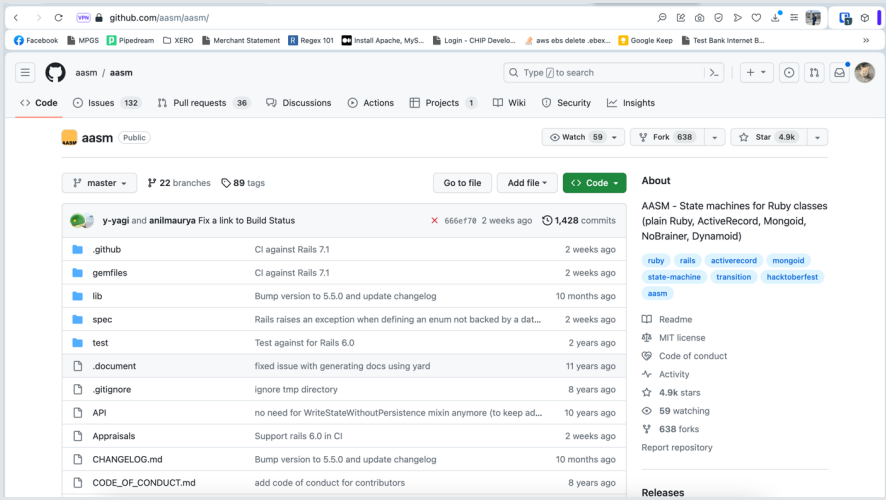
AASM Callback & Sidekiq

By Wan Zulkarnain

https://github.com/wzul

AASM

- Short form for “act_as_state_machine”.
- Useful for manipulating state for a specific record.
- https://github.com/aasm/aasm/



Sidekiq

- Framework to run a task independently from main thread.
- **Ultra-fast** compared to delayed job (introduced by Shopify)
- https://github.com/sidekiq/sidekiq

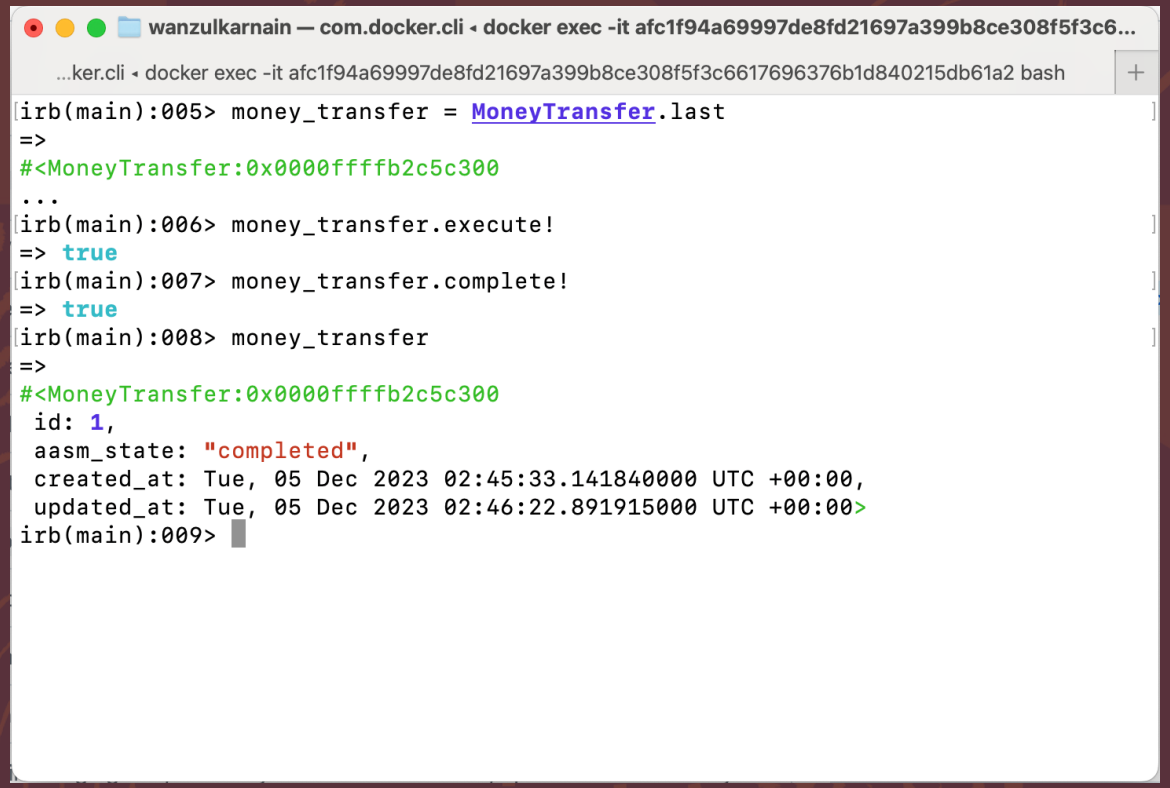
money_transfer.rb U ● | 20231205020936_create_money_transfer.rb U ●

app > models > money_transfer.rb

```ruby
class MoneyTransfer < ApplicationRecord
  include AASM

  aasm do
    state :received, initial: true
    state :executing
    state :completed

    event :execute do
      transitions from: :received, to: :executing
    end

    event :complete do
      transitions from: :executing, to: :completed
    end
  end
end
```

Example: Money Transfer

- 3 states:
  - Received
  - Executing
  - Completed
- 2 events:
  - Execute
  - Complete



AASM facilitate state changes

- irb(main):012> money_transfer.**aasm_state**
- => **"received"**
- irb(main):013> money_transfer.**execute!**
- => **true**
- irb(main):014> money_transfer.aasm_state
- => **"executing"**
- irb(main):015> money_transfer.**complete!**
- => **true**
- irb(main):016> money_transfer.aasm_state
- => **"completed"**

money_transfer.rb U × charge_customer_worker.rb U

app > models > money_transfer.rb

```ruby
class MoneyTransfer < ApplicationRecord
  include AASM

  aasm do
    state :received, initial: true
    state :executing
    state :completed

    event :execute do
      transitions from: :received, to: :executing
    end

    event :complete, after: :charge_customer do
      transitions from: :executing, to: :completed
    end
  end

  def charge_customer
    ChargeCustomerWorker.perform_async(id)
  end
end

```

Let's add callback

- Hook **:complete** event with **:charge_customer**
- `event :complete, after: :charge_customer do`
- Method **charge_customer**
- `ChargeCustomerWorker.perform_async id`

settlement

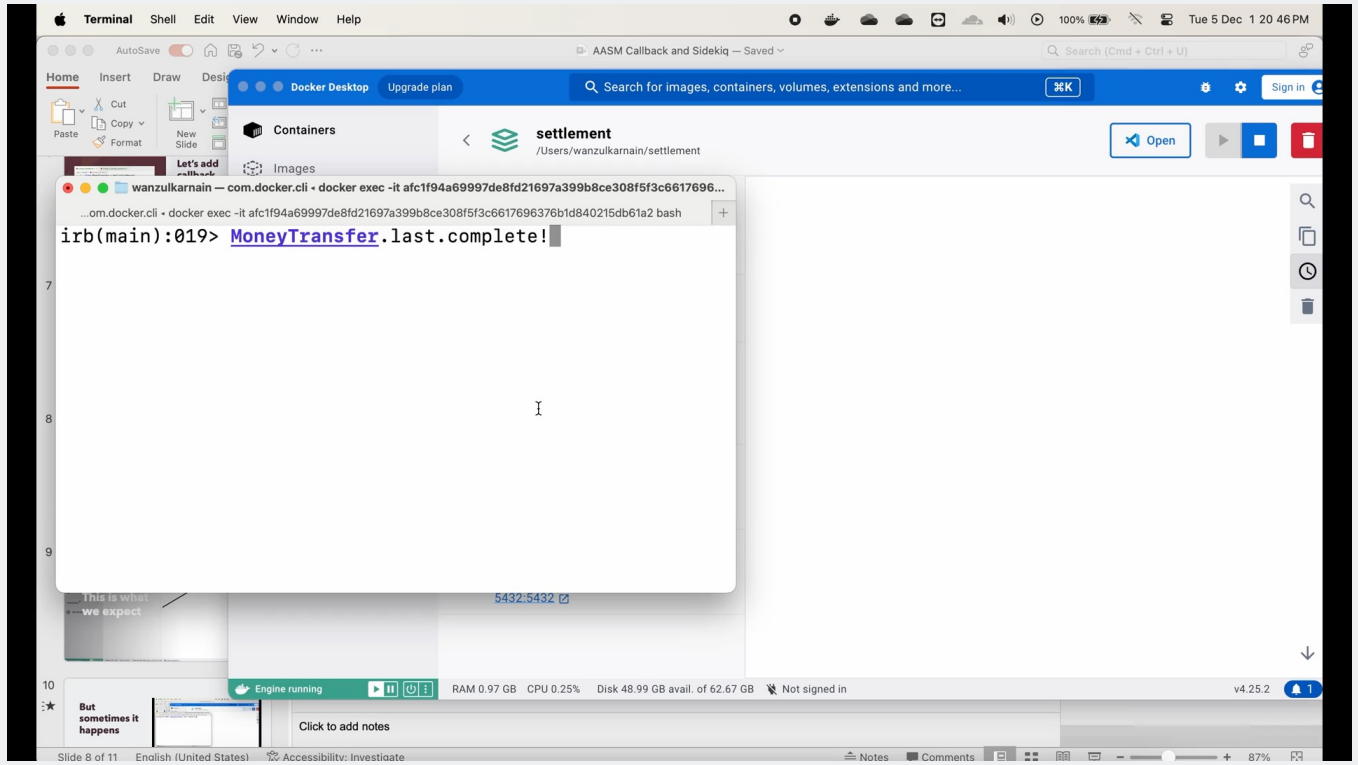
money_transfer.rb U | charge_customer_worker.rb U

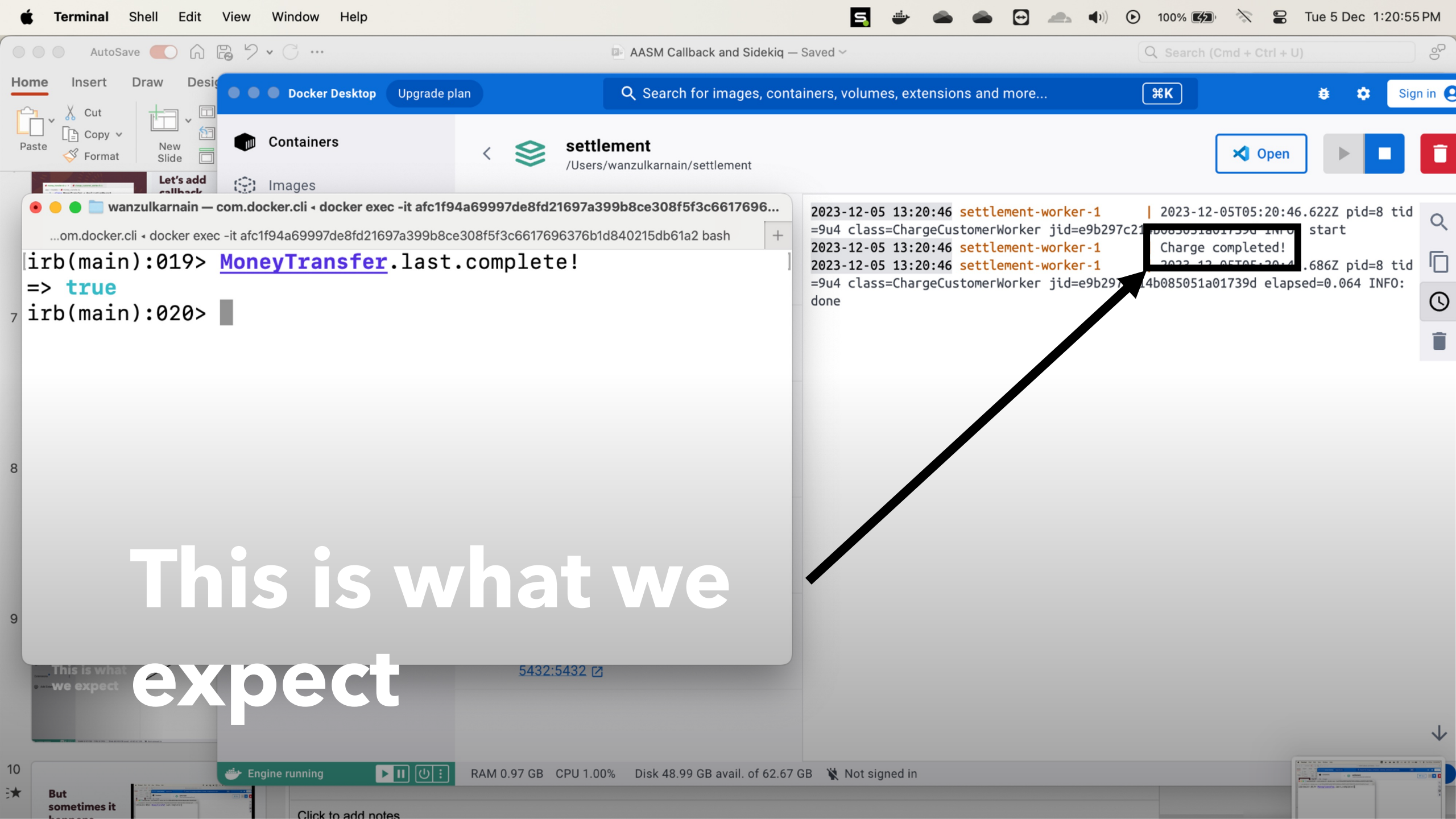
app > sidekiq > charge_customer_worker.rb

```ruby
class ChargeCustomerWorker < SidekiqJob
  def perform(id)
    money_transfer = MoneyTransfer.find(id)

    if money_transfer.aasm_read_state == :completed
      puts 'Charge completed!'
    else
      puts 'Charge not completed!'
    end
  end
end

```

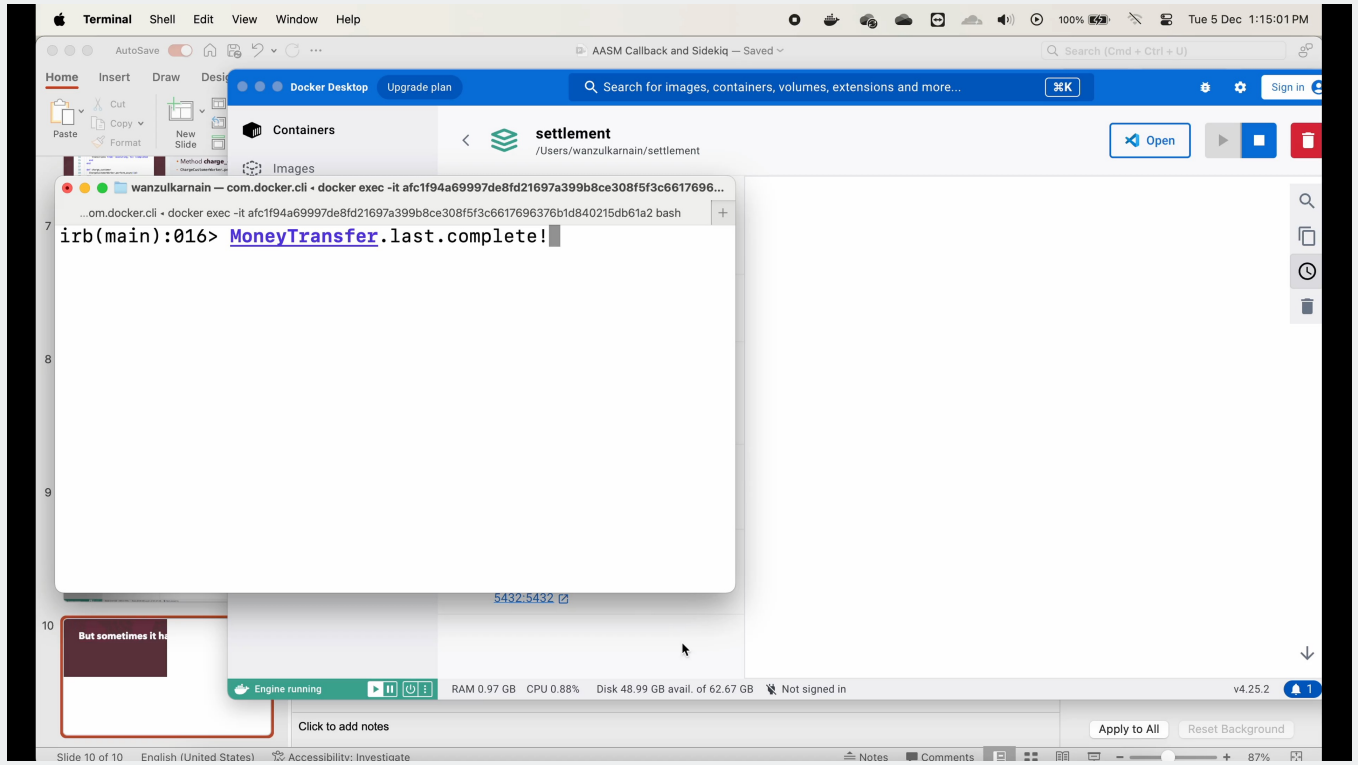
And define the worker

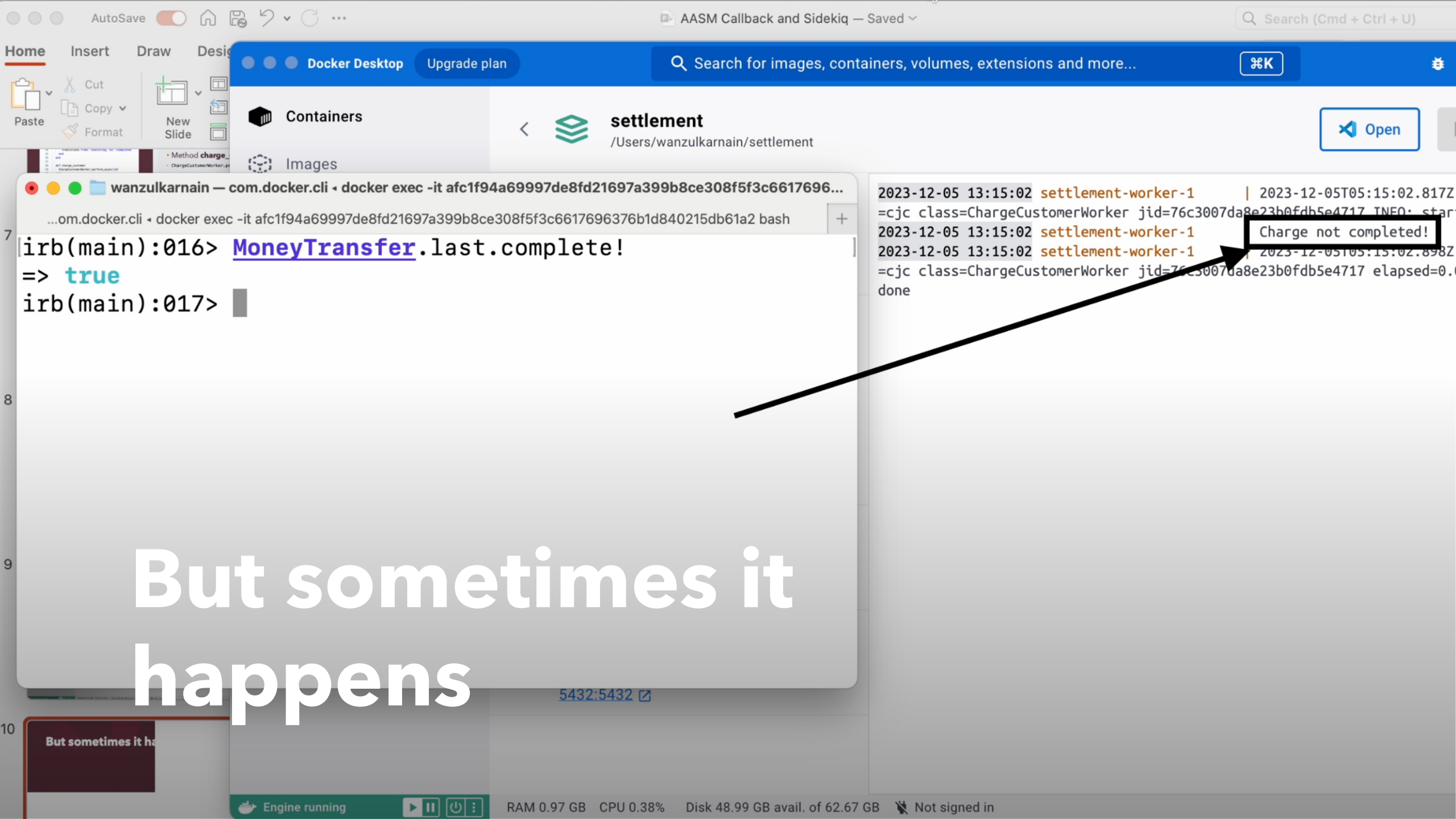
This is what we expect




Terminal Shell Edit View Window Help
Tue 5 Dec 1:20:55 PM
AASM Callback and Sidekiq — Saved
Docker Desktop
Upgrade plan
Search for images, containers, volumes, extensions and more...
Sign in
Containers
Images
settlement
/Users/wanzulkarnain/settlement
Open
wanzulkarnain — com.docker.cli ◂ docker exec -it afc1f94a69997de8fd21697a399b8ce308f5f3c6617696...
...om.docker.cli ◂ docker exec -it afc1f94a69997de8fd21697a399b8ce308f5f3c6617696376b1d840215db61a2 bash
irb(main):019> MoneyTransfer.last.complete!
=> true
irb(main):020>
2023-12-05 13:20:46 settlement-worker-1 | 2023-12-05T05:20:46.622Z pid=8 tid
=9u4 class=ChargeCustomerWorker jid=e9b297c2 start
2023-12-05 13:20:46 settlement-worker-1 Charge completed!
2023-12-05 13:20:46 settlement-worker-1 .686Z pid=8 tid
=9u4 class=ChargeCustomerWorker jid=e9b297 4b085051a01739d elapsed=0.064 INFO:
done
This is what we expect
5432:5432
Engine running
RAM 0.97 GB CPU 1.00% Disk 48.99 GB avail. of 62.67 GB Not signed in
But sometimes it
Click to add notes

But sometimes it happens




AASM Callback and Sidekiq — Saved
Search (Cmd + Ctrl + U)
Home
Insert
Draw
Paste
Cut
Copy
Format
New Slide
Docker Desktop
Upgrade plan
Search for images, containers, volumes, extensions and more...
⌘K
Containers
Images
settlement
/Users/wanzulkarnain/settlement
Open
wanzulkarnain — com.docker.cli ◂ docker exec -it afc1f94a69997de8fd21697a399b8ce308f5f3c6617696...
...om.docker.cli ◂ docker exec -it afc1f94a69997de8fd21697a399b8ce308f5f3c6617696376b1d840215db61a2 bash
irb(main):016> MoneyTransfer.last.complete!
=> true
irb(main):017>
2023-12-05 13:15:02 settlement-worker-1 | 2023-12-05T05:15:02.817Z
=cjc class=ChargeCustomerWorker jid=76c3007da8e23b0fdb5e4717 INFO: star
2023-12-05 13:15:02 settlement-worker-1 | Charge not completed!
2023-12-05 13:15:02 settlement-worker-1 | 2023-12-05T05:15:02.898Z
=cjc class=ChargeCustomerWorker jid=76c3007da8e23b0fdb5e4717 elapsed=0.
done
But sometimes it happens
5432:5432
But sometimes it ha
Engine running
RAM 0.97 GB
CPU 0.38%
Disk 48.99 GB avail. of 62.67 GB
Not signed in

Cause?

AASM after callback is executed before database commit.

Sidekiq execute the job earlier than database commit.

Usually, it doesn’t happen in local development, but happened in production.

How to replicate?

- Add before_commit block and delay for X seconds.

```
before_commit :delay

def delay
  sleep(2)
end
```

money_transfer.rb U ● | charge_customer_worker.rb U

app > models > money_transfer.rb

```ruby
class MoneyTransfer < ApplicationRecord
  include AASM

  before_commit :delay

  def delay
    sleep(2)
  end

  aasm do
    state :received, initial: true
    state :executing
    state :completed

    event :execute do
      transitions from: :received, to: :executing
    end

    event :complete, after: :charge_customer do
      transitions from: :executing, to: :completed
    end
  end
```

How to replicate?

How to prevent?

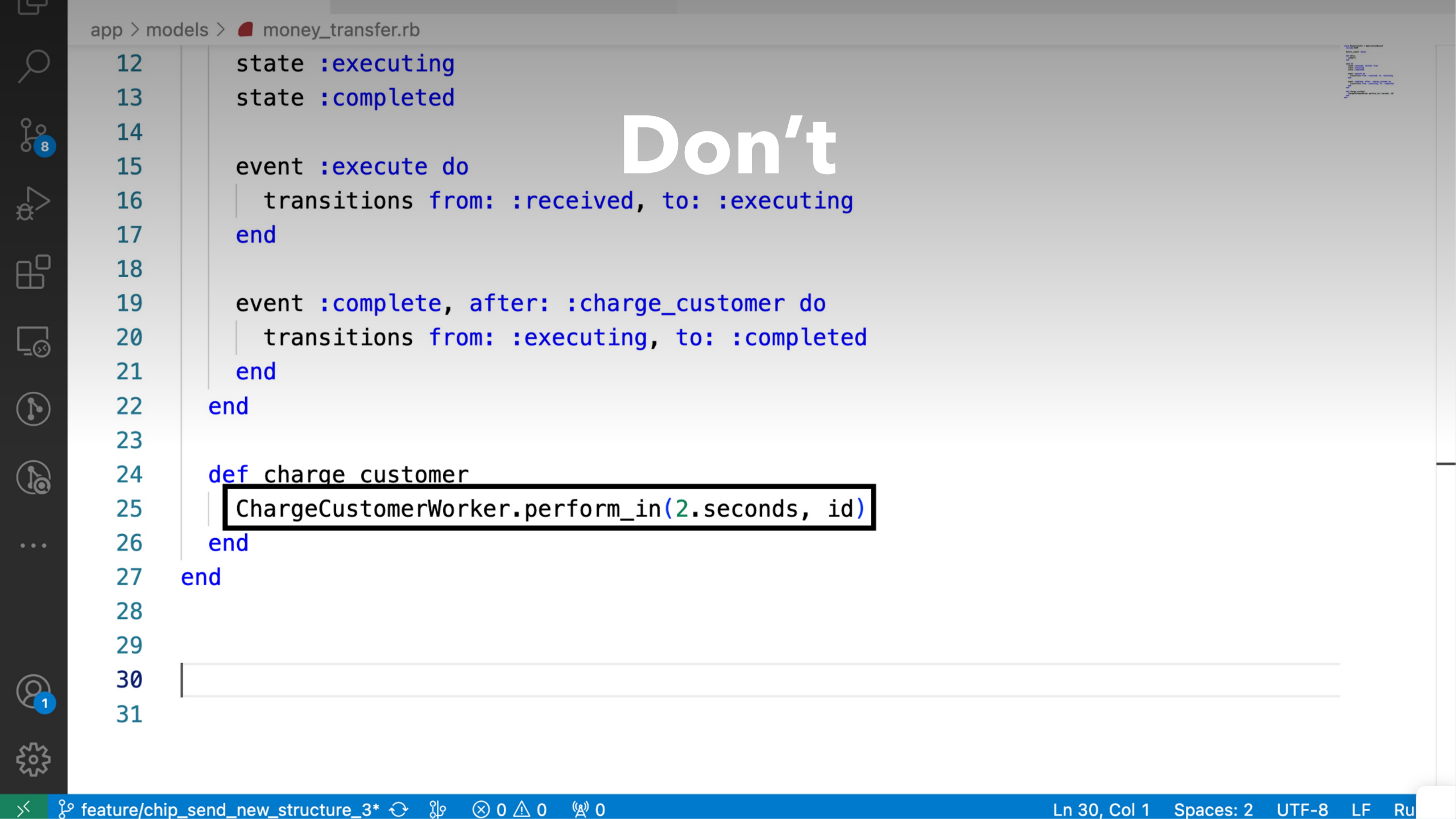
Use callback **after_commit** instead of **after**.

Don’t attempt to add delay to the job!

app > models > money_transfer.rb

```ruby
class MoneyTransfer < ApplicationRecord
  include AASM

  before_commit :delay

  def delay
    sleep(2)
  end

  aasm do
    state :received, initial: true
    state :executing
    state :completed

    event :execute do
      transitions from: :received, to: :executing
    end

    event :complete, after_commit: :charge_customer do
      transitions from: :executing, to: :completed
    end
  end
```

Do


app > models > money_transfer.rb
state :executing
state :completed
event :execute do
transitions from: :received, to: :executing
end
event :complete, after: :charge_customer do
transitions from: :executing, to: :completed
end
end
def charge_customer
ChargeCustomerWorker.perform_in(2.seconds, id)
end
end
Don't
feature/chip_send_new_structure_3*
Ln 30, Col 1
Spaces: 2
UTF-8
LF

Thanks!