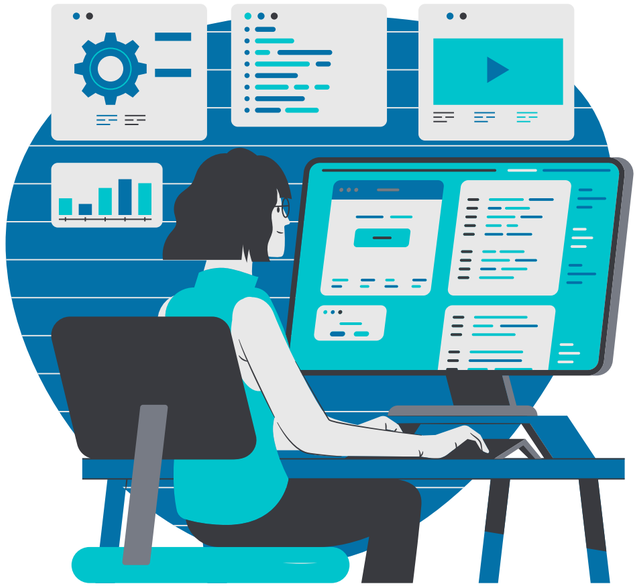
IIC2113

Grupo 20



ENTREGAS

Entrega 2

Entrega 3

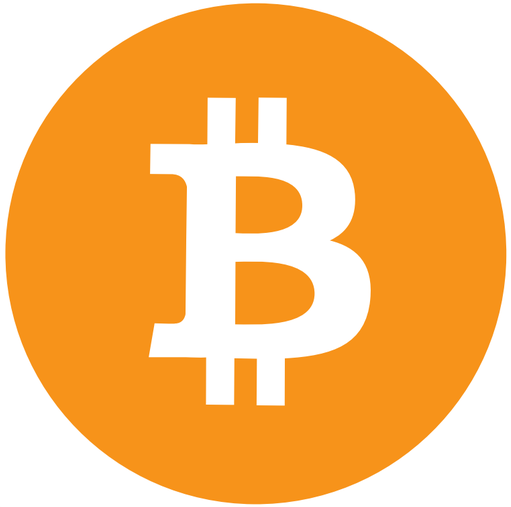
ENTREGAS

Entrega 2

Entrega 3

REFACTORING

Entrega 3

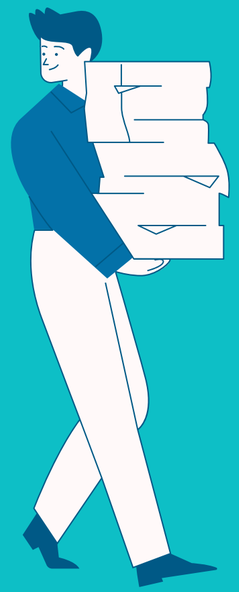


ANTES...

```ruby
class BankTransfersController < InheritedResources::Base
  before_action :authenticate_user!
  before_action :get_user_bank_account
  before_action :get_user_wallet

  def check
    bank_token = HTTParty.post(
      'http://www.bank8.tk/api/v1/auth/sign-in',
      :body => {
        :user => {
          :rut => ENV["RUT"],
          :password => ENV["NOTIFICATION_EMAIL_PASSWORD"]
        }
      }.to_json,
      :headers => { 'Content-Type' => 'application/json' }
    ).header["Authorization"]
    date = @bank_account.last_transfer
    if !date
      response = HTTParty.get(
        'http://www.bank8.tk/api/v1/transfers',
        :headers => {
          'Content-Type' => 'application/json',
          'Authorization' => bank_token
        }
      )
      received_transfers = response["received_transfers"].select {
        |transfer| transfer["sender_account"].to_s == @bank_account.account_number
      }
      received_transfers.each do |transfer|
        transfer_account(transfer)
      end
      @bank_account.last_transfer = Time.now()
      @bank_account.save
      @user_wallet.save
      render json: received_transfers
    else
      response = HTTParty.get(
        'http://www.bank8.tk/api/v1/transfers',
        :query => {
          :start => date.strftime("%Y-%m-%d")
        },
        :headers => {
          'Content-Type' => 'application/json',
          'Authorization' => bank_token
        }
      )
      received_transfers = response["received_transfers"].select {
        |transfer| transfer["sender_account"].to_s == @bank_account.account_number and
        DateTime.parse(transfer["created_at"]) > @bank_account.last_transfer
      }
      received_transfers.each do |transfer|
        transfer_account(transfer)
      end
      @bank_account.last_transfer = Time.now()
      @bank_account.save
      @user_wallet.save
      render json: received_transfers
    end
  end
```

```ruby
class BankTransfersController < InheritedResources::Base
  before_action :authenticate_user!
  before_action :get_user_bank_account
  before_action :get_user_wallet

  def check
    bank_token = HTTParty.post(
      'http://www.bank8.tk/api/v1/auth/sign-in',
      :body => {
        :user => {
          :rut => ENV["RUT"],
          :password => ENV["NOTIFICATION_EMAIL_PASSWORD"]
        }
      }.to_json,
      :headers => { 'Content-Type' => 'application/json' }
    ).header["Authorization"]
    date = @bank_account.last_transfer
    if !date
      response = HTTParty.get(
        'http://www.bank8.tk/api/v1/transfers',
        :headers => {
          'Content-Type' => 'application/json',
          'Authorization' => bank_token
        }
      )
      received_transfers = response["received_transfers"].select {
        |transfer| transfer["sender_account"].to_s == @bank_account.account_number
      }
      received_transfers.each do |transfer|
        transfer_account(transfer)
      end
      @bank_account.last_transfer = Time.now()
      @bank_account.save
      @user_wallet.save
      render json: received_transfers
    else
      response = HTTParty.get(
        'http://www.bank8.tk/api/v1/transfers',
        :query => {
          :start => date.strftime("%Y-%m-%d")
        },
        :headers => {
          'Content-Type' => 'application/json',
          'Authorization' => bank_token
        }
      )
      received_transfers = response["received_transfers"].select {
        |transfer| transfer["sender_account"].to_s == @bank_account.account_number and
        DateTime.parse(transfer["created_at"]) > @bank_account.last_transfer
      }
      received_transfers.each do |transfer|
        transfer_account(transfer)
      end
      @bank_account.last_transfer = Time.now()
      @bank_account.save
      @user_wallet.save
      render json: received_transfers
    end
  end
```

BIG METHOD

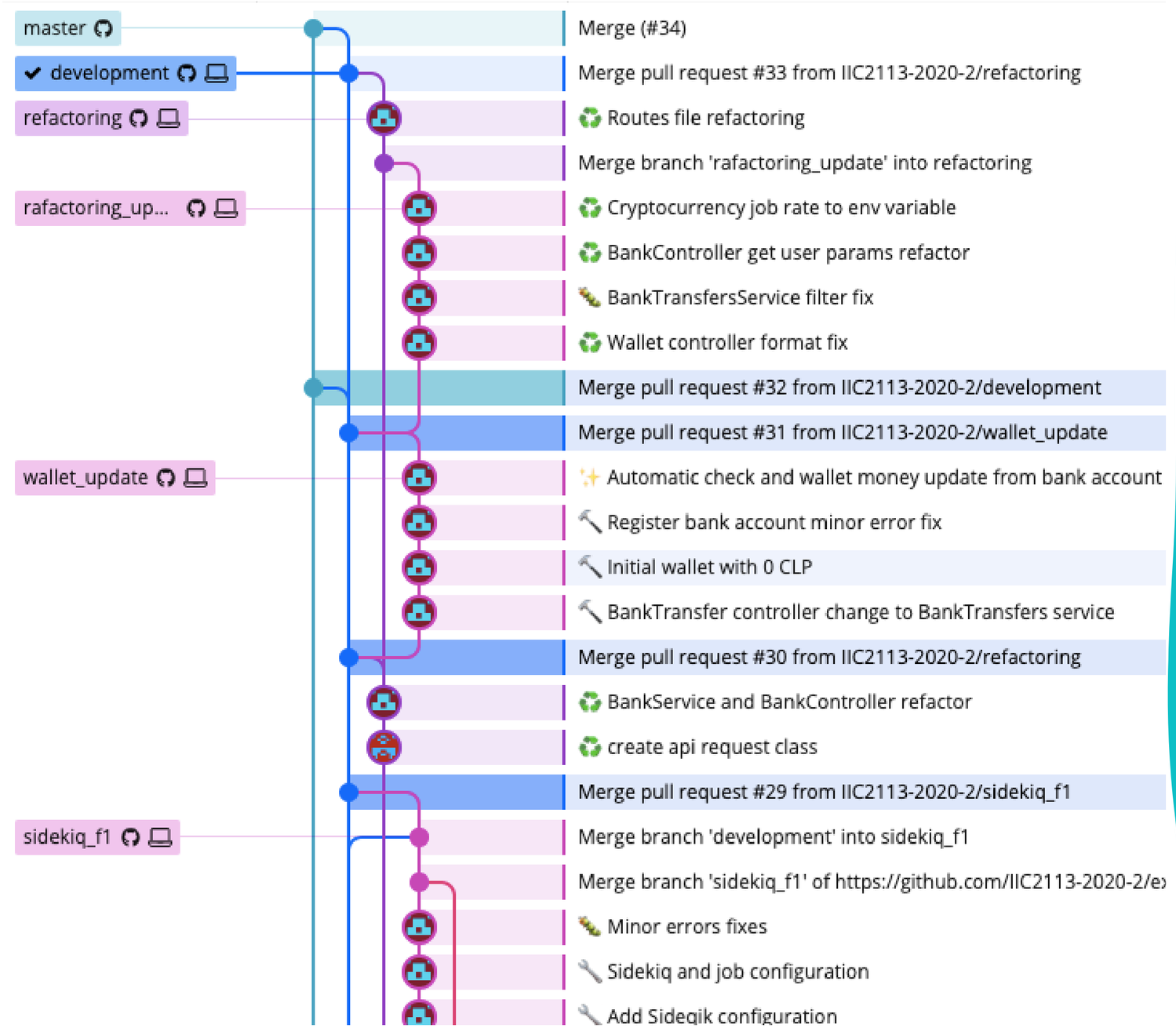


CÓDIGO DUPLICADO

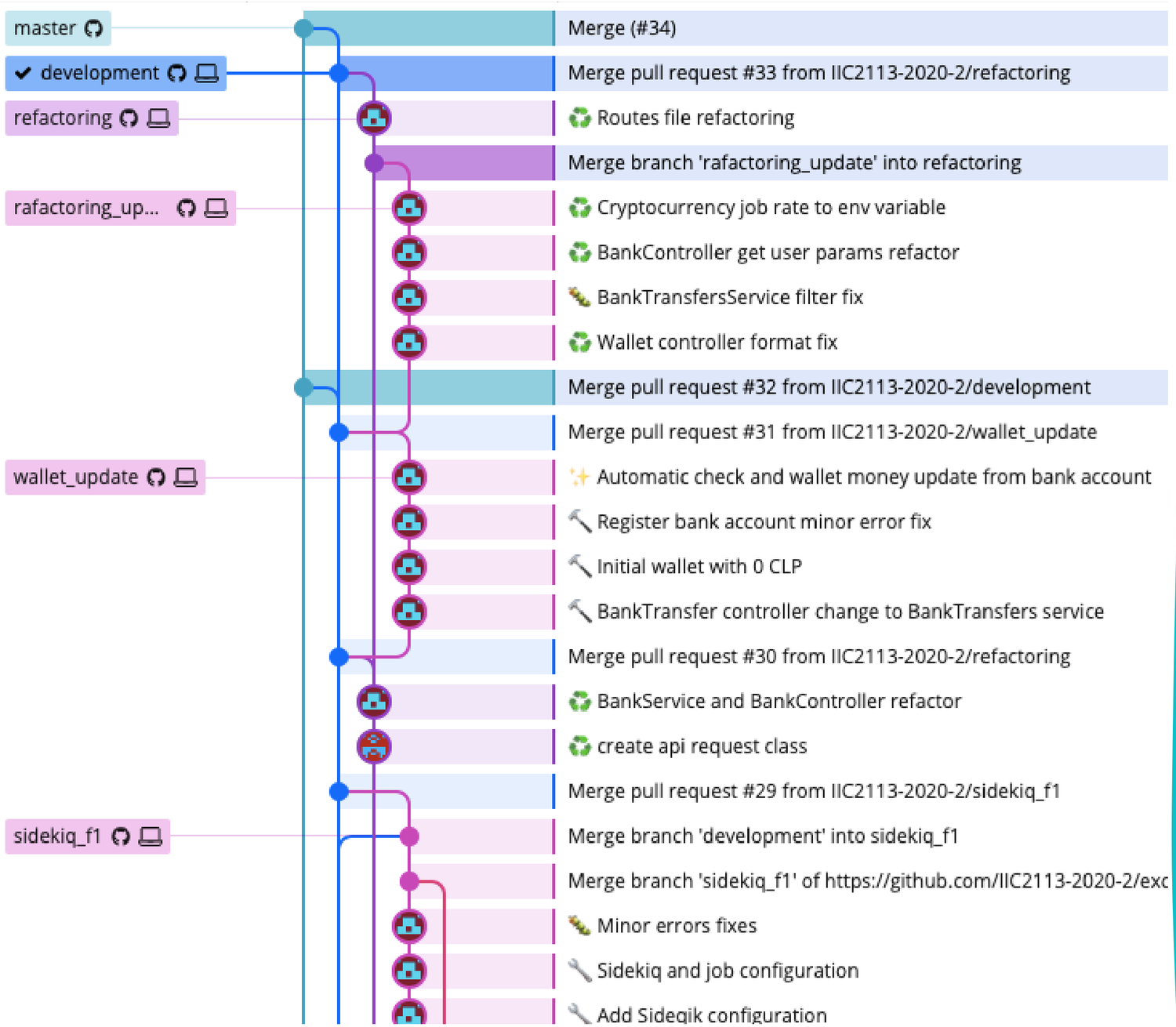
```ruby
class BankTransfersController < InheritedResources::Base
  before_action :authenticate_user!
  before_action :get_user_bank_account
  before_action :get_user_wallet

  def check
    bank_token = HTTParty.post(
      'http://www.bank8.tk/api/v1/auth/sign-in',
      :body => {
        :user => {
          :rut => ENV["RUT"],
          :password => ENV["NOTIFICATION_EMAIL_PASSWORD"]
        }
      }.to_json,
      :headers => { 'Content-Type' => 'application/json' }
    ).header["Authorization"]
    date = @bank_account.last_transfer
    if !date
      response = HTTParty.get(
        'http://www.bank8.tk/api/v1/transfers',
        :headers => {
          'Content-Type' => 'application/json',
          'Authorization' => bank_token
        }
      )
      received_transfers = response["received_transfers"].select {
        |transfer| transfer["sender_account"].to_s == @bank_account.account_number
      }
      received_transfers.each do |transfer|
        transfer_account(transfer)
      end
      @bank_account.last_transfer = Time.now()
      @bank_account
      @user_wallet.save
      render json: received_transfers
    else
      response = HTTParty.get(
        'http://www.bank8.tk/api/v1/transfers',
        :query => {
          :start => date.strftime("%Y-%m-%d")
        },
        :headers => {
          'Content-Type' => 'application/json',
          'Authorization' => bank_token
        }
      )
      received_transfers = response["received_transfers"].select {
        |transfer| transfer["sender_account"].to_s == @bank_account.account_number and
        DateTime.parse(transfer["created_at"]) > @bank_account.last_transfer
      }
      received_transfers.each do |transfer|
        transfer_account(transfer)
      end
      @bank_account.last_transfer = Time.now()
      @bank_account.save
      @user_wallet.save
      render json: received_transfers
    end
  end
```

```ruby
require_relative "./bank_service"

class BankTransfersService

  def initialize(current_user)
    @current_user = current_user
    @bank_account = current_user.bank_accounts.first
    @user_wallet = current_user.wallet
  end

  def check_user_transfers
    bank_token = BankService.get_bank_token(nil)
    date = @bank_account.last_transfer
    response = BankService.get_transactions_data(date, bank_token)
    received_transfers = get_recieved_transfers(response, @bank_account, date)
    return update_received_transfers(received_transfers)
  end

  protected

  def get_recieved_transfers(response, bank_account, date)
    if !date
      return response["received_transfers"].select {
        |transfer| (transfer["sender_account"].to_s == bank_account.account_number) and
                   (transfer["status"] == "completed")
      }
    else
      return response["received_transfers"].select {
        |transfer| (transfer["sender_account"].to_s == bank_account.account_number) and
                   (DateTime.parse(transfer["created_at"]) > date) and
                   (transfer["status"] == "completed")
      }
    end
  end

  def update_received_transfers(received_transfers)
    received_transfers.each do |transfer|
      transfer_account(transfer)
    end
    @bank_account.last_transfer = Time.now()
    @bank_account.save
    @user_wallet.save
    return received_transfers
  end

  def transfer_account(transfer)
    if transfer["amount"]
      @user_wallet.clp += transfer["amount"]
    end
  end

end
```

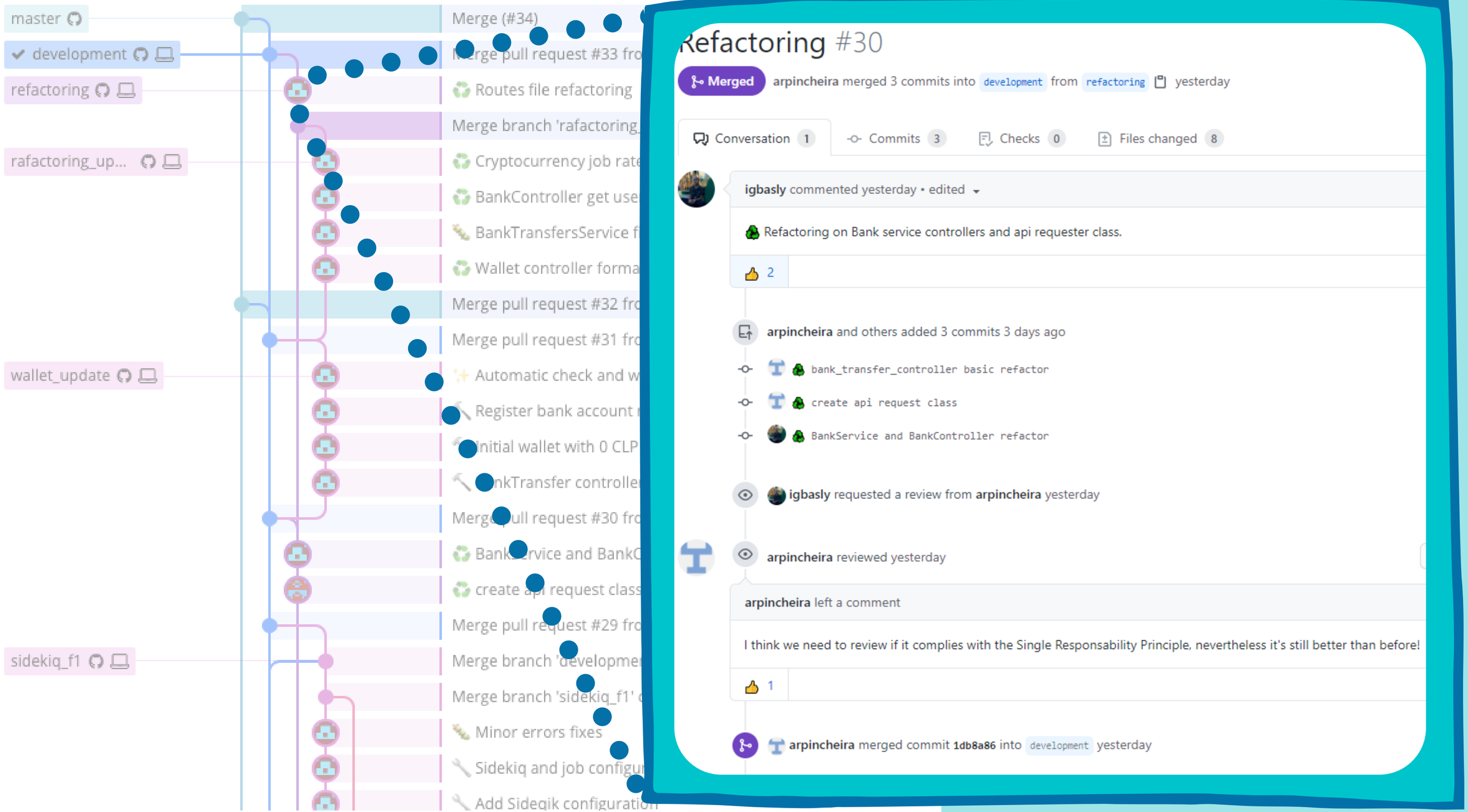
DESPUÉS



FLUJO DE TRABAJO



CICLO DE REFACTORING


master
development
refactoring
rafactoring_up...
wallet_update
sidekiq_f1
Merge (#34)
Routes file refactoring
Merge branch 'rafactoring_
Cryptocurrency job rate
BankController get use
BankTransfersService f
Wallet controller forma
Merge pull request #32 fro
Merge pull request #31 fro
Automatic check and w
Register bank account
Initial wallet with 0 CLP
nkTransfer controlle
Merge pull request #30 fro
Bank ervice and BankC
create api request class
Merge pull request #29 fro
Merge branch 'developme
Merge branch 'sidekiq_f1'
Minor errors fixes
Sidekiq and job configu
Add Sidekiq configuration
Refactoring #30
Merged
arpincheira merged 3 commits into development from refactoring yesterday
Conversation 1
Commits 3
Checks 0
Files changed 8
igbasly commented yesterday • edited
Refactoring on Bank service controllers and api requester class.
2
arpincheira and others added 3 commits 3 days ago
bank_transfer_controller basic refactor
create api request class
BankService and BankController refactor
igbasly requested a review from arpincheira yesterday
arpincheira reviewed yesterday
arpincheira left a comment
I think we need to review if it complies with the Single Responsability Principle, nevertheless it's still better than before!
1
arpincheira merged commit 1db8a86 into development yesterday

Conclusiones

Tiempo para refactorizar

Documentación clara

Buenas prácticas desde un inicio

MUCHAS GRACIAS