DREAMBOX

Team members: Alexander Bahas
Meggie Yun
Xiakan Xu
Kee Sern Chua
Levon Nie

What is Dreambox?

Dreambox is a place for posting and sharing your dreams to the world.

a. A user can record their dreams in their personal calendar in their DreamBox.

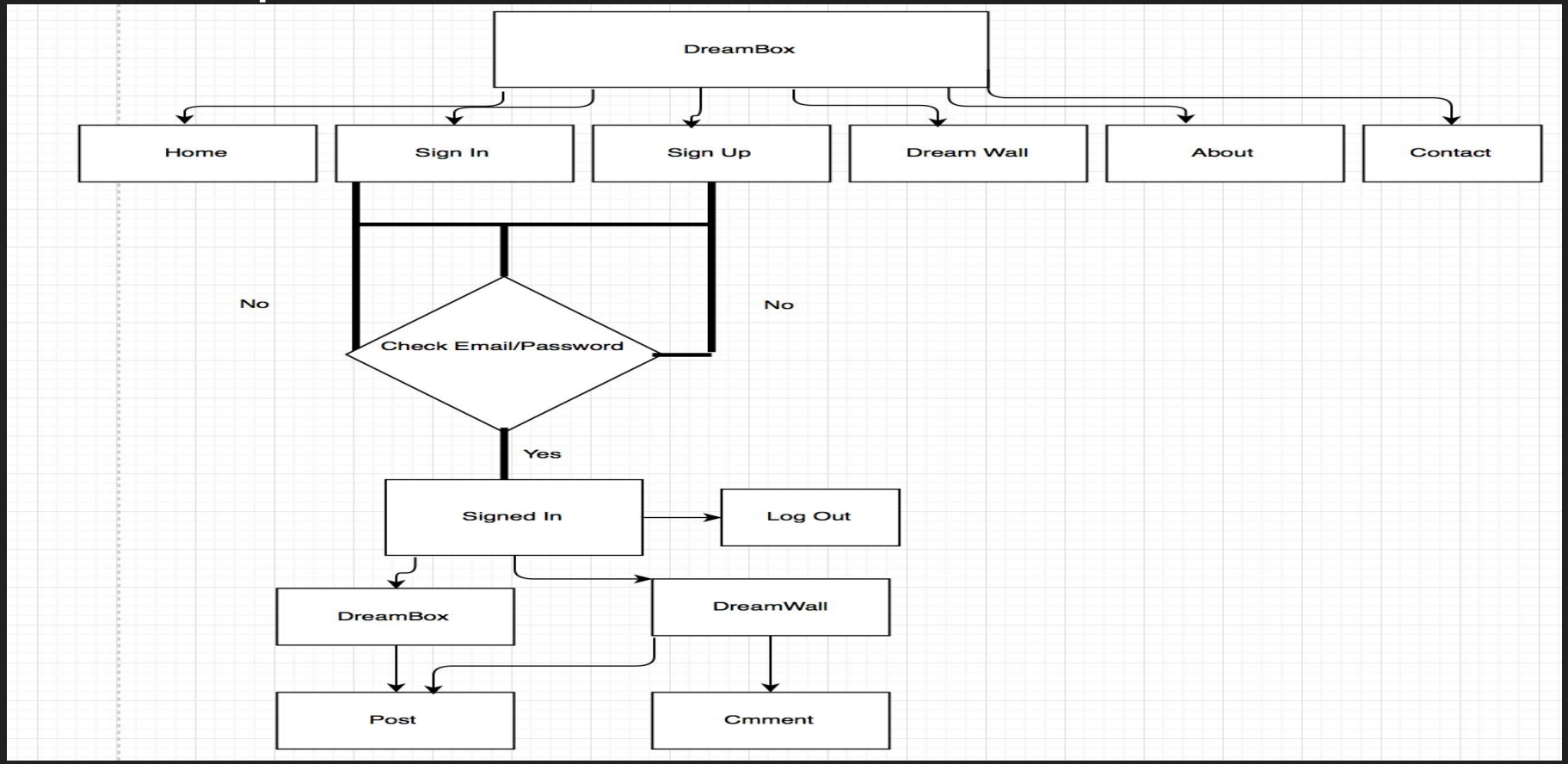
b. A user can post their dream on Wall of Dreams.

I. In the Wall of Dreams users can share their dreams with other users in Dreambox.

Features of Dreambox

- Sign in/out/up
- Wall of Dreams
  - Post New Dreams
  - Insert Photos
  - Add Comments
  - Edit/Delete Dreams
- Personal DreamBox
  - Private dream journal
  - Calendar
  - Search

Site Map


DreamBox
Home
Sign In
Sign Up
Dream Wall
About
Contact
No
Check Email/Password
No
Yes
Signed In
Log Out
DreamBox
DreamWall
Post
Cmment

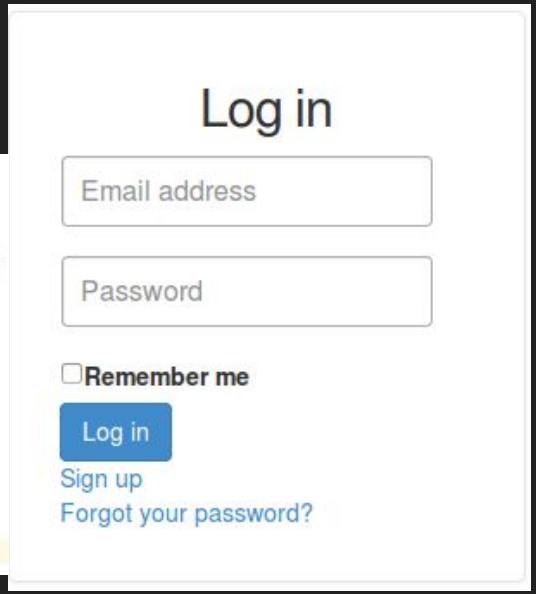
Technologies used

- SQLite3
- Bootstrap-SASS

Technologies used

- Devise(User Authentication)

```
<div class="border-form-div">
  <h2>Log in</h2>

  <%= simple_form_for(resource, as: resource_name, url: session_path(resource_name)) do |f| %>
      <%= f.email_field :email, required: false,:autofocus => true ,:placeholder => 'Email address'%>
      <%= f.password_field :password , required: false, :placeholder => 'Password'%>

    <div class="form-inputs">
      <%= f.input :remember_me, as: :boolean if devise_mapping.rememberable? %>
    </div>

    <%= f.submit "Log in",  :class=>'btn btn-primary'  %>

  <% end %>

  <%= render "devise/shared/links" %>
</div>
```

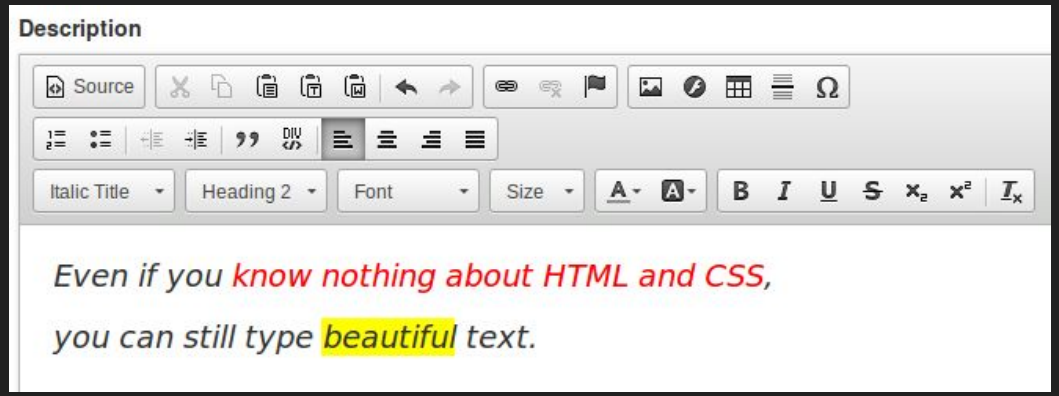


Technologies used

- Paperclip

Technologies used

- CKEditor

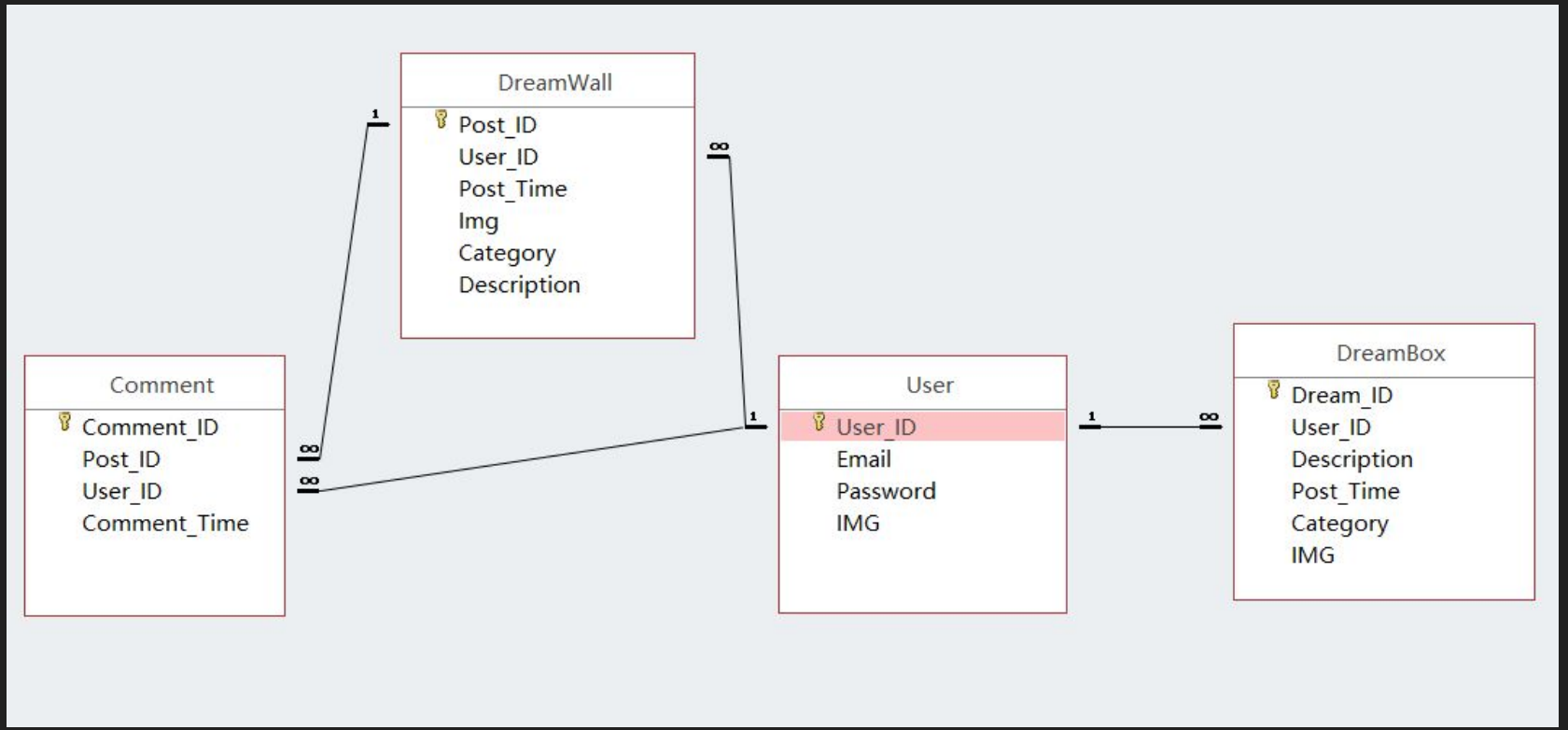


Technologies used

- Simple_Calendar


```
<div class="simple-calendar">
  <div class="col-md-8 col-md-offset-4">
  <%= link_to "<", calendar.url_for_previous_view , class:"btn btn-sm btn-primary"%>
  <Strong><%= I18n.t("date.month_names")[start_date.month] %> <%= start_date.year %></strong>
  <%= link_to ">", calendar.url_for_next_view , class:"btn btn-sm btn-primary"%>
  </div>
  <table class="table table-striped">
    <thead>
      <tr>
        <% date_range.slice(0, 7).each do |day| %>
          <th><%= I18n.t("date.abbr_day_names")[day.wday] %></th>
        <% end %>
      </tr>
    </thead>
```

ER diagram


DreamWall
Post_ID
User_ID
Post_Time
Img
Category
Description
Comment
Comment_ID
Post_ID
User_ID
Comment_Time
User
User_ID
Email
Password
IMG
DreamBox
Dream_ID
User_ID
Description
Post_Time
Category
IMG
1
∞
1
∞
∞
1
1
∞

Tables & Models

- ckeditor_assets
- comments
- dreams
- posts
- users

Table: Ckeditor_assets

data_file_name	data_content_name	data_file_size	assetable_id	assetable_type	type	width	height	created_at	updated_at
F1	C1	..	01	..	text	..	..	Jan 1st	Feb 1st
F2	C2	..	01	..	text	..	..	Feb 2nd	Mar 2nd

Model: Ckeditor_assets

```ruby
class Ckeditor::AttachmentFile < Ckeditor::Asset
  has_attached_file :data,
                    :url => "/ckeditor_assets/attachments/:id/:filename",
                    :path => ":rails_root/public/ckeditor_assets/attachments/:id/:filename"

  validates_attachment_presence :data
  validates_attachment_size :data, :less_than => 100.megabytes
  do_not_validate_attachment_file_type :data

  def url_thumb
    @url_thumb ||= Ckeditor::Utils.filethumb(filename)
  end
end
```

```ruby
class Ckeditor::Picture < Ckeditor::Asset
  has_attached_file :data,
                    :url  => "/ckeditor_assets/pictures/:id/:style_:basename.:extension",
                    :path => ":rails_root/public/ckeditor_assets/pictures/:id/:style_:basename.:extension",
                    :styles => { :content => '800>', :thumb => '118x100#' }

  validates_attachment_presence :data
  validates_attachment_size :data, :less_than => 2.megabytes
  validates_attachment_content_type :data, :content_type => /\Aimage/

  def url_content
    url(:content)
  end
end
```

Table: Comments

comment	Post_id	User_id	created_at	updated_at	image_file_name	image_content_type	image_type_size	image_ updated_ at
S1	1	1	Jan 1st	Feb 1st	n1	Jpg	..	Mar 1st
S2	2	2	Feb 2nd	Mar 2nd	N2	jpg	..	Apr 2nd

Model: Comments

```ruby
class Comment < ActiveRecord::Base
  belongs_to :post
  belongs_to :user
  #Set up image size in different conditions
  has_attached_file :image, styles: { small: "100x100>", medium: "300x300>", large: "600x600>", thumb: "100x100#"}
  validates_attachment_content_type :image, content_type:  /\Aimage\/.*\Z/
end
```

Table: Dreams

Description	start_time	created_at	updated_at	user_id
D1	Jan 1st	Jan 1st	Feb 1st	1
D2	Feb 2nd	Feb 2nd	Mar 2nd	2

Model: Dreams

```ruby
class Dream < ActiveRecord::Base
  belongs_to :user
  #Set up image size in different conditions
  def self.search(search)
    if search
      where(["description LIKE?","%#{search}%"])
    else
      all
    end
  end
  #Validation: description can't be blank
  validates :description, presence: true
end
```

Table: Posts

name	description	category	created_at	updated_at	user_id	image_ file_ name	image_ content_ type	image_ file_ size	image_ updated_ at
A	D1		Jan 1st	Feb 1st	1	n1	Jpg	..	Mar 1st
B	D2		Feb 2nd	Mar 2nd	2	n2	jpg	..	Apr 2nd

Model: Posts

```ruby
class Post < ActiveRecord::Base
  # Set rules for post controller
  #posts belongs to user
  belongs_to :user
  #Set up image size in different conditions
  has_attached_file :image, styles: { small: "100x100>", medium: "300x300>", large: "600x600>", thumb: "100x100#"}
  validates_attachment_content_type :image, content_type:  /\Aimage\/.*\Z/
  #User can have many comments
  has_many :comments
  #Validation: Name and description can't be blank
  validates :name, presence: true
  validates :description, presence: true
```

Table: Users

email	encrypted_ password	reset_ password_ token	reset_ password_ sent_at_	remember_ created_at	Sign_ in_count	current_ sign_in_at	last_ sign_in_at	current_ sign_in_ip	last_ sign_in_ip	created_at	updated_at
ma.710	123456	..	..	..	1	Feb 1st	Jan 1st	Ip2	Ip1	Jan 1st	Feb 1st
chua.40	122222	..	..	..	2	Mar 1st	Feb 1st	Ip4	Ip3	Feb 1st	Mar 1st

Model: Users

```ruby
class User < ActiveRecord::Base
  # Include default devise modules. Others available are:
  # :confirmable, :lockable, :timeoutable and :omniauthable
  devise :database_authenticatable, :registerable,
         :recoverable, :rememberable, :trackable, :validatable
  has_many :posts
  has_many :comments
  has_many :dreams
end
```

Posts Controller

```
=begin
  Edited by: Ma Yun
  Details: Edited so destory to give user a notice massage of sucess build.
=end
  # POST /posts
  # POST /posts.json
  def create
    @post = current_user.posts.build(post_params)

    respond_to do |format|
      if @post.save
        format.html { redirect_to @post, notice: 'Post was successfully created.' }
        format.json { render :show, status: :created, location: @post }
      else
        format.html { render :new }
        format.json { render json: @post.errors, status: :unprocessable_entity }
      end
    end
  end
```

Posts Controller

```ruby
def update
  respond_to do |format|
    if @post.update(post_params)
      format.html { redirect_to @post, notice: 'Post was successfully updated.' }
      format.json { render :show, status: :ok, location: @post }
    else
      format.html { render :edit }
      format.json { render json: @post.errors, status: :unprocessable_entity }
    end
  end
end
```

```ruby
def destroy
  @post.destroy
  respond_to do |format|
    format.html { redirect_to posts_url, notice: 'Post was successfully destroyed.' }
    format.json { head :no_content }
  end
end
```

Posts Controller

```ruby
class PostsController < ApplicationController
  before_action :set_post, only: [:show, :edit, :update, :destroy]
  before_action :authenticate_user!, except: [:index, :show]
  # GET /posts
  # GET /posts.json
  def index
    @posts = Post.all.order("created_at DESC")
  end

  # GET /posts/1
  # GET /posts/1.json
  def show
  end
```

Comments Controller

```ruby
class CommentsController < ApplicationController
 # Set rules for comment controller
  before_action :authenticate_user!
  def create
    @post = Post.find(params[:post_id])
    @comment = @post.comments.create(params[:comment].permit(:comment,:image))
    @comment.user_id = current_user.id if current_user
    @comment.save

    if @comment.save
      redirect_to post_path(@post)
    else
      render 'new'
    end
  end

  def destroy  ... end

  def update ... end
```

Dreams Controller

```ruby
class DreamsController < ApplicationController
  # Set rules for dream controller
  before_action :set_dream, only: [:show, :edit, :update, :destroy]
  before_action :authenticate_user!
  before_action :current_user
  # GET /dreams
  # GET /dreams.json
  # Only show current user's private dreambox and provide the search condition
  def index
    @dreams = current_user.dreams.all.order("created_at DESC").search(params[:search])
  end

  # GET /dreams/1
  # GET /dreams/1.json
  # Only show current user's private dreambox
  def show
    @dreams = current_user.dreams.all
  end

  # GET /dreams/new
  # let current user write down their dream
  def new
    @dream = current_user.dreams.build
  end
```

Page Controller

```
class PageController < ApplicationController
=begin
  Edited by: Levon Nie
  Details: Changing the autherntication of user
=end
  before_action :authenticate_user!, only:[:wall]
  before_action :authenticate_user!, only:[:wall]

  def home
  end

  def about
  end

  def contact
  end

end
```

Program Demo

Expectation in the future

1. Mobile device compatible
2. More media types - video, audio, GIF
3. Various login ways
4. “Likes” or “Unlikes” buttons