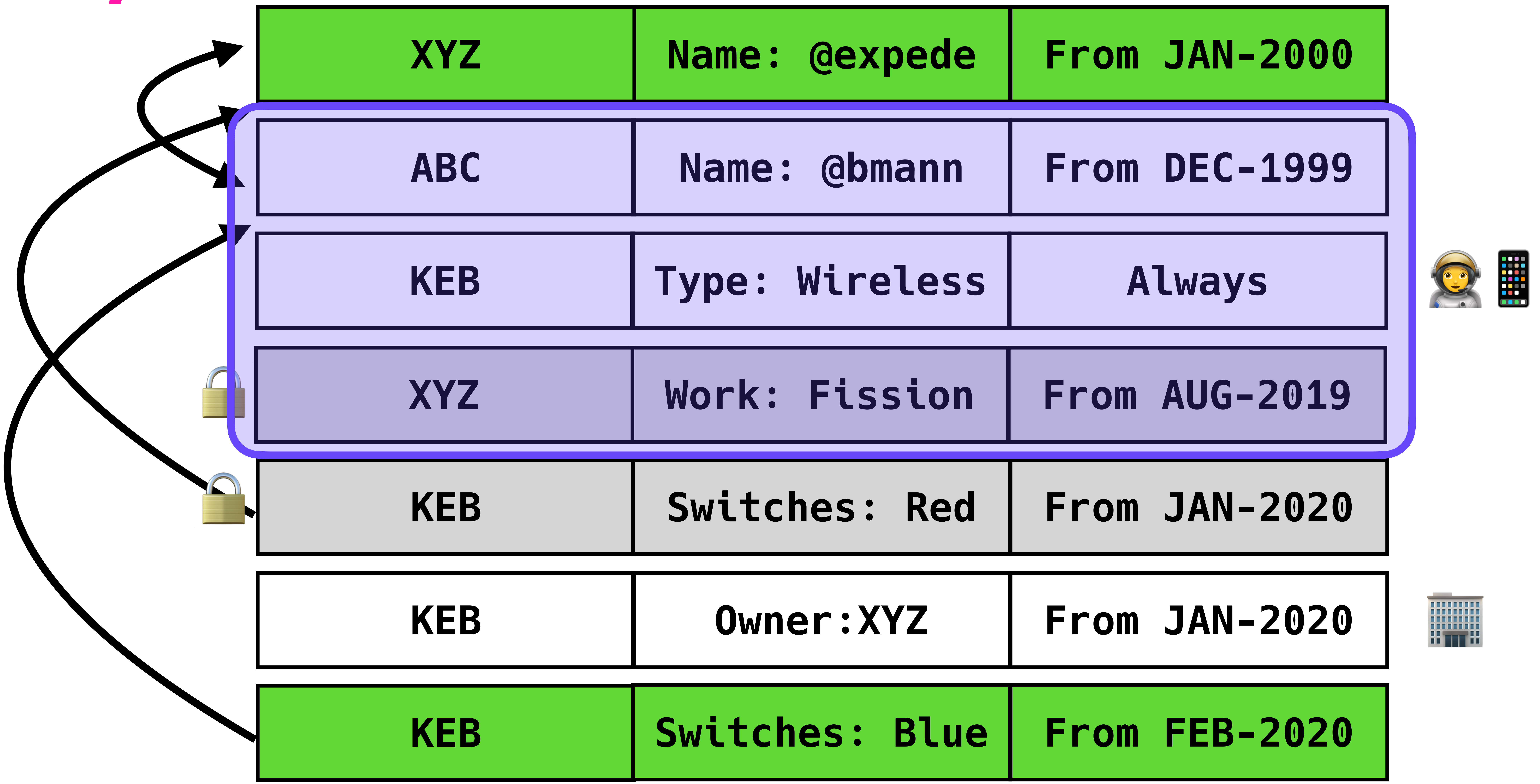
A Sequel to SQL: Nonlinear DBs



	XYZ	Name: @expede	From JAN-2000	
	ABC	Name: @bmann	From DEC-1999	
	KEB	Type: Wireless	Always	
🔒	XYZ	Work: Fission	From AUG-2019	
🔒	KEB	Switches: Red	From JAN-2020	
	KEB	Owner:XYZ	From JAN-2020	🏢
	KEB	Switches: Blue	From FEB-2020	

Towards a Solution ✨

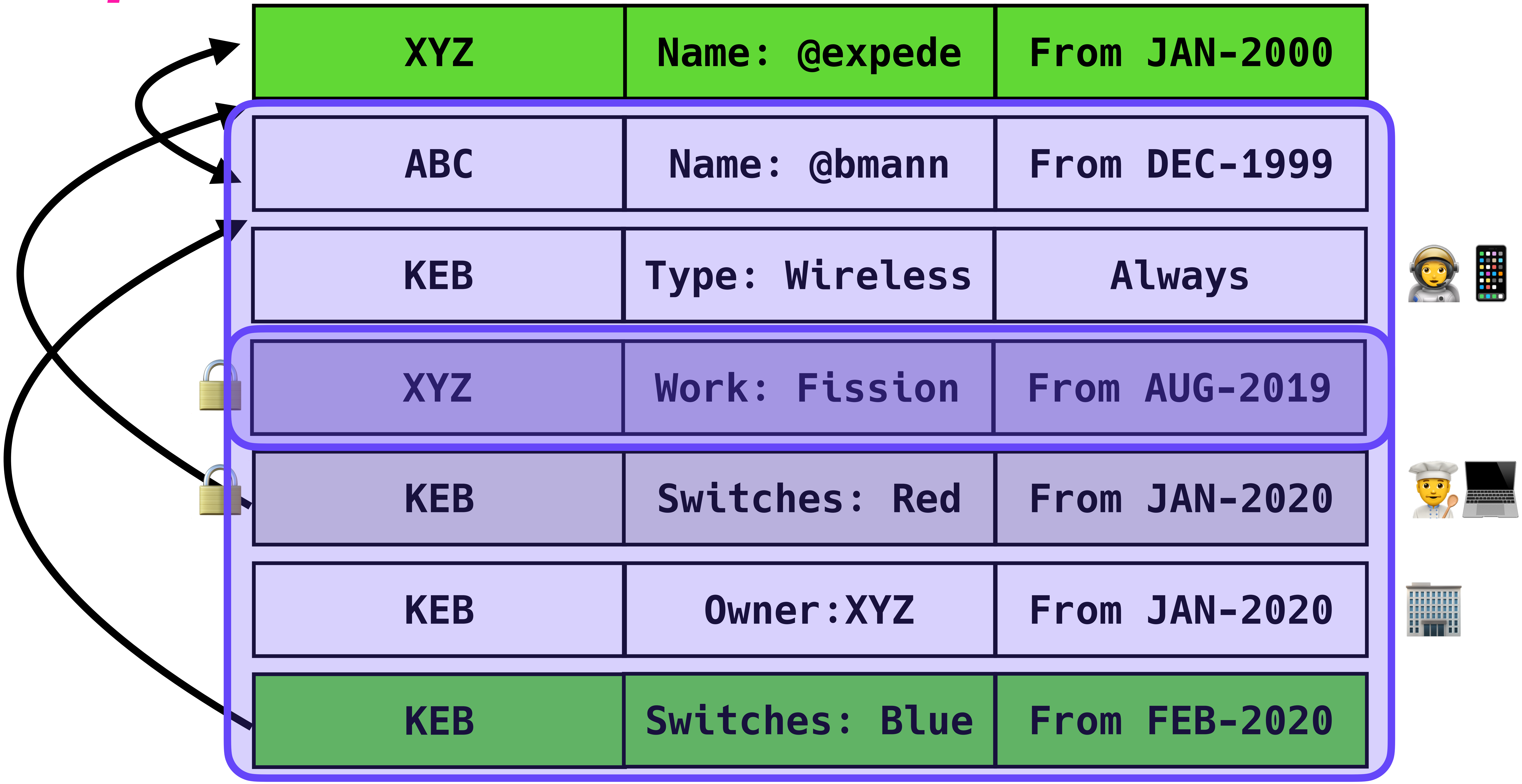
A Sequel to SQL: Nonlinear DBs



XYZ	Name: @expede	From JAN-2000
ABC	Name: @bmann	From DEC-1999
KEB	Type: Wireless	Always
XYZ	Work: Fission	From AUG-2019
KEB	Switches: Red	From JAN-2020
KEB	Owner:XYZ	From JAN-2020
KEB	Switches: Blue	From FEB-2020

Towards a Solution ⚗️✨

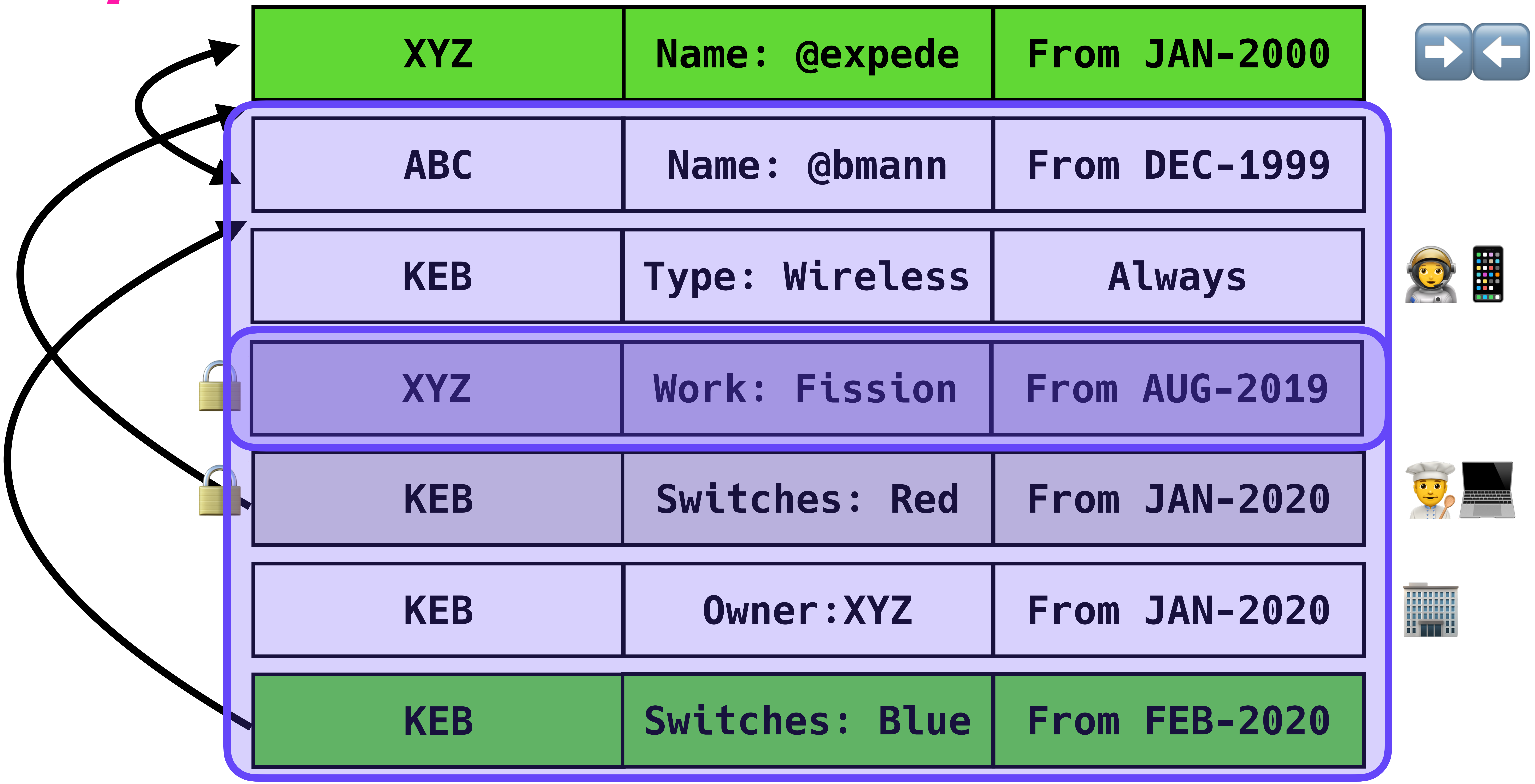
A Sequel to SQL: Nonlinear DBs



	XYZ	Name: @expede	From JAN-2000	
	ABC	Name: @bmann	From DEC-1999	
	KEB	Type: Wireless	Always	🧑‍🚀📱
🔒	XYZ	Work: Fission	From AUG-2019	
🔒	KEB	Switches: Red	From JAN-2020	🧑‍🍳💻
	KEB	Owner:XYZ	From JAN-2020	🏢
	KEB	Switches: Blue	From FEB-2020	

Towards a Solution ⚗️✨

A Sequel to SQL: Nonlinear DBs



	XYZ	Name: @expede	From JAN-2000	➡️⬅️
	ABC	Name: @bmann	From DEC-1999	
	KEB	Type: Wireless	Always	🧑‍🚀📱
🔒	XYZ	Work: Fission	From AUG-2019	
🔒	KEB	Switches: Red	From JAN-2020	🧑‍🍳💻
	KEB	Owner:XYZ	From JAN-2020	🏢
	KEB	Switches: Blue	From FEB-2020	

Towards a Solution ⚗️✨

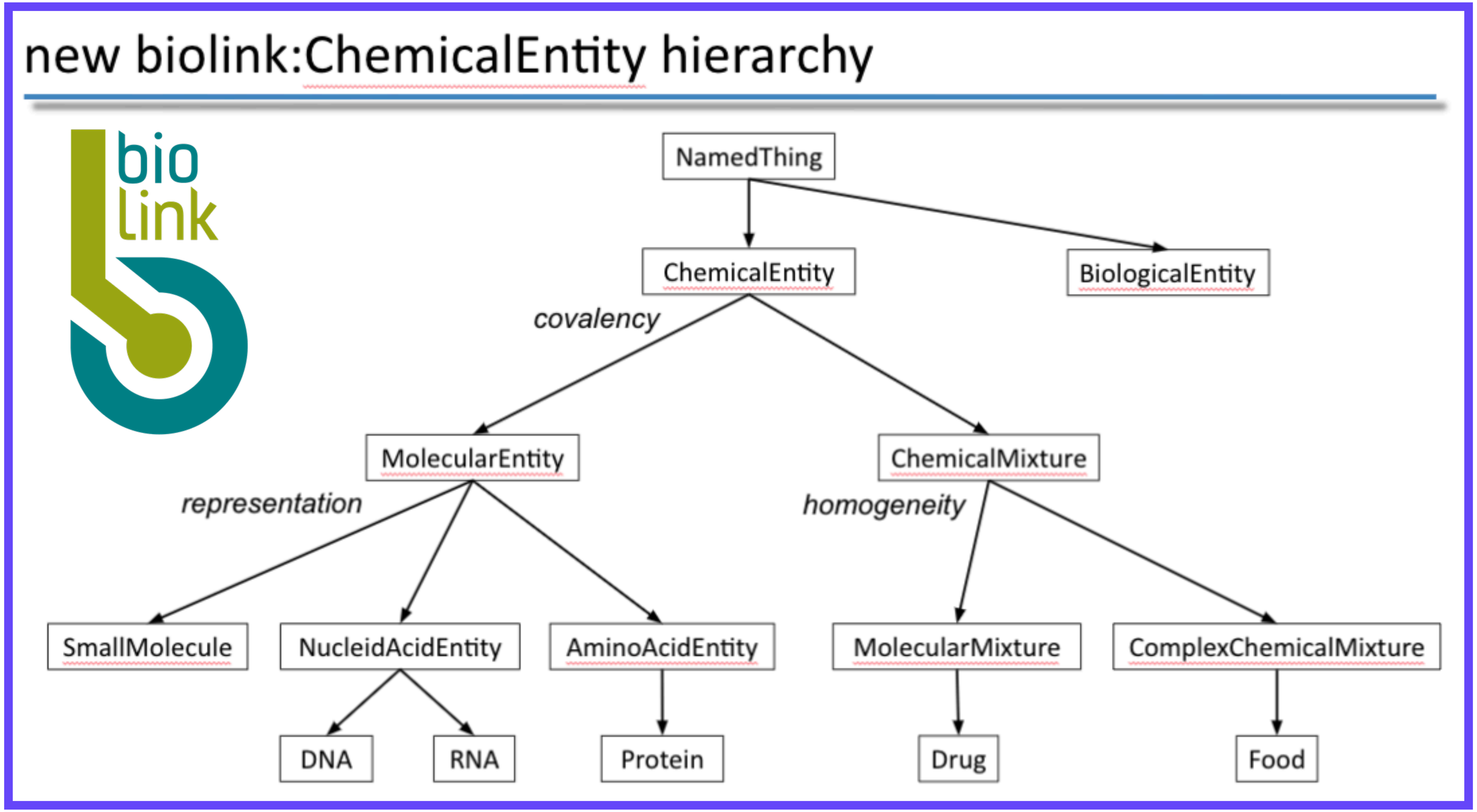
Towards a Solution ⚗️✨

> With over 1.5 million publications per year and more than 50 million total peer-reviewed articles, the ***rate and volume of novel discoveries has surpassed our ability to fully utilize and understand what is known***

– William E. Byrd et al, mediKanren: a System for Biomedical Reasoning

Towards a Solution ⚗️✨

Standardized Knowledge Graphs



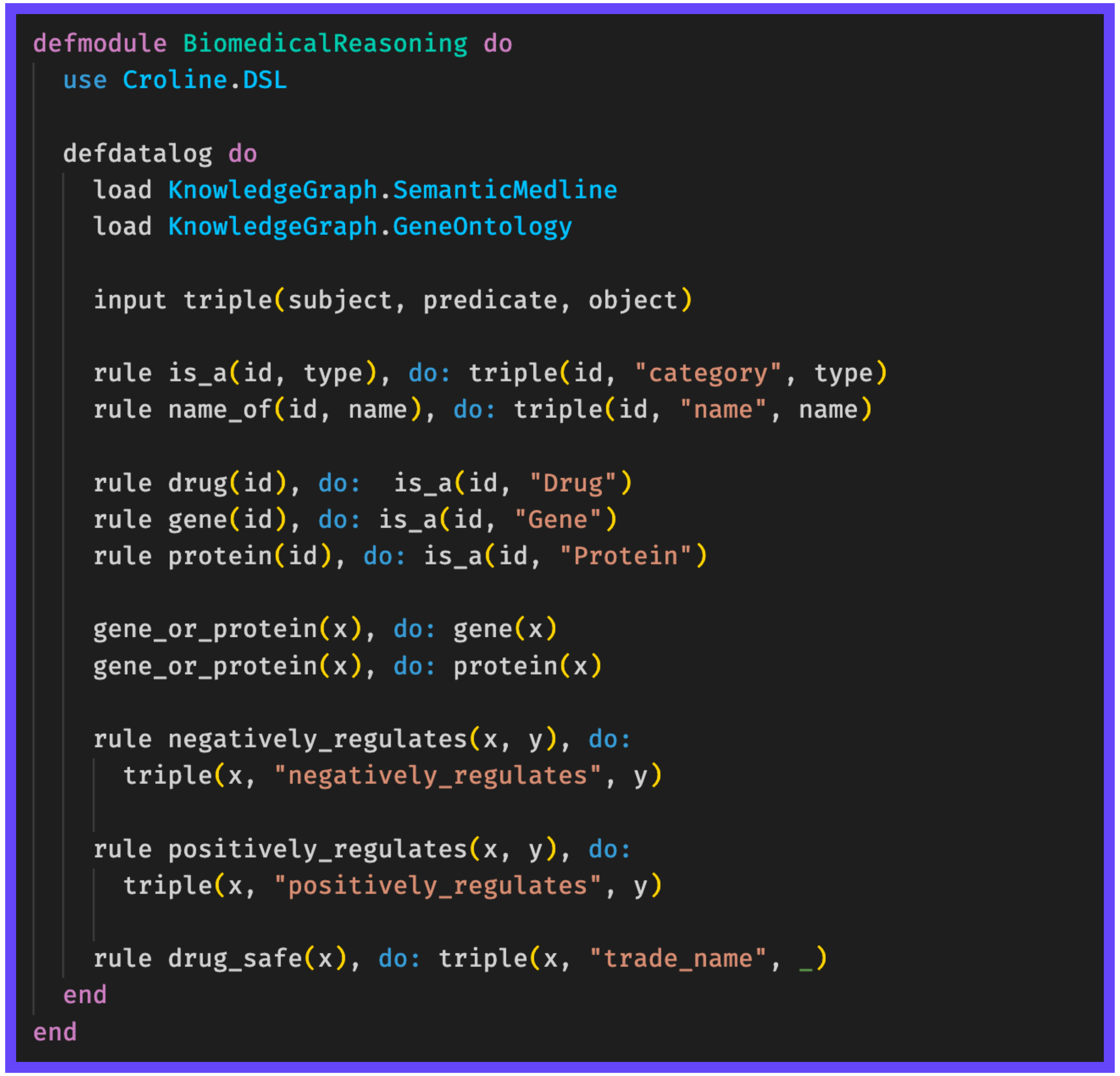
new biolink:ChemicalEntity hierarchy

Towards a Solution ⚗️✨

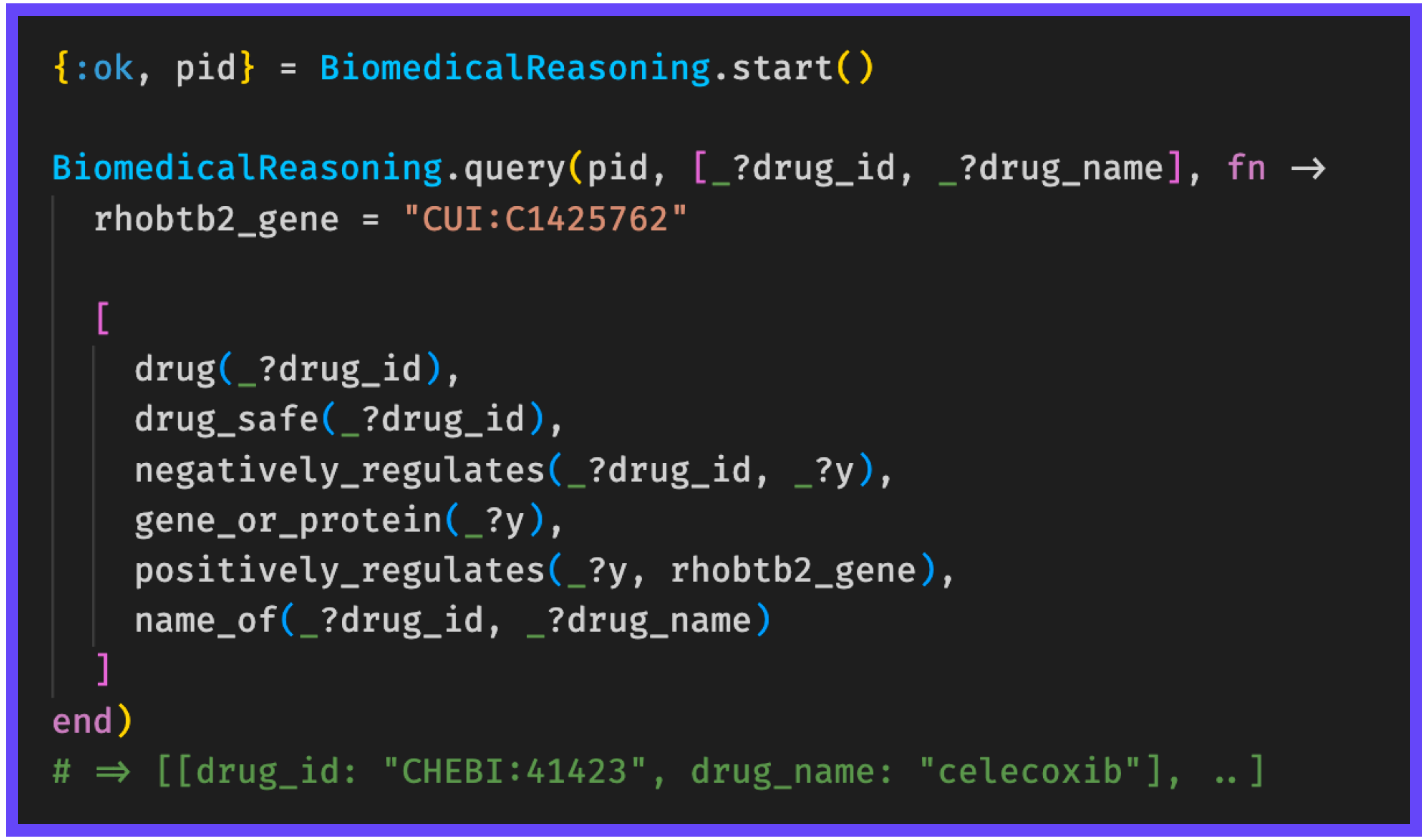
Data Integration

Towards a Solution ⚗️✨

Data Integration



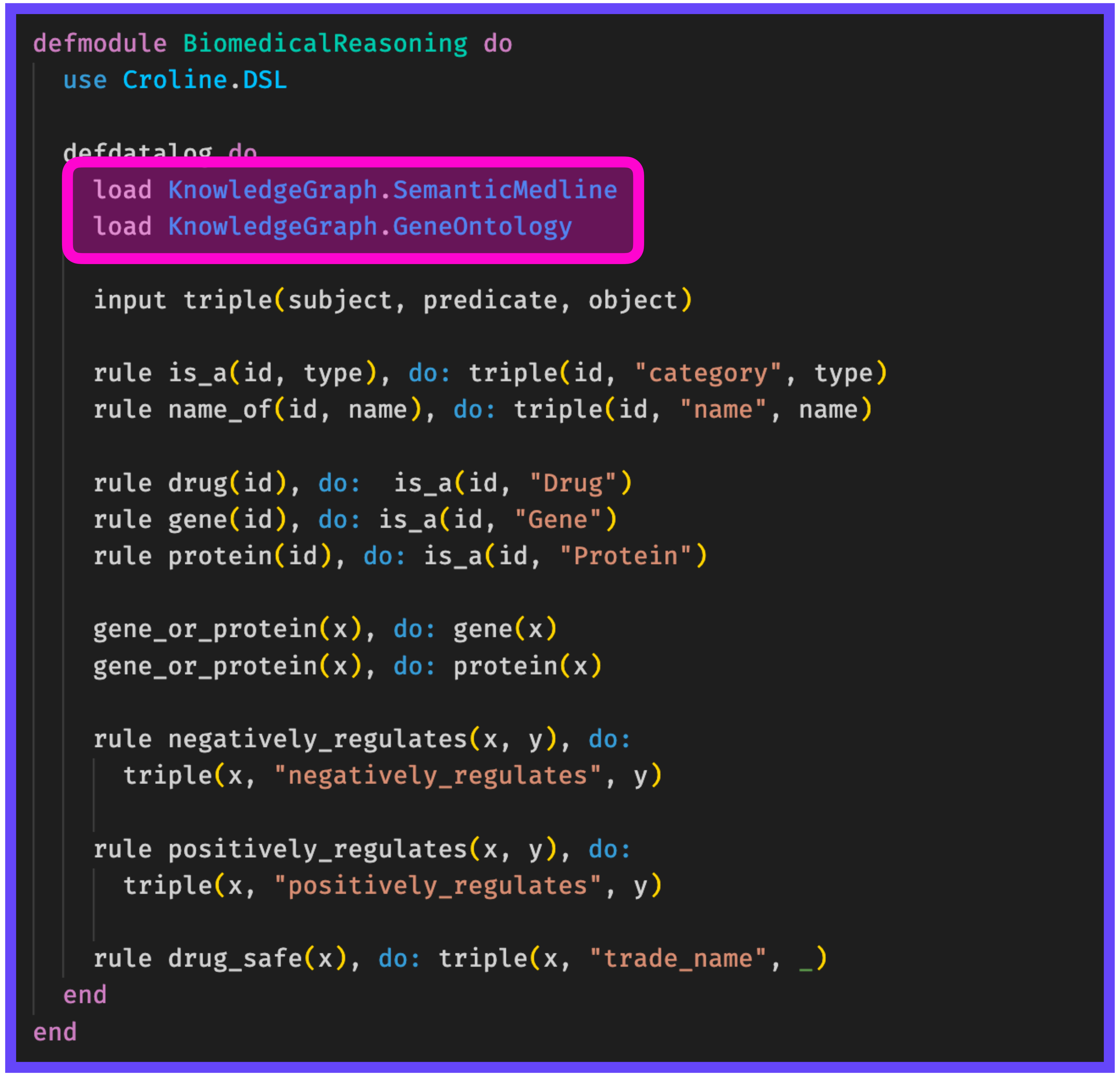
```
defmodule BiomedicalReasoning do
  use Croline.DSL

  defdatalog do
    load KnowledgeGraph.SemanticMedline
    load KnowledgeGraph.GeneOntology

    input triple(subject, predicate, object)

    rule is_a(id, type), do: triple(id, "category", type)
    rule name_of(id, name), do: triple(id, "name", name)

    rule drug(id), do:  is_a(id, "Drug")
    rule gene(id), do: is_a(id, "Gene")
    rule protein(id), do: is_a(id, "Protein")

    gene_or_protein(x), do: gene(x)
    gene_or_protein(x), do: protein(x)

    rule negatively_regulates(x, y), do:
      triple(x, "negatively_regulates", y)

    rule positively_regulates(x, y), do:
      triple(x, "positively_regulates", y)

    rule drug_safe(x), do: triple(x, "trade_name", _)
  end
end
```



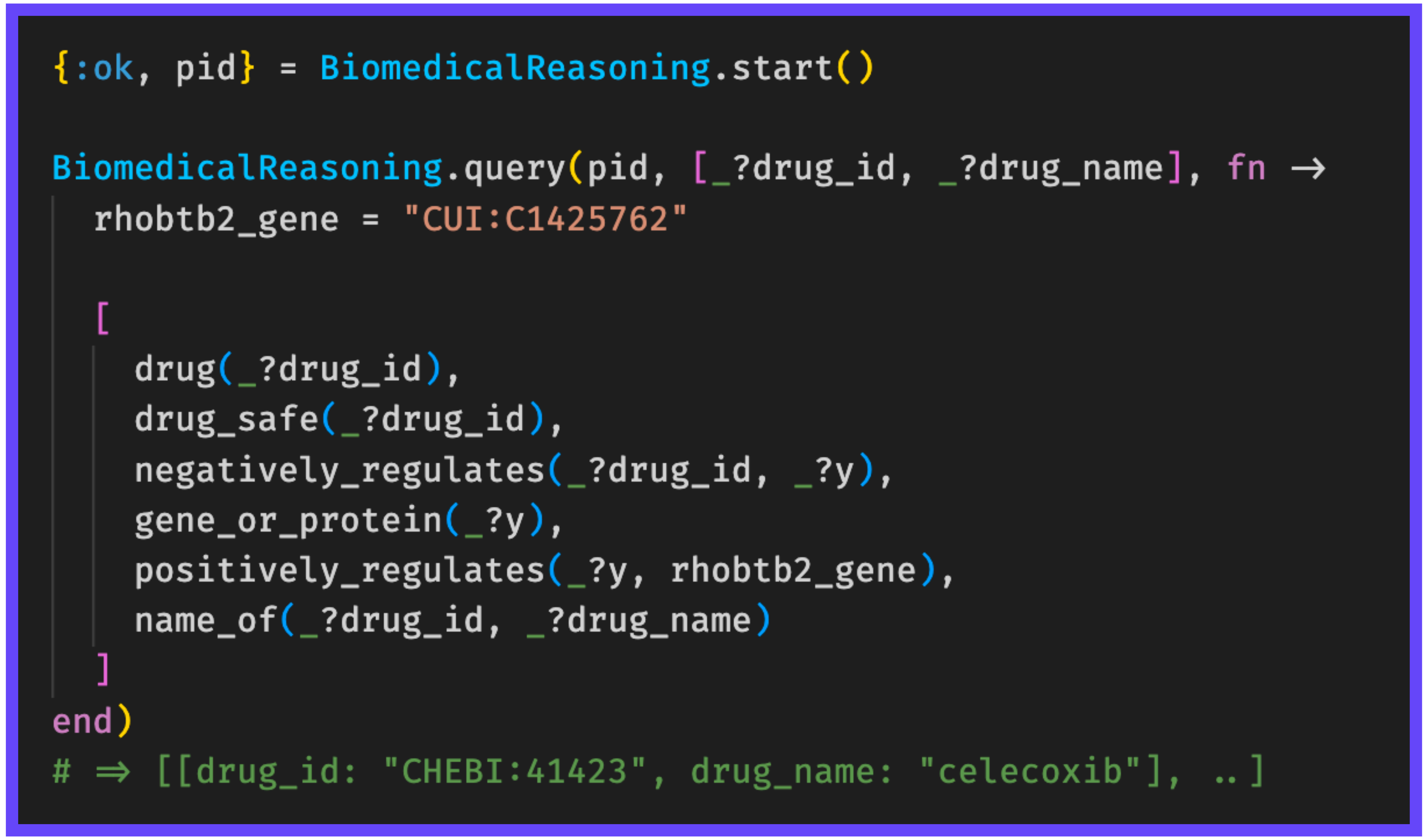
```
{:ok, pid} = BiomedicalReasoning.start()

BiomedicalReasoning.query(pid, [_?drug_id, _?drug_name], fn →
  rhobtb2_gene = "CUI:C1425762"

  [
    drug(_?drug_id),
    drug_safe(_?drug_id),
    negatively_regulates(_?drug_id, _?y),
    gene_or_protein(_?y),
    positively_regulates(_?y, rhobtb2_gene),
    name_of(_?drug_id, _?drug_name)
  ]
end)
# ⇒ [[drug_id: "CHEBI:41423", drug_name: "celecoxib"], ..]
```

Towards a Solution ⚗️✨

Data Integration



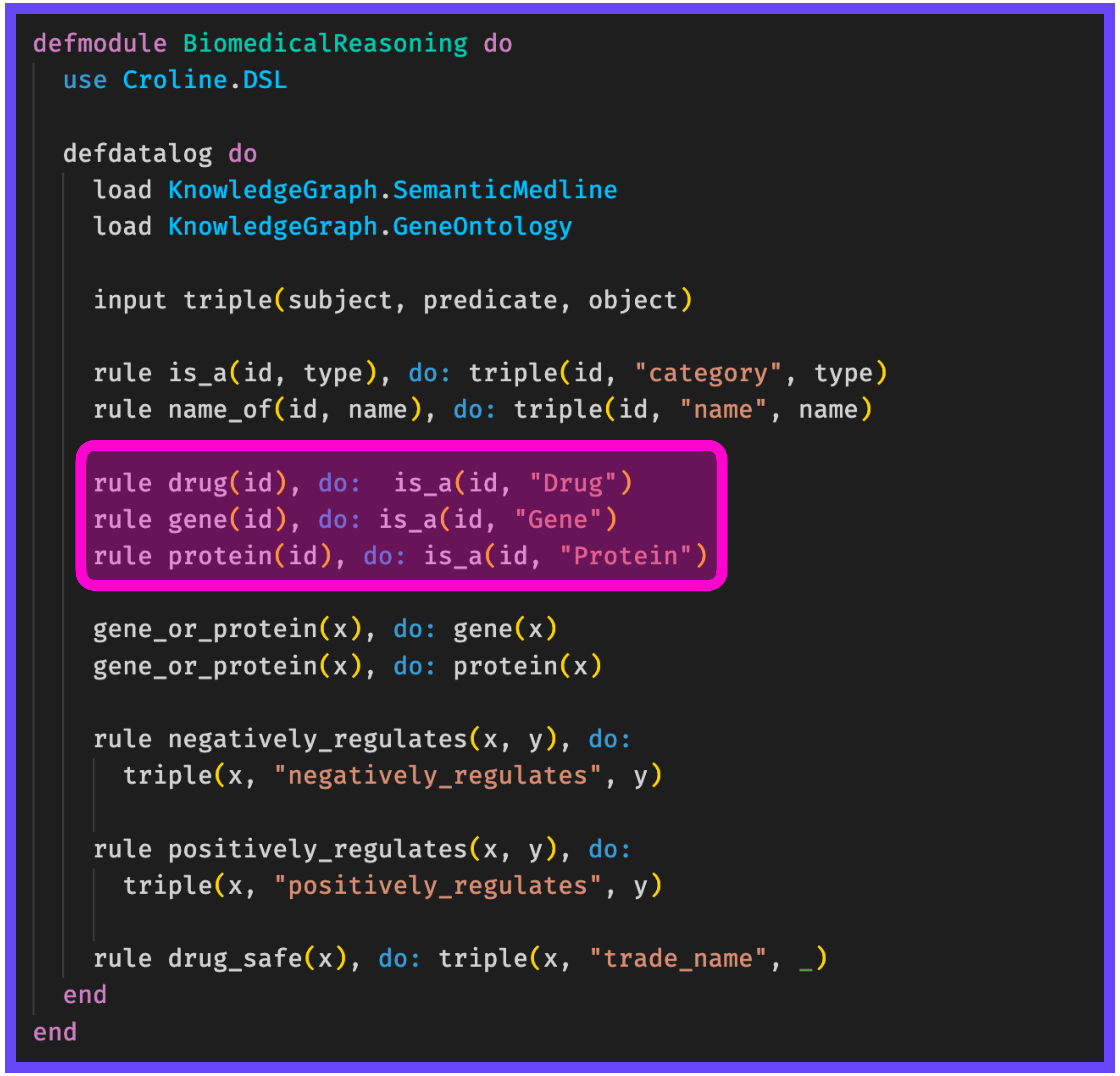
```elixir
defmodule BiomedicalReasoning do
  use Croline.DSL

  defdatalog do
    load KnowledgeGraph.SemanticMedline
    load KnowledgeGraph.GeneOntology

    input triple(subject, predicate, object)

    rule is_a(id, type), do: triple(id, "category", type)
    rule name_of(id, name), do: triple(id, "name", name)

    rule drug(id), do:  is_a(id, "Drug")
    rule gene(id), do: is_a(id, "Gene")
    rule protein(id), do: is_a(id, "Protein")

    gene_or_protein(x), do: gene(x)
    gene_or_protein(x), do: protein(x)

    rule negatively_regulates(x, y), do:
      triple(x, "negatively_regulates", y)

    rule positively_regulates(x, y), do:
      triple(x, "positively_regulates", y)

    rule drug_safe(x), do: triple(x, "trade_name", _)
  end
end
```



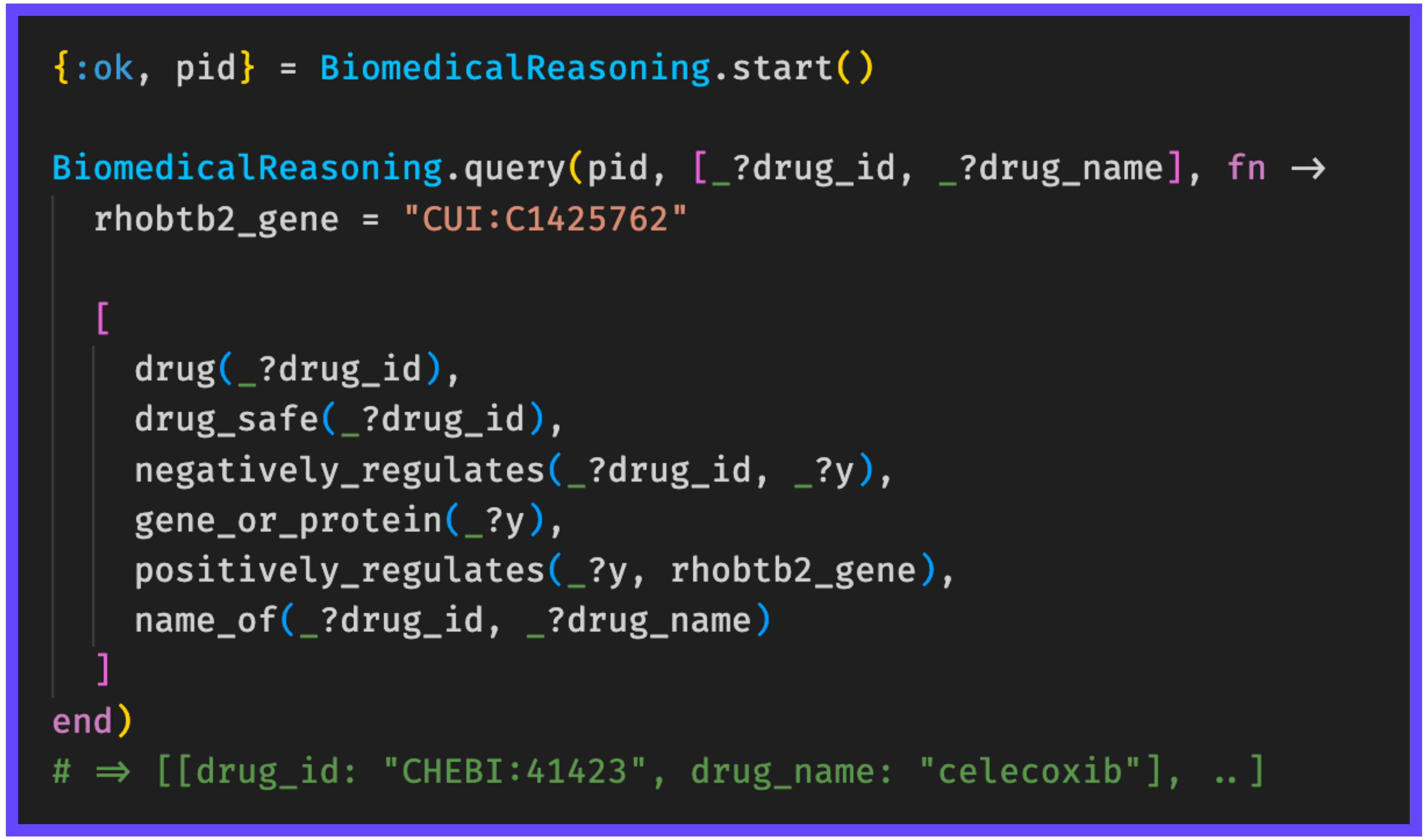
```elixir
{:ok, pid} = BiomedicalReasoning.start()

BiomedicalReasoning.query(pid, [_?drug_id, _?drug_name], fn ->
  rhobtb2_gene = "CUI:C1425762"

  [
    drug(_?drug_id),
    drug_safe(_?drug_id),
    negatively_regulates(_?drug_id, _?y),
    gene_or_protein(_?y),
    positively_regulates(_?y, rhobtb2_gene),
    name_of(_?drug_id, _?drug_name)
  ]
end)
# => [[drug_id: "CHEBI:41423", drug_name: "celecoxib"], ..]
```

Towards a Solution ⚗️✨

Data Integration



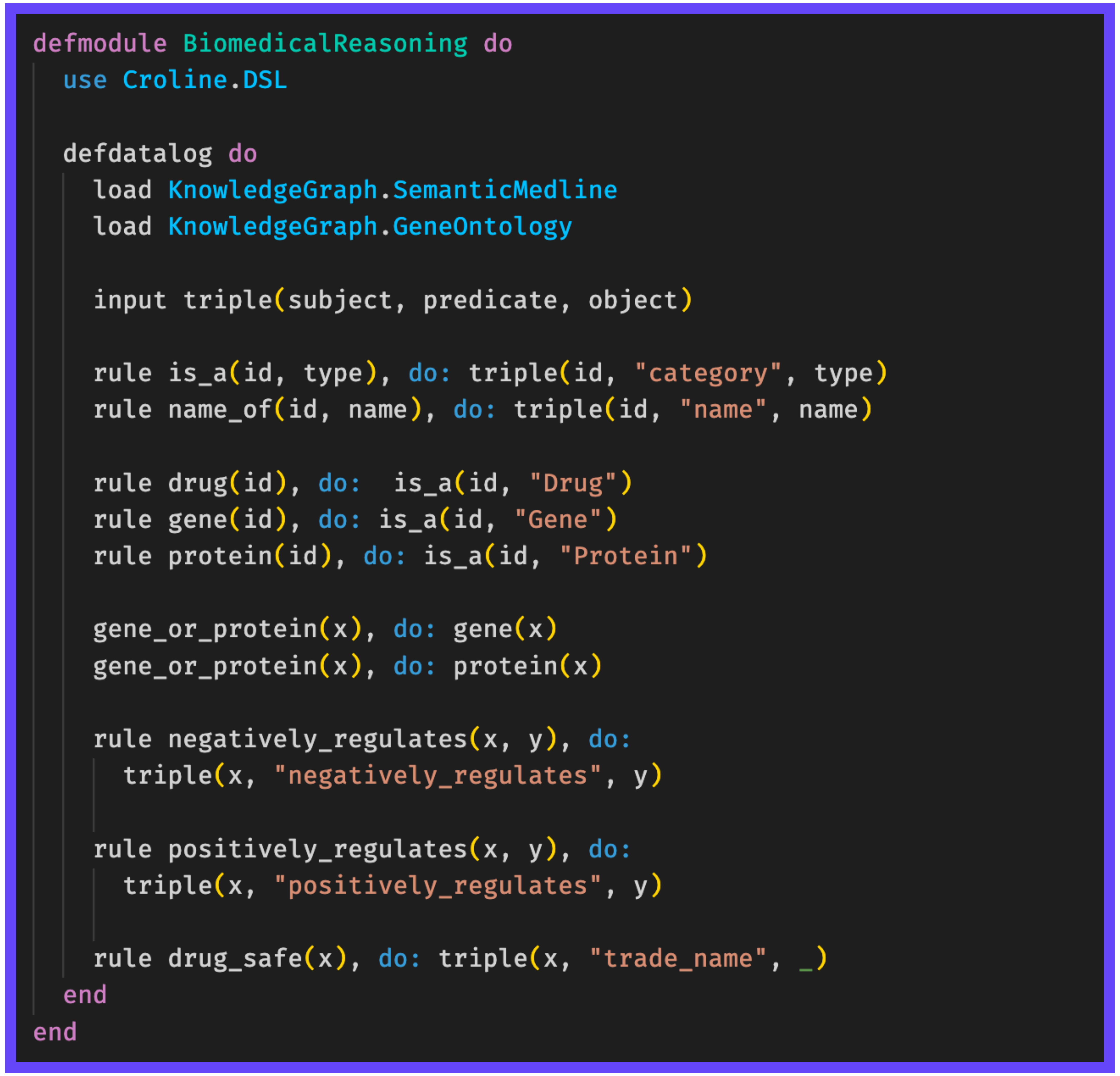
```elixir
defmodule BiomedicalReasoning do
  use Croline.DSL

  defdatalog do
    load KnowledgeGraph.SemanticMedline
    load KnowledgeGraph.GeneOntology

    input triple(subject, predicate, object)

    rule is_a(id, type), do: triple(id, "category", type)
    rule name_of(id, name), do: triple(id, "name", name)

    rule drug(id), do:  is_a(id, "Drug")
    rule gene(id), do: is_a(id, "Gene")
    rule protein(id), do: is_a(id, "Protein")

    gene_or_protein(x), do: gene(x)
    gene_or_protein(x), do: protein(x)

    rule negatively_regulates(x, y), do:
      triple(x, "negatively_regulates", y)

    rule positively_regulates(x, y), do:
      triple(x, "positively_regulates", y)

    rule drug_safe(x), do: triple(x, "trade_name", _)
  end
end
```



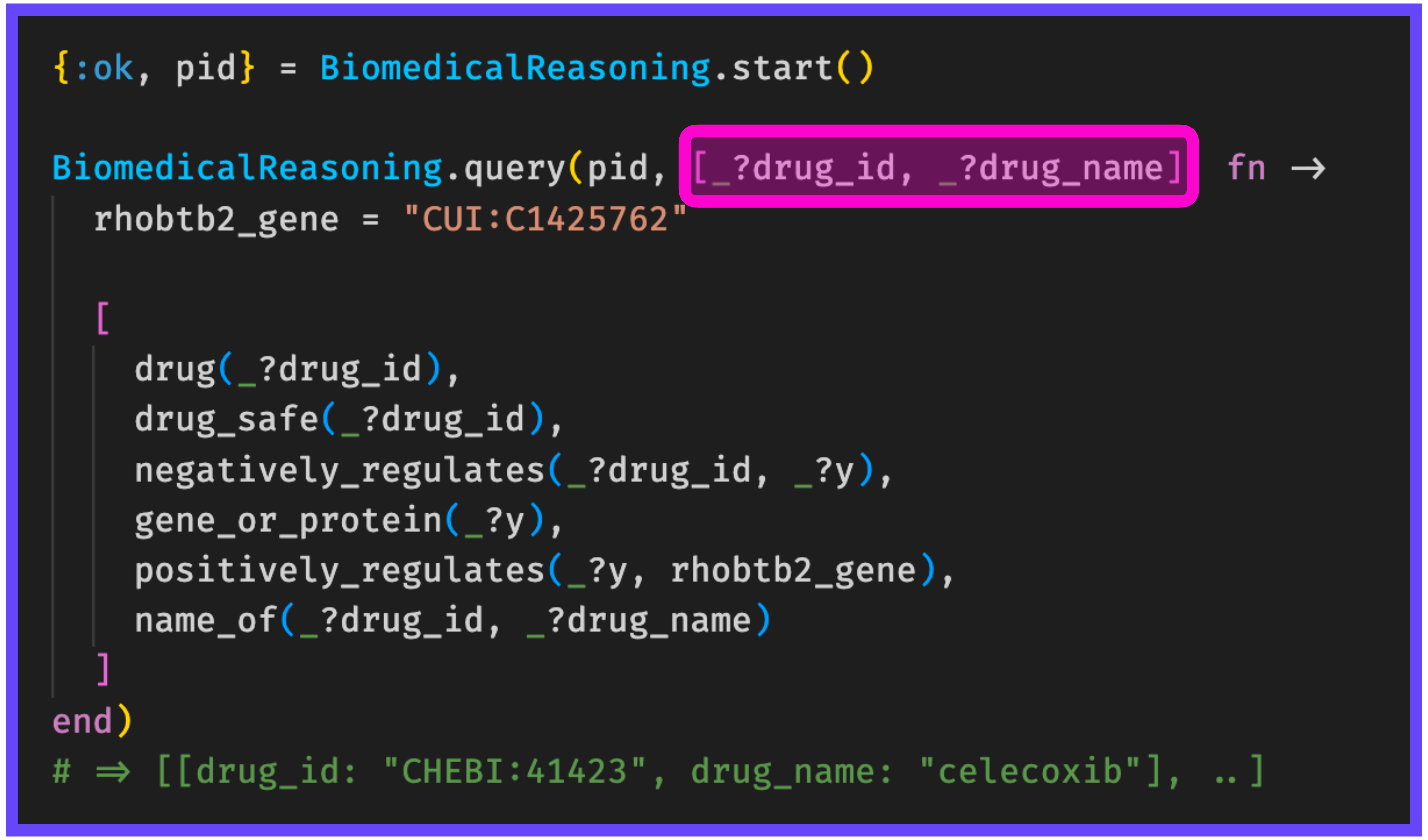
```elixir
{:ok, pid} = BiomedicalReasoning.start()

BiomedicalReasoning.query(pid, [_?drug_id, _?drug_name], fn ->
  rhobtb2_gene = "CUI:C1425762"

  [
    drug(_?drug_id),
    drug_safe(_?drug_id),
    negatively_regulates(_?drug_id, _?y),
    gene_or_protein(_?y),
    positively_regulates(_?y, rhobtb2_gene),
    name_of(_?drug_id, _?drug_name)
  ]
end)
# => [[drug_id: "CHEBI:41423", drug_name: "celecoxib"], ..]
```

Towards a Solution 🔮✨

Data Integration



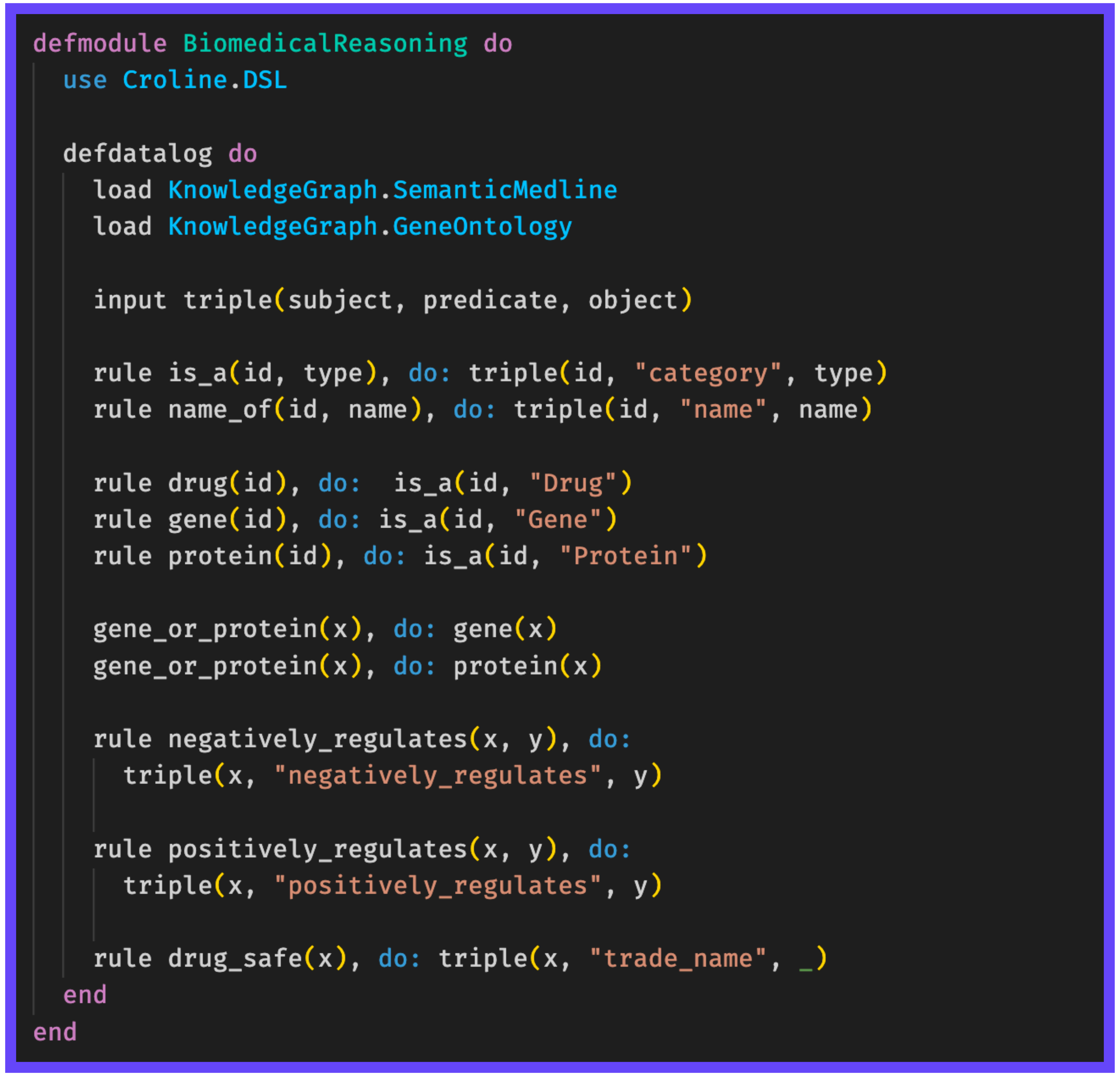
```
defmodule BiomedicalReasoning do
  use Croline.DSL

  defdatalog do
    load KnowledgeGraph.SemanticMedline
    load KnowledgeGraph.GeneOntology

    input triple(subject, predicate, object)

    rule is_a(id, type), do: triple(id, "category", type)
    rule name_of(id, name), do: triple(id, "name", name)

    rule drug(id), do:  is_a(id, "Drug")
    rule gene(id), do: is_a(id, "Gene")
    rule protein(id), do: is_a(id, "Protein")

    gene_or_protein(x), do: gene(x)
    gene_or_protein(x), do: protein(x)

    rule negatively_regulates(x, y), do:
      triple(x, "negatively_regulates", y)

    rule positively_regulates(x, y), do:
      triple(x, "positively_regulates", y)

    rule drug_safe(x), do: triple(x, "trade_name", _)
  end
end
```



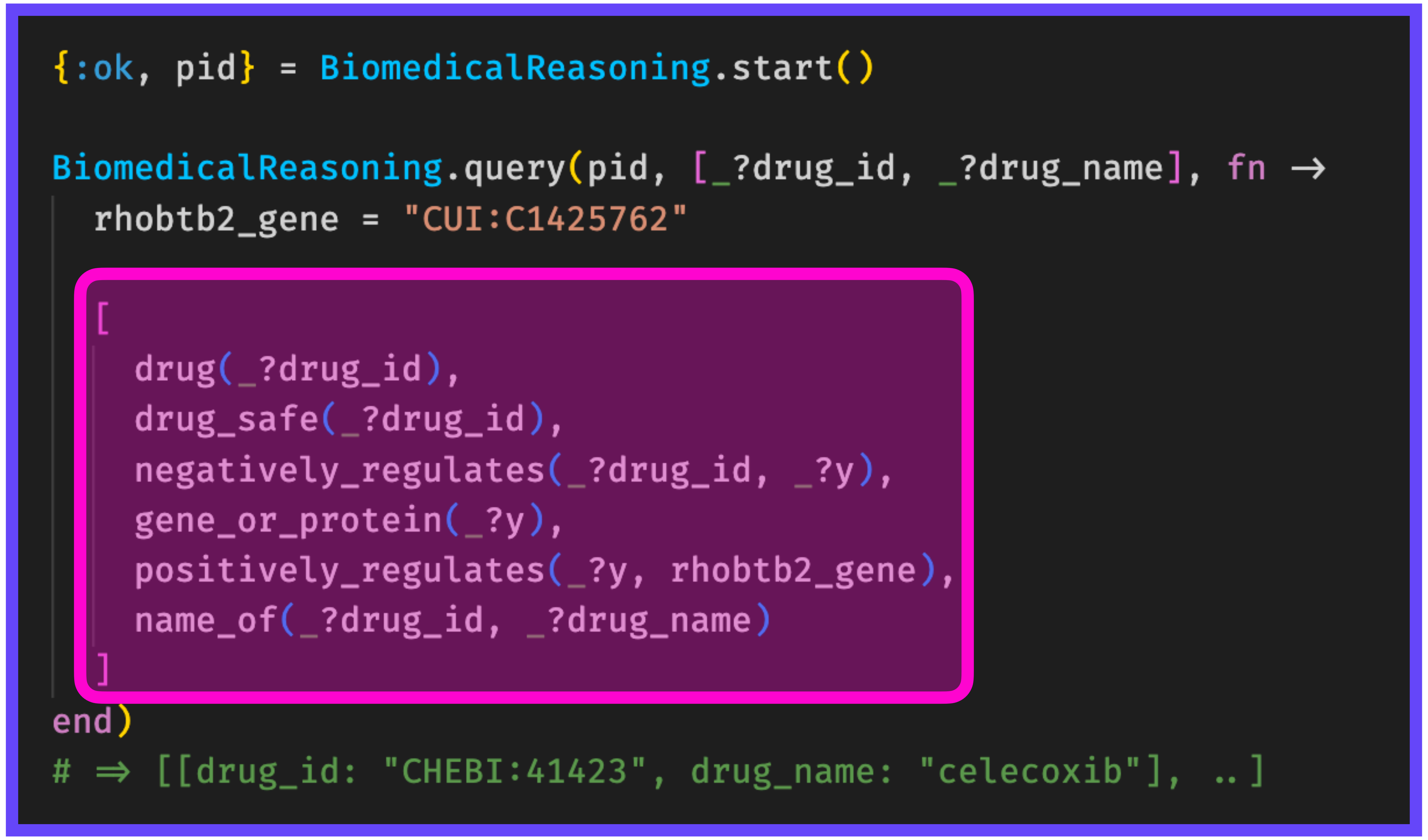
```
{:ok, pid} = BiomedicalReasoning.start()

BiomedicalReasoning.query(pid, [_?drug_id, _?drug_name] fn →
  rhobtb2_gene = "CUI:C1425762"

  [
    drug(_?drug_id),
    drug_safe(_?drug_id),
    negatively_regulates(_?drug_id, _?y),
    gene_or_protein(_?y),
    positively_regulates(_?y, rhobtb2_gene),
    name_of(_?drug_id, _?drug_name)
  ]
end)
# ⇒ [[drug_id: "CHEBI:41423", drug_name: "celecoxib"], ..]
```

Towards a Solution ⚗️✨

Data Integration



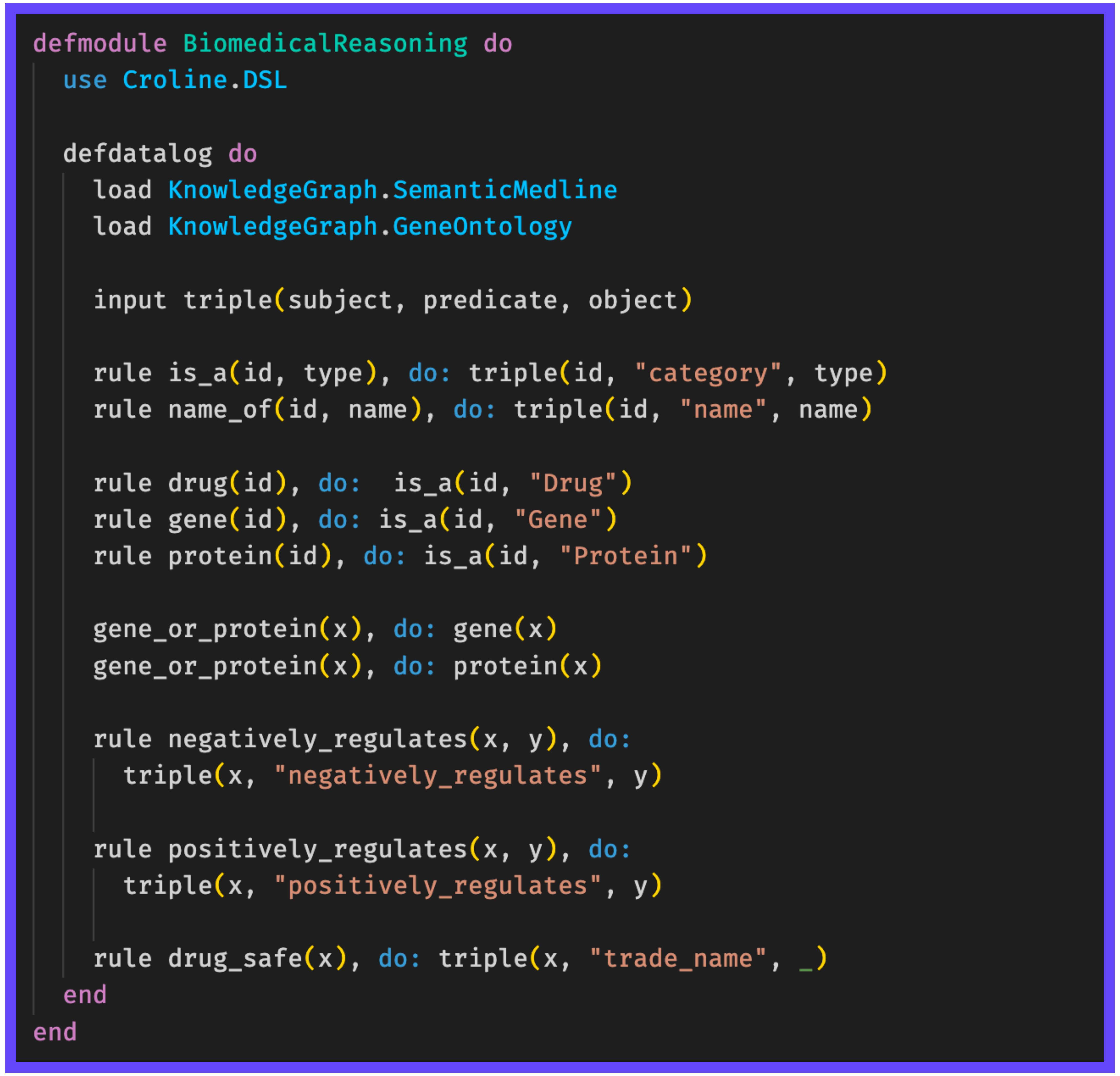
```
defmodule BiomedicalReasoning do
  use Croline.DSL

  defdatalog do
    load KnowledgeGraph.SemanticMedline
    load KnowledgeGraph.GeneOntology

    input triple(subject, predicate, object)

    rule is_a(id, type), do: triple(id, "category", type)
    rule name_of(id, name), do: triple(id, "name", name)

    rule drug(id), do:  is_a(id, "Drug")
    rule gene(id), do: is_a(id, "Gene")
    rule protein(id), do: is_a(id, "Protein")

    gene_or_protein(x), do: gene(x)
    gene_or_protein(x), do: protein(x)

    rule negatively_regulates(x, y), do:
      triple(x, "negatively_regulates", y)

    rule positively_regulates(x, y), do:
      triple(x, "positively_regulates", y)

    rule drug_safe(x), do: triple(x, "trade_name", _)
  end
end
```



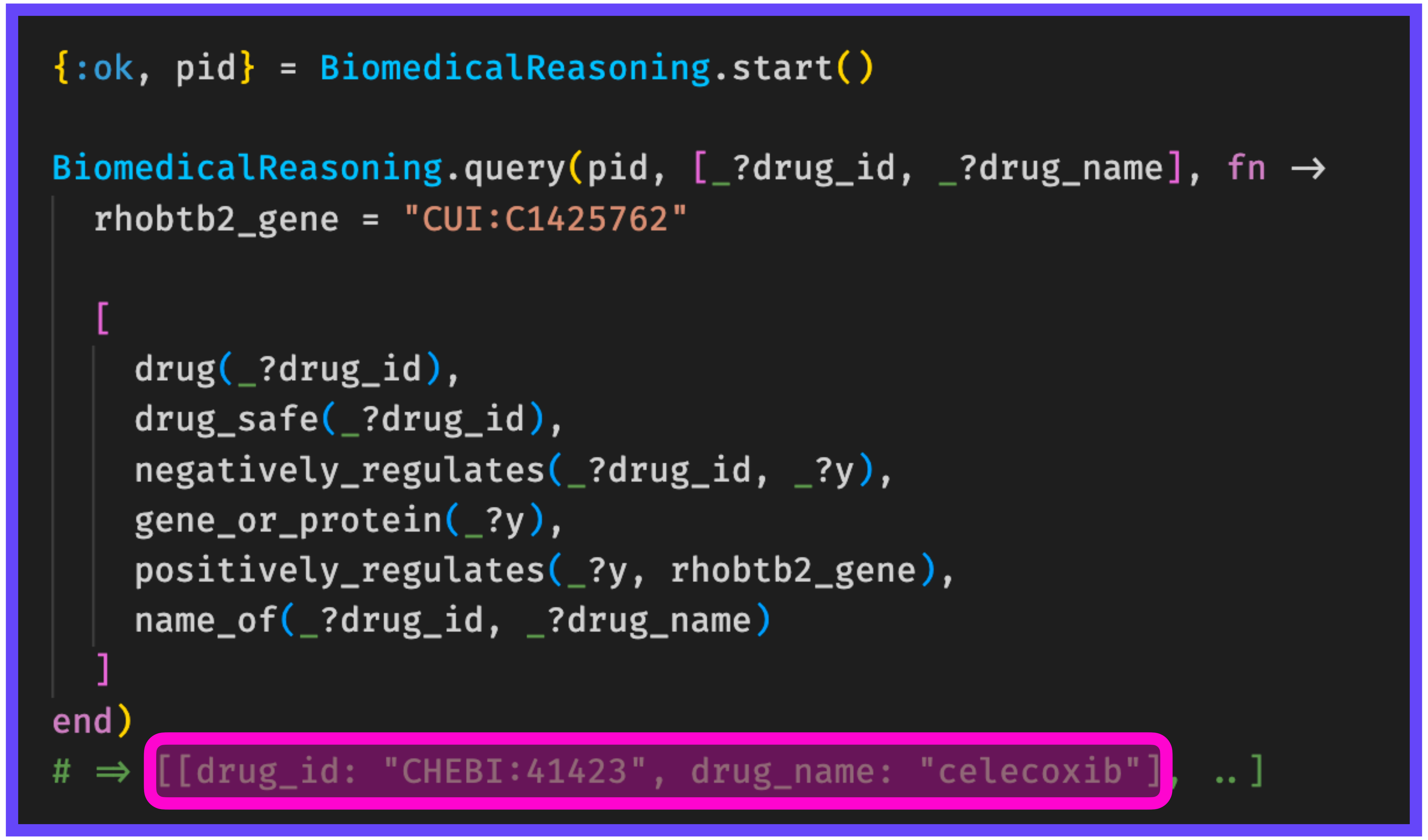
```
{:ok, pid} = BiomedicalReasoning.start()

BiomedicalReasoning.query(pid, [_?drug_id, _?drug_name], fn →
  rhobtb2_gene = "CUI:C1425762"

  [
    drug(_?drug_id),
    drug_safe(_?drug_id),
    negatively_regulates(_?drug_id, _?y),
    gene_or_protein(_?y),
    positively_regulates(_?y, rhobtb2_gene),
    name_of(_?drug_id, _?drug_name)
  ]
end)
# ⇒ [[drug_id: "CHEBI:41423", drug_name: "celecoxib"], ..]
```

Towards a Solution ⚗️✨

Data Integration



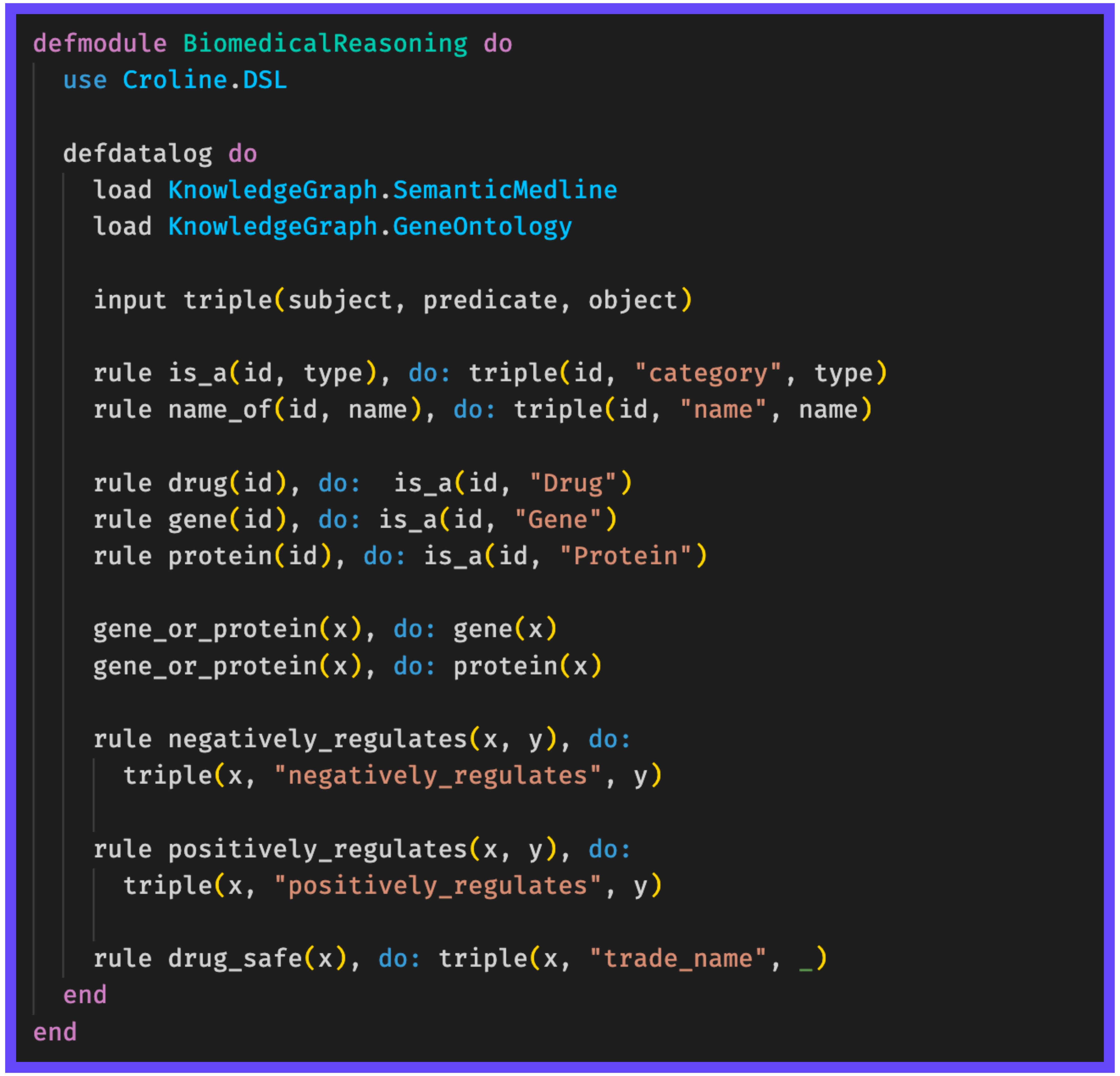
```
defmodule BiomedicalReasoning do
  use Croline.DSL

  defdatalog do
    load KnowledgeGraph.SemanticMedline
    load KnowledgeGraph.GeneOntology

    input triple(subject, predicate, object)

    rule is_a(id, type), do: triple(id, "category", type)
    rule name_of(id, name), do: triple(id, "name", name)

    rule drug(id), do:  is_a(id, "Drug")
    rule gene(id), do: is_a(id, "Gene")
    rule protein(id), do: is_a(id, "Protein")

    gene_or_protein(x), do: gene(x)
    gene_or_protein(x), do: protein(x)

    rule negatively_regulates(x, y), do:
      triple(x, "negatively_regulates", y)

    rule positively_regulates(x, y), do:
      triple(x, "positively_regulates", y)

    rule drug_safe(x), do: triple(x, "trade_name", _)
  end
end
```



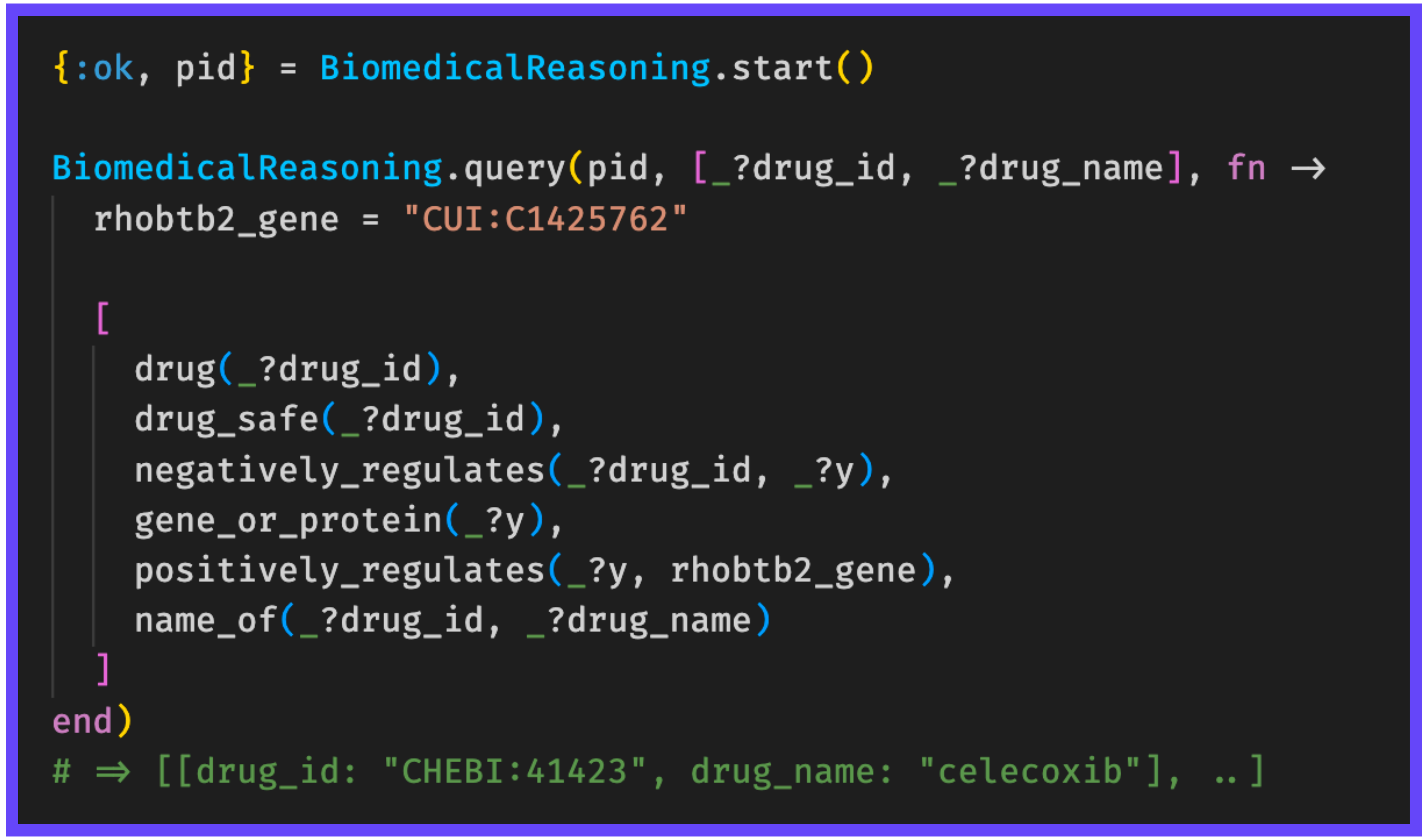
```
{:ok, pid} = BiomedicalReasoning.start()

BiomedicalReasoning.query(pid, [_?drug_id, _?drug_name], fn ->
  rhobtb2_gene = "CUI:C1425762"

  [
    drug(_?drug_id),
    drug_safe(_?drug_id),
    negatively_regulates(_?drug_id, _?y),
    gene_or_protein(_?y),
    positively_regulates(_?y, rhobtb2_gene),
    name_of(_?drug_id, _?drug_name)
  ]
end)
# => [[drug_id: "CHEBI:41423", drug_name: "celecoxib"], ..]
```

Towards a Solution 🔮✨

Data Integration



```
defmodule BiomedicalReasoning do
  use Croline.DSL

  defdatalog do
    load KnowledgeGraph.SemanticMedline
    load KnowledgeGraph.GeneOntology

    input triple(subject, predicate, object)

    rule is_a(id, type), do: triple(id, "category", type)
    rule name_of(id, name), do: triple(id, "name", name)

    rule drug(id), do:  is_a(id, "Drug")
    rule gene(id), do: is_a(id, "Gene")
    rule protein(id), do: is_a(id, "Protein")

    gene_or_protein(x), do: gene(x)
    gene_or_protein(x), do: protein(x)

    rule negatively_regulates(x, y), do:
      triple(x, "negatively_regulates", y)

    rule positively_regulates(x, y), do:
      triple(x, "positively_regulates", y)

    rule drug_safe(x), do: triple(x, "trade_name", _)
  end
end
```



```
{:ok, pid} = BiomedicalReasoning.start()

BiomedicalReasoning.query(pid, [_?drug_id, _?drug_name], fn →
  rhobtb2_gene = "CUI:C1425762"

  [
    drug(_?drug_id),
    drug_safe(_?drug_id),
    negatively_regulates(_?drug_id, _?y),
    gene_or_protein(_?y),
    positively_regulates(_?y, rhobtb2_gene),
    name_of(_?drug_id, _?drug_name)
  ]
end)
# ⇒ [[drug_id: "CHEBI:41423", drug_name: "celecoxib"], ..]
```

A 'high-speed Dr. House' for medical breakthroughs

– University of Alabama News, on mediKanren

Your Turn

Call to Action

🦸

Call to Action 🦸

Call to Action 🦸

We have a system that applies **cutting edge** CS research to **tackle day-to-day problems** in the applications we all write.

Phoenix Presence
- has **no single point of failure**
- has **no single source of truth**
-[...]
- **self heals**

~ Chris McCord, “What Makes Phoenix Presence Special”

Call to Action 🦸

Next Steps

Call to Action 🦸

Next Steps

1. **Embrace** the distributed nature of the network 🫂

Call to Action 🦸

Next Steps

1. **Embrace** the distributed nature of the network 🫂
2. Put data into **interoperable** forms

Call to Action 🦸

Next Steps

1. **Embrace** the distributed nature of the network 🫂
2. Put data into **interoperable** forms
3. Better living through **replication**

🎉 Thank You, A Coruña 🇪🇸

https://fission.codes
{brooklyn,quinn}@fission.codes
github.com/expede | github.com/quinnwilton
@expede | @wilton_quinn