soda App

最終更新 6/1

主な構成要素

- app/
  - models/ #モデル内訳
    - income_category.rb #収入カテゴリモデル
    - income.rb #収入
    - outcome_category.rb #支出カテゴリ
    - outcome.rb #支出
    - user.rb #ユーザー
    - want.rb #やりたいこと
  - uploaders/
    - image_uploader.rb #画像アップロードモデル
  - controllers/ #コントローラーファイル
    - admin/
      - base_controller.rb #管理者ログインの設定
      - sessions_contoller.rb
    - application_controller.rb #ログイン機能の設定
    - homes_controller.rb #ログイン後のトップページコントローラ
    - income_categories_controller.rb #収入カテゴリ
    - incomes_controller.rb #収入
    - omniauth_callbacks_controller.rb #LINEログイン
    - outcome_categories_controller.rb #支出カテゴリ
    - outcomes_controller.rb #支出
    - users_controller.rb #ユーザー
    - wants_controller.rb #やりたいこと
  - views/ #ビューファイル
    - dashboard/
      - index.html.erb #管理者の一覧ビュー
      - _header.html.erb #ヘッダー
    - registrations/
      - new.html.erb #管理者新規作成（名前、メール）
    - sessions/
      - new.html.erb #管理者ログイン
    - devise/
      - registrations/new.html.erb #ユーザー新規作成画面（現時点ではLINEログインのみ有効）
      - sessions/new.html.erb #ユーザーログイン画面
    - homes/index.html.erb #ログイン後のトップページ(タブform,list)
    - income_categories/ #収入カテゴリの一覧、追加、編集

        - incomes/ #収入の一覧、新規作成、詳細、編集
        - layouts/ #全体のレイアウト
            - _header.html.erb #トップページのヘッダーテンプレート
            - _footer.html.erb #トップページのフッターテンプレート
            - _form.html.erb #トップ・タブ切り替え（支出の入力画面）
            - _list.html.erb #トップ・タブ切り替え（収入の入力画面）
            - _month_outcome.html.erb # マイページ・月別レポート（month_outcome）
            - _month_income.html.erb # (month_outcome)
            - _year.html.erb #マイページ・タブ切り替え（年間レポート）
            - admin.html.erb #管理者画面のレイアウト(dashboard)
            - application.html.erb #ユーザー画面のレイアウト
        - outcome_categories/ #支出カテゴリの一覧、追加、編集
        - outcomes/ #支出の一覧、新規作成、詳細、編集
        - users/
            - edit.html.erb #ユーザーの名前を編集
            - mypage.html.erb #マイページ画面のレイアウト(タブmonth,year)
            - month.html.erb #マイページ月別のタブ切り替え画面
        - wants/ #やりたいことの一覧、追加、編集
- config/ #設定ファイル
    - devise.rb #管理者ログイン時に名前で認証できる
    - rotes.rb #ルーティングの設定
- db/ #データベース関連のファイル
    - migrate/ #マイグレーションファイル
        - 20240425052831_devise_create_users.rb #ユーザーのテーブル作成マイグレーション
        - 20240425055230_add_column_to_users.rb #ユーザーにLINEログインに必要なカラムを追加
        - 20240425064523_create_income_categories.rb #収入カテゴリのテーブル作成
        - 20240425064744_create_outcome_categories.rb #支出カテゴリのテーブル作成
        - 20240425064841_create_incomes.rb #収入テーブル
        - 20240425064852_create_outcomes.rb #支出テーブル
        - 20240425064904_create_wants.rb #やりたいことテーブル
        - 20240501022026_add_role_to_user.rb #ユーザーのテーブルに管理者用のカラム(role)を追加
    - scheme.rb #テーブルの詳細
    - seeds.rb #収入カテゴリ、支出カテゴリの初期設定
- Gemfile #使用するGemのリスト

    - gem 'rails', '5.2.4.2' #Ruby on Railsのバージョン5.0系を使用
    - gem 'devise' #認証機能を提供するDevise
    - gem 'omniauth-line' #LINEログイン機能を追加
    - gem 'omniauth-rails_csrf_protection'
    - gem 'dotenv-rails' #環境変数を設定
    - gem 'bootstrap', '~> 4.6.0' #bootstrapを導入
    - gem 'jquery-rails' #jqueryを導入
    - gem 'carrierwave' #画像機能を追加
    - gem 'mini_magick'

- `README.md` #プロジェクトの概要と説明

- `docs/`

    - `description.md` #プロジェクトの詳細説明
    - `description.pdf` #プロジェクトの詳細説明のPDF

- `ER.md` #ER図

主要なコードスニペット

app/models/user.rb

Userモデル

```
class User < ApplicationRecord
  #ユーザーと収入、支出、やりたいことを関連づけ
  has_many :incomes
  has_many :outcomes
  has_many :wants

  #ログイン時のバリデーションを設定
  validates :name,   presence: true
  validates :role,  inclusion: {in: [true, false]}

  # deviseの設定、LINEと連携
  devise :database_authenticatable, :registerable,
         :recoverable, :rememberable, :validatable,
         :omniauthable, omniauth_providers: %i[line]

  def social_plofile(provider)
    social_plofile.select { |sp| sp.provider == provider.to_s }.first
  end

  def set_values(omniauth)
    return if provider.to_s != omniauth["provider"].to_s || uid !=
omniauth["uid"]
    credentials = omniauth["credentials"]
    info = omniauth["info"]

    access_token = credentials["refresh_token"]
```

```
    access_secret = credentials["secret"]
    credentials = credentials.to_json
    name = info["name"]
  end

  def set_values_by_raw_info(raw_info)
    self.raw_info = raw_info.to_json
    self.save!
  end

  #管理者とユーザーを区別
  def admin?
    role == true
  end
end
```

/app/controllers/users_controller.rb

Userコントローラー

```
class UsersController < ApplicationController
  before_action :authenticate
  before_action :set_user

  def mypage

    # 収入の総計
    @income_total = current_user.incomes.sum(:price)
    @outcome_total = current_user.outcomes.sum(:price)
    @balance = @income_total - @outcome_total

    # 年、月の取得
    @year = Date.today.year
    @month = Date.today.month

    # 月毎の収支
    # 月ごとの収入
    @monthly_incomes = (1..12).map do |month|
      start_date = Date.new(@year, month, 1)
      end_date = start_date.end_of_month
      category_incomes = {}

      IncomeCategory.all.each do |income_category|
        category_income_price = current_user.incomes.where(created_at:
start_date..end_date, income_category_id: income_category.id).sum(:price)
        category_incomes[income_category.name] = category_income_price
      end
        {
          month: start_date.strftime("%-m月"),
          total_incomes: current_user.incomes.where(created_at:
start_date..end_date).sum(:price),
```

```
          category_incomes: category_incomes
        }
    end

    # 月ごとの支出
    @monthly_outcomes = (1..12).map do |month|
      start_date = Date.new(@year, month, 1)
      end_date = start_date.end_of_month
      category_outcomes = {}

      OutcomeCategory.all.each do |outcome_category|
        category_outcome_price = current_user.outcomes.where(created_at:
start_date..end_date, outcome_category_id:
outcome_category.id).sum(:price)
        category_outcomes[outcome_category.name] = category_outcome_price
      end
        {
          month: start_date.strftime("%-m月"),
          total_outcomes: current_user.outcomes.where(created_at:
start_date..end_date).sum(:price),
          category_outcomes: category_outcomes
        }
    end


    # 当月の収支
    current_month_income = @monthly_incomes.find { |income| income[:month]
== Date.current.strftime("%-m月") }
    @current_month_income_value = current_month_income[:total_incomes]

    current_month_outcome = @monthly_outcomes.find { |outcome|
outcome[:month] == Date.current.strftime("%-m月") }
    @current_month_outcome_value = current_month_outcome[:total_outcomes]
    @current_month_outcome_name = current_month_outcome[:total_outcomes]

    # 年間収支
    @yearly_incomes_total  = @monthly_incomes.sum { |income|
income[:total_incomes] }
    @yearly_outcomes_total = @monthly_outcomes.sum { |outcome|
outcome[:total_outcomes] }
    @yearly_balance = @yearly_incomes_total - @yearly_outcomes_total
  end

  def month
    (mypageと同じ)
  end

  def edit
    @user = User.find(current_user.id)
  end

  def update
    @user = User.find(current_user.id)
    @user = @user.update(user_params)
```

```
    redirect_to homes_path
  end

  private
  def set_user
    @user = current_user
  end

  def user_params
    params.permit(:name)
  end

end
```

/app/controllers/omniauth_callbacks_controller.rb

LINEログインの設定

```
class OmniauthCallbacksController < Devise::OmniauthCallbacksController
  def line; basic_action end

  private
  def basic_action
    @omniauth = request.env["omniauth.auth"]
    if @omniauth.present?
      @profile = User.find_or_initialize_by(provider:
@omniauth["provider"], uid: @omniauth["uid"])
      if @profile.email.blank?
        email = @omniauth["info"]["email"] ? @omniauth["info"]["email"] :
"#{@omniauth["uid"]}-#{@omniauth["provider"]}@example.com"
        @profile = current_user || User.create!(
          provider: @omniauth["provider"],
          uid: @omniauth["uid"],
          email: email,
          name: @omniauth["info"]["name"],
          password: Devise.friendly_token[0, 20]
          )
      end
      @profile.set_values(@omniauth)
      sign_in(:user, @profile)
    end
    flash[:notice] = "ログインしました"
    redirect_to homes_path
  end

  def fake_email(uid, provider)
    "#{auth.uid}-#{auth.provider}@example.com"
  end
end
```

/app/controllers/admin/base_controller.rb

管理者ログインの設定

```
class Admin::BaseController < ApplicationController
  before_action :authenticate_user!
  before_action :check_admin
  before_action :configure_permitted_parameters, if: :devise_controller?
  layout "admin"

  private

  def check_admin
    unless current_user&.admin?
      redirect_to homes_path, alert: '権限がありません。'
    end
  end
end
```

/app/controllers/admin/sessions_controller.rb

管理者ログイン機能

```
class Admin::SessionsController < Devise::SessionsController
  before_action :configure_permitted_parameters, only:
[:new,:create,:destroy]

  #オーバーライド
  # GET /resource/sign_in
  def new
    super
  end

  # POST /resource/sign_in
  def create
    super
  end

  protected

  # ログイン後のリダイレクト先を設定
  def after_sign_in_path_for(resource_or_scope)
    admin_root_url
  end

  def after_sign_out_path_for(resource)
    admin_login_path
  end
```

```
end
```

/app/controllers/application_controller.rb

ログイン機能の設定

```
class ApplicationController < ActionController::Base
  before_action :configure_permitted_parameters, if: :devise_controller?

  def authenticate
    # 登録していないユーザーはログイン画面に戻る
    redirect_to new_user_session_url unless user_signed_in?
  end

  protected
    # 管理者ログイン後のリダイレクト先
    def after_sign_in_path_for(resource_or_scope)
      if current_user.role == true
        admin_root_path
      else
        homes_path
      end
    end

    # ログアウト後のリダイレクト先
    def after_sign_out_path_for(resource_or_scope)
      # ログイン画面へ遷移
      new_user_session_url
    end

    def configure_permitted_parameters
      # 管理者登録時に許可するパラメータを設定
      devise_parameter_sanitizer.permit(:sign_up, keys: [:name, :email,
:role])
      devise_parameter_sanitizer.permit(:sign_in, keys: [:name, :email,
:role])
      devise_parameter_sanitizer.permit(:account_update, keys: [:name,
:email, :role])
    end
  end
```

/app/models/outcome.rb

支出モデル

```
class Outcome < ApplicationRecord
  #支出カテゴリと、ユーザーに関連付けを行う
  belongs_to :outcome_category,   dependent: :destroy
  belongs_to :user

  #画像を追加できる
  mount_uploader :image, ImageUploader
end
```

/app/assets/javascripts/application.js

タブ切り替え機能

```
//= require rails-ujs
//= require activestorage
//= require turbolinks
//= require_tree .

//= require jquery3
//= require popper
//= require bootstrap-sprockets

// タブの切り替え
// turbolinksの無効化
$(document).on('turbolinks:load', function() {
$(function() {
  $('.tab').click(function(){
    $('.tab-active').removeClass('tab-active');
    $(this).addClass('tab-active');
    $('.box-show').removeClass('box-show');
    const index = $(this).index();
    $('.tabbox').eq(index).addClass('box-show');
  });
});
});
```