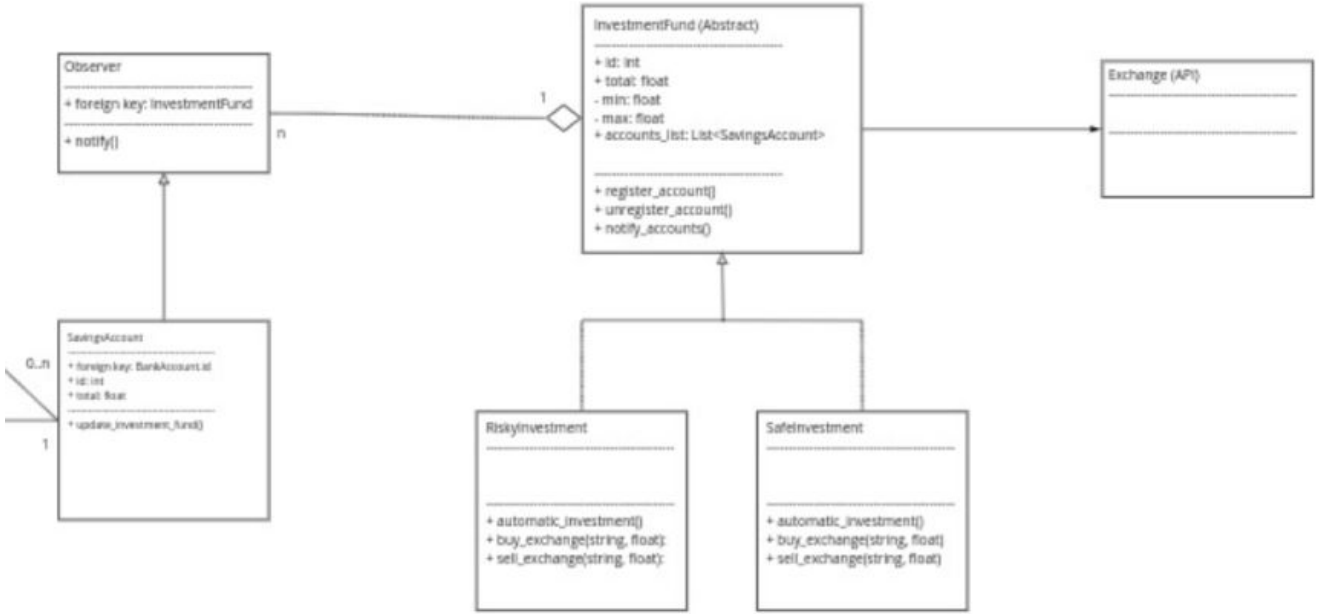
Presentación Final

Grupo 22 - Bank Ten

Elementos Relevantes

Diagrama de Componentes



Single Table Inheritance

```ruby
class InvestmentFund < ApplicationRecord
  # The set defaults will only work if the object is new
  after_initialize :set_defaults, unless: :persisted?

  MESS = "SYSTEM ERROR: method missing"

  def update_balance(amount)
    self.balance += amount

    # this change should not affect observers,
    # so dont notify them
    dont_notify_observers_and_save
  end

  def simulate_investment; raise MESS; end
  def buy_exchange; raise MESS; end
  def sell_exchange; raise MESS; end

private
  def set_defaults
    self.balance = 0 if balance.nil?
  end

  # this function is not optimized, since it makes to saves to the database instead of one
  def notify_observers_and_save
    self.notify_observers = true
    self.save
    dont_notify_observers_and_save
  end

  def dont_notify_observers_and_save
    self.notify_observers = false
    self.save
  end

  def random_investment(amount)
    if rand < 0.5
      # decrease balance
      change_balance(-amount)
    else
      # increment balance
      change_balance(amount)
    end
  end
```

```ruby
class SafeInvestment < InvestmentFund
  include PowerTypes::Observable
  validate :check_record, on: :create

  def simulate_investment
    # percentage of change from simulated investment
    # 0.5 factor makes this investment fund 'safe', since it only works wit
    delta = self.balance * 0.03 * 0.5

    random_investment(delta)

    notify_observers_and_save
  end

private
  def check_record
    if SafeInvestment.all.count === 1
      errors[:base] << "You can only have one active Safe Investment Fund"
    end
  end
end
```

```ruby
class RiskyInvestment < InvestmentFund
  include PowerTypes::Observable
  validate :check_record, on: :create

  def simulate_investment
    # percentage of change from simulated investment
    delta = self.balance * 0.03

    random_investment(delta)

    notify_observers_and_save
  end


private
  def check_record
    if RiskyInvestment.all.count === 1
      errors[:base] << "You can only have one active Risky Investment Fund"
    end
  end
end
```

Automatización y Patrón ‘Observador’

```ruby
class SavingsAccount < ApplicationRecord
  @valid_types = %w[RISKY SAFE]

  before_validation :up_case_fields

  # The set_defaults will only work if the object is new
  after_initialize :set_defaults, unless: :persisted?
  validates :account_number, presence: true, uniqueness: true, length: { is: 19 }
  validates :investment_type, presence: true
  validates :investment_type, inclusion: { in: @valid_types }

  belongs_to :bank_account
  has_many :account_movements, dependent: :destroy

  def update_account_and_investment_fund(amount)
    # Update this account's balance
    self.balance += amount
    self.save

    # Get associated investment fund
    investment_fund = nil
    case self.investment_type
    when 'RISKY'
      # Get Risky investment fund and update balance
      investment_fund = RiskyInvestment.all.take
    when 'SAFE'
      # Get Safe investment fund and update balance
      investment_fund = SafeInvestment.all.take
    else
      logger.error 'Error: SavingsAccount:update_investment_fund - Fell out of case,
    end

    # Update investment fund's balance
    investment_fund.update_balance(amount)
  end

  def update_share(last_fund_balance, fund_balance)
    # Calculate share percentage
    share_percentage = self.balance/last_fund_balance
    # Adjust balance to new fund_balance

    self.balance = share_percentage * fund_balance

    self.save
  end
```

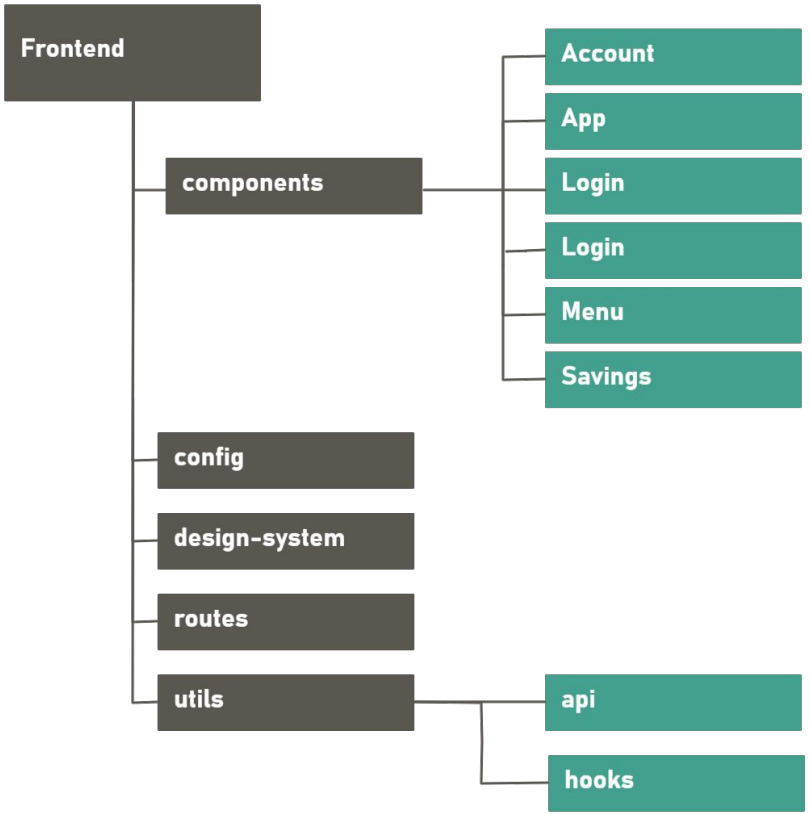
```ruby
class SafeInvestmentObserver < PowerTypes::Observer
  after_save :run

  def run
    # Check if investment fund changes should affect all associated savings accounts
    if object.notify_observers
      safe_accounts = SavingsAccount.where(investment_type: 'SAFE')
      safe_accounts.each do |account|
        account.update_share(object.last_balance, object.balance)
      end
    end
  end
end
```

```ruby
desc "This task is called by the Heroku scheduler add-
task :automatic_investment => :environment do
  puts "Simulating real investments..."
  risky_fund = RiskyInvestment.all.take
  safe_fund = SafeInvestment.all.take

  risky_fund.simulate_investment
  safe_fund.simulate_investment
  puts "done."
end
```

Estructura del proyecto



Rutas de la aplicación

account.js

```
const root = "/account";

export const accountRoute = root;
export const investmentRoute = `${root}/investement`;
```

home.js

```
const root = "/";

export const homeRoute = root;
```

saving_account.js

```
const root = "/saving_account";

export const savingAccountRoute = root;
export const newSavingAccountRoute = `${root}/new`;
```

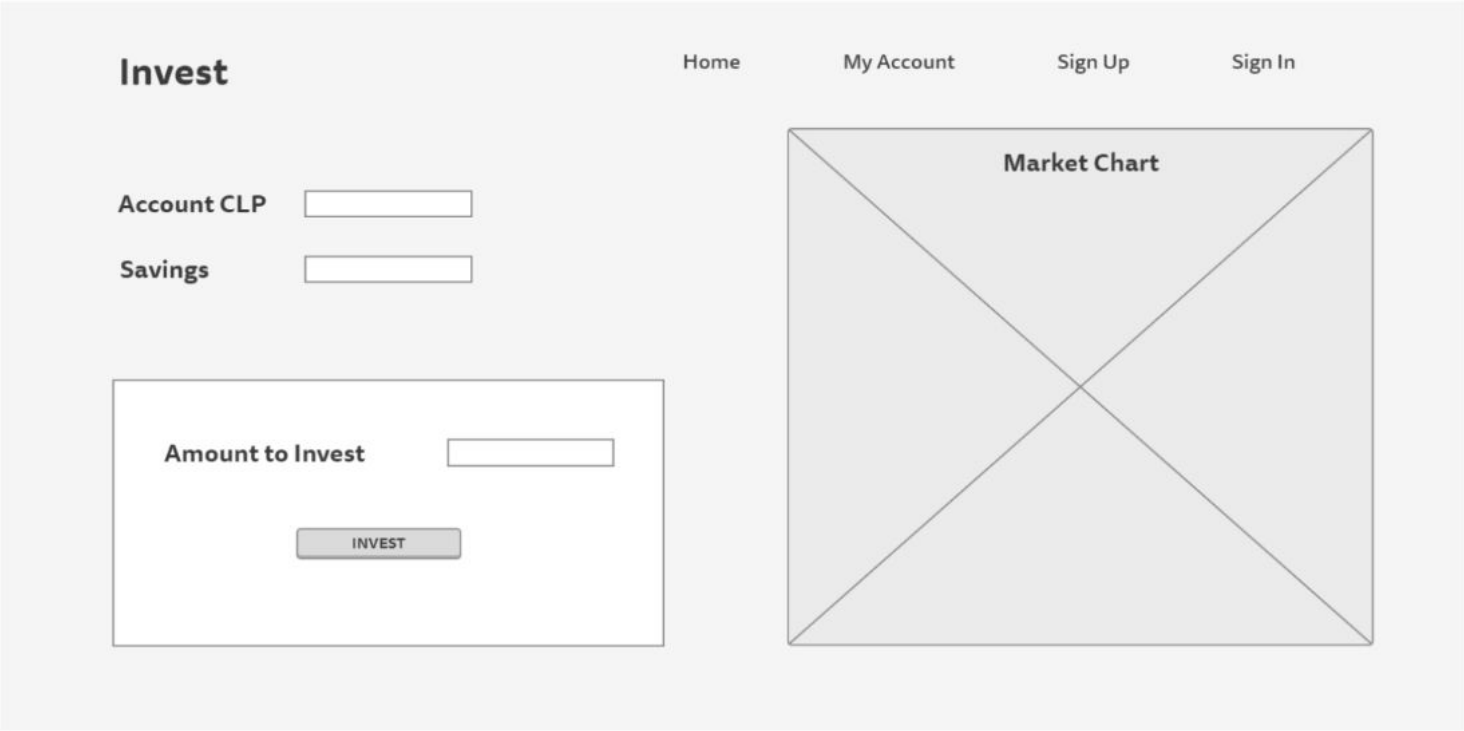
transaction.js

```
const root = "/transaction";

export const transactionRoute = root;
export const newTransactionRoute = `${root}/new`;
```

Problemas Encontrados

Wireframe


Invest
Home
My Account
Sign Up
Sign In
Market Chart
Account CLP
Savings
Amount to Invest
INVEST

Refactor

Code smell: Long method	métodos auxiliares
Mantenibilidad y escalabilidad: 2 entradas nuevas en la tabla por operación	Cambios en el modelo

```ruby
def create
  @user_account = current_user.bank_account
  msg = nil
  error = false
  if params.require(:transfer).require(:amount) > @user_account.balance
    error = true
    msg = "You don't have enough money to transfer that amount"
  elsif @dest_account.nil?
    error = true
    msg = "Destination account does not exist."
  end
  if error
    render json: {
      messages: msg,
      is_success: false,
      data: {}
    }, status: :bad_request
    return
  end
  transfer = Transfer.new transfer_params
  transfer.transfer_type = "transfer"
  transfer.bank_account = @user_account
  transfer.other_user = "#{@dest_account.user.name} #{@dest_account.user.lastname}"
  transfer.save
  @user_account.balance -= transfer.amount
  @user_account.save

  deposit = Transfer.new transfer_params
  deposit.transfer_type = "deposit"
  deposit.bank_account = @dest_account
  deposit.other_user = "#{current_user.name} #{current_user.lastname}"
  deposit.save
  @dest_account.balance += deposit.amount
  @dest_account.save
```

Presentación Final

Grupo 22 - Bank Ten