MVC

**User**

**Handled by devise gem**

**Authentication**

use devise (sign up and login)

in the terminal

gem 'devise'

bundle install

rails generate devise:install

**default\_url\_options, root\_url,flash messages**

1. Ensure you have defined default url options in your environments files. Here

is an example of default\_url\_options appropriate for a development environment

in config/environments/development.rb:

config.action\_mailer.default\_url\_options = { host: 'localhost', port: 3000 }

In production, :host should be set to the actual host of your application.

2. Ensure you have defined root\_url to \*something\* in your config/routes.rb.

For example:

root to: "pages#home"

3. Ensure you have flash messages in app/views/layouts/application.html.erb.

For example:

<p class="notice"><%= notice %></p>

<p class="alert"><%= alert %></p>

**Configure email**

# config/initializers/devise.rb

# ==> Mailer Configuration

# Configure the e-mail address which will be shown in Devise::Mailer,

# note that it will be overwritten if you use your own mailer class

# with default "from" parameter.

config.mailer\_sender = 'please-change-me-at-config-initializers-devise@example.com'

**Generate Model**

rails generate devise User

# invoke active\_record

# create db/migrate/TIMESTAMP\_devise\_create\_users.rb

# create app/models/user.rb

# insert app/models/user.rb

# route devise\_for :users

rails db:migrate

# app/models/user.rb

class User < ApplicationRecord

# Include default devise modules. Others available are:

# :confirmable, :lockable, :timeoutable, :trackable and :omniauthable

devise :database\_authenticatable, :registerable,

:recoverable, :rememberable, :validatable

End

rails routes

rails routes | grep new\_

**Helpers**

user\_signed\_in?

# => true / false

current\_user

# => User instance / nil

mkdir app/views/shared

curl https://raw.githubusercontent.com/lewagon/awesome-navbars/master/templates/\_navbar\_wagon.html.erb > app/views/shared/\_navbar.html.erb

curl https://raw.githubusercontent.com/lewagon/karr-images/master/white\_logo\_red\_circle.png > app/assets/images/logo.png

<!-- app/views/layouts/application.html.erb -->

<%= render 'shared/navbar' %>

**Alerts**

<!-- app/views/layouts/application.html.erb -->

<%= render 'shared/flashes' %>

<!-- app/views/shared/\_flashes.html.erb -->

<% if notice %>

<div class="alert alert-info alert-dismissible" role="alert">

<button type="button" class="close" data-dismiss="alert" aria-label="Close"><span aria-hidden="true">&times;</span></button>

<%= notice %>

</div>

<% end %>

<% if alert %>

<div class="alert alert-warning alert-dismissible" role="alert">

<button type="button" class="close" data-dismiss="alert" aria-label="Close"><span aria-hidden="true">&times;</span></button>

<%= alert %>

</div>

<% end %>

**Devise Views**

rails g devise:views

**controller**

# app/controllers/application\_controller.rb

class ApplicationController < ActionController::Base

before\_action :authenticate\_user!

end

class PagesController < ApplicationController

skip\_before\_action :authenticate\_user!, only: :home

def home

end

end (no login required on home page)

has\_many :orders

**Order**

rails g model Order state:string teddy\_sku:string amount:monetize checkout\_session\_id:string user:references teddy:references

Rails g model order total\_price:integer items\_list: string user:references; rails db:migrate

Rails g controller orders index show; rails db:migrate

Validates : items\_list, presence: true

Validates : total\_price, presence: true

has\_many :items

belongs\_to :user

**Inside orders\_controller.rb**

Def index

@orders = Order.all

End

Def show

@Order =Order.find(:id)

end

**Payments**

This has to do with stripe

Categories

Rails g model category name, rails db:migrate

has\_many :art\_subcategories

has\_many :home\_subcategories

has\_many :garden\_subcategories

Validates : name, presence: true, uniqueness: true

Rails g controller categories index show

**In categories\_controller.rb**

Def index

@categories = Category.all

End

Def show

@category =Category.find(:id)

@arts = Art\_subcategories.all

@homes = Home\_subcategories.all

@gardens = Garden\_subcategories.all

end

@arts, @homes and @gardens are included as the show of Categories is the index of home\_subcategories, garden\_subcategories and art\_subcategories. We will need them to work with as the view is shared.

**Subcategories**

**Art**

Rails g model Art\_sub\_cat name image\_url:string category:references, db:migrate

belongs\_to :category

Validates : name, presence: true, uniqueness: true

Validates : category\_name, presence: true

Rails s g contoller Art\_sub\_cats show, db:migrate

**Inside Art\_sub\_cats\_controller.rb**

Def show

@ art\_sub\_cat = Art\_sub\_cat.find(:id)

end

**Garden**

Rails g model garden\_sub\_cat name image\_url:string category:references, db:migrate

belongs\_to :category

Validates : name, presence: true, uniqueness: true

Validates : category\_name, presence: true

Rails s g contoller garden\_sub\_cats show, db:migrate

**Inside garden\_sub\_cats\_controller.rb**

Def show

@ garden\_sub\_cat = Garden\_sub\_cat.find(:id)

end

**Home**

Rails g model home\_sub\_cat name image\_url:string category:references, db:migrate

belongs\_to :category

Validates : name, presence: true, uniqueness: true

Validates : category\_name, presence: true

Rails s g contoller home\_sub\_cats show, db:migrate

**Inside home\_sub\_cats\_controller.rb**

Def show

@ home\_sub\_cat = Home\_sub\_cats.find(:id)

End

Now we need to do models for each subcategory. In garden: furniture, planters, trellis=fencing, bird feeders

Home: bed,bath,lounge, other

Art: toys,sculpture, busts, smoke stuff

NOTE each will need index, show, and possibly new,create,edit,update

**For each item/product**

Rails g model item-name name price:integer image\_url:string category:references … rails db:migrate

belongs\_to :order

Validates : name, presence: true, uniqueness: true

Validates : price, presence: true

Rails g controller item-name plural index show new create edit update destroy

rails db:migrate

**Inside the controller of each item**

def index

@items = Item.all

end

def show

@item = Item.find(params[:id])

end

def new

@item = item.new

end

def create

@item = Item.new(params[:item])

@item.save

end

def edit note we render the same form here as we do in the new

@item.find(params[:id])

end

def update

@item = Item.find(params[:id])

@item.update(item\_params)

redirect\_to item\_path(@item)

end

def destroy

@item = Item.find(params[:id])

@item.destroy

redirect\_to items\_path(@item)

end

Private

def item\_params

params.require(:item).permit(:name, :price, etc.)

end

**Components and Partials**

Partials to be rendered

\_form

Simple\_form\_for(@item)

<div class=”container”>

<div class=”inputs”>

<%= simple\_form\_for *@item* do |f| %>

<%= f.input :name %>

<%= f.input :price %>

<%= f.input :description %>

<%= f.input :image %>

<%= f.button :submit, class: "btn-primary" %>

<div class=”inputs”>

</div>

Only to be used by Daniel when adding an item to the store. Users have no access to the create a new item function.

\_navbar

\_footer

Uses font-awesome

<div class="footer">

<div class="footer-links">

<a href="#"><i class="fab fa-instagram"></i></a>

<a href="#"><i class="fab fa-facebook"></i></a>

<a href="#"><i class="fab fa-twitter"></i></a>

<a href="#"><i class="fab fa-linkedin"></i></a>

</div>

<div class="footer-copyright">

This footer is made with <i class="fas fa-heart"></i> at Le Wagon

</div>

</div>

.footer {

background: #F4F4F4;

display: flex;

align-items: center;

justify-content: space-between;

height: 100px;

padding: 0px 50px;

color: rgba(0,0,0,0.3);

}

.footer-links {

display: flex;

align-items: center;

}

.footer-links a {

color: black;

opacity: 0.15;

text-decoration: none;

font-size: 24px;

padding: 0px 10px;

}

.footer-links a:hover {

opacity: 1;

}

.footer .fa-heart {

color: #D23333;

}

\_category\_card

<div class="card-category" style="background-image: linear-gradient(rgba(0,0,0,0.3), rgba(0,0,0,0.3)), url(https://raw.githubusercontent.com/lewagon/fullstack-images/master/uikit/breakfast.jpg)">

Breakfast

</div>

.card-category {

background-size: cover;

background-position: center;

height: 180px;

display: flex;

justify-content: center;

align-items: center;

color: white;

font-size: 24px;

font-weight: bold;

text-shadow: 1px 1px 3px rgba(0,0,0,0.2);

border-radius: 5px;

box-shadow: 0 0 15px rgba(0,0,0,0.2);

\_product\_card

<div class="card-trip">

<img src="https://raw.githubusercontent.com/lewagon/fullstack-images/master/uikit/greece.jpg" />

<div class="card-trip-infos">

<div>

<h2>Title here</h2>

<p>Short description here!</p>

</div>

<h2 class="card-trip-pricing">£99.99</h2>

<img src="https://kitt.lewagon.com/placeholder/users/krokrob" class="card-trip-user avatar-bordered" />

</div>

</div>

.card-trip {

overflow: hidden;

background: white;

box-shadow: 0 0 15px rgba(0,0,0,0.2);

}

.card-trip > img {

height: 200px;

width: 100%;

object-fit: cover;

}

.card-trip h2 {

font-size: 16px;

font-weight: bold;

margin: 0;

}

.card-trip p {

font-size: 12px;

opacity: .7;

margin: 0;

}

.card-trip .card-trip-infos {

padding: 16px;

display: flex;

justify-content: space-between;

align-items: flex-end;

position: relative;

}

.card-trip-infos .card-trip-user {

position: absolute;

right: 16px;

top: -20px;

width: 40px;

}

\_button

3 examples of buttons.

<%= link\_to "Write a story", "#", class: "btn btn-ghost" %>

<%= link\_to "Book now", "#", class: "btn btn-flat" %>

<%= link\_to "Start now", "#", class: "btn btn-gradient" %>

.btn-ghost {

color: #4A4A4A;

border: 1px solid #4A4A4A;

padding: 8px 24px;

border-radius: 50px;

font-weight: lighter;

opacity: 0.6;

transition: opacity 0.3s ease;

}

.btn-ghost:hover {

opacity: 1;

}

.btn-flat {

color: white;

padding: 8px 24px;

border-radius: 4px;

background: #1EDD88;

transition: background 0.3s ease;

}

.btn-flat:hover {

background: #1BCB7F;

color: white;

}

.btn-gradient {

color: white;

padding: 8px 24px;

border-radius: 4px;

font-weight: bold;

background: linear-gradient(#167FFB, #0F60C4);

transition: background 0.3s ease;

border: 1px solid #0F60C4;

}

.btn-gradient:hover {

background: linear-gradient(#147EFF, #0F67DA);

color: white;

}

Banner

<div class="banner" style="background-image: linear-gradient(rgba(0,0,0,0.4),rgba(0,0,0,0.4)), url(https://raw.githubusercontent.com/lewagon/fullstack-images/master/uikit/background.png);">

<div class="container">

<h1>Eco-conscious Crafts. <strong> Bringing upcycled home accessories </strong>to <strong> those wanting a greener lifestyle </strong>!</h1>

<p> Our stylish, eco-friendly products blend craftsmanship and art, making for a truly unique range.</p>

<a class="btn btn-flat" href="#">Browse our products</a>

<%= link\_to "Browse Products", "categories\_path", class: "btn btn-gradient" %>

</div>

</div>

.banner {

background-size: cover;

background-position: center;

padding: 150px 0;

}

.banner h1 {

margin: 0;

color: white;

text-shadow: 1px 1px 3px rgba(0,0,0,0.2);

font-size: 32px;

font-weight: bold;

}

.banner p {

font-size: 20px;

color: white;

opacity: .7;

text-shadow: 1px 1px 3px rgba(0,0,0,0.2);

}

**Views**

**Sidenote: navbar and footer will be in application.html.erb**

[Home.html.erb](http://Home.html.erb)

<div class=”container”

Render ‘banner’

<%= link\_to "Browse Products", "categories\_path", class: "btn btn-gradient" %>

**Contact.html.erb**

<div class=container>

<img src=”#”>

<div class=”contact-info”>

<h3> contact information <h3>

<p> <strong>cell-number :0718877764</strong> <p>

<p><strong> email :daniel@gmail.com </strong> <p>

</div>

<div class="social-media-links">

<h3> Social Media</h3>

<a href="linktoinstagram"><i class="fab fa-instagram"></i></a>

<a href=" linktofacebook "><i class="fab fa-facebook"></i></a>

<a href=" linktotwitter "><i class="fab fa-twitter"></i></a>

<a href=" linktolinkedin "><i class="fab fa-linkedin"></i></a>

</div>

</div>

**About.html.erb**

<div class=container>

<img src=”#”>

<div class=”mission-statement”>

<h3> Our Story <h3>

<p> mission statement as written by dan will go here<p>

</div>

<div class=”about-outreach”>

<p>about our social outreach section as written by dan <p>

<%= link\_to “social outreach programs”, outreach\_path class: "btn btn-ghost".

</div>

</div>

**Generic item index.html.erb**

<div class=”cards”>

<% @items.each do |item| %>

<%=link\_to item\_path(item)do %>

<div class="card-trip/item-card">

<%=item.image %>

<div class="card-trip-infos">

<div>

<h2><%=item.name%></h2>

<p><%=item.description%><p>

</div>

<h2 class="card-trip-pricing">item.price</h2>

</div>

</div>

<%end%>

</div>

CSS grid for displaying all the cards

.cards {

display: grid;

grid-template-columns: 1fr 1fr 1fr;

grid-gap: 16px;

}

@media(max-width: 992px) {

.cards {

grid-template-columns: 1fr 1fr;

}

}

@media(max-width: 768px) {

.cards {

grid-template-columns: 1fr;

}

}

**Generic item show.html.erb**

<div class=”sales-card”>

<div class=”row”>

<div class=” col-12”>

<div class="card-trip/item-card">

<%=@item.image %>

<div class="card-trip-infos">

<div>

<h2><%=@item.name%></h2>

<p><%=@item.detailed\_description%><p>

</div>

<h2 class="card-trip-pricing">@item.price</h2>

</div>

</div>

<div class=sales-button btn-btn-ghost>

link\_to “purchase this item/add to cart”, payment\_path

</div>

</div>

</div>

</div>

Generic Item index is the same as generic subcategory show eg. Toysitems index is the art\_sub\_category toys show

**Category index.html.erb**

<div class=”cards”>

<% @categories.each do |category| %>

<%=link\_to category\_path(category/@category)do %>

<div class="card-trip/item-card">

<%=category.image %>

<div class="card-trip-infos">

<div>

<h2><%=category.name%></h2>

</div>

</div>

</div>

<%end%>

</div>

Alternatively, build the page using bootstrap grid and put the categories in their own columns individually.

CSS grid for displaying all the cards

.cards {

display: grid;

grid-template-columns: 1fr 1fr 1fr;

grid-gap: 16px;

}

@media(max-width: 992px) {

.cards {

grid-template-columns: 1fr 1fr;

}

}

@media(max-width: 768px) {

.cards {

grid-template-columns: 1fr;

**Category\_show.html.erb**

this is the same as the individual subcategory index.

Display all cards for subcategories where category name == name of subcategory’s category

**Artsubcategory.index.html.erb**

<div class=”cards”>

<% @artcategories.each do |category| %>

<%=link\_to artcategory\_path(artcategory/@category)do %>

<div class="card-trip/item-card">

<%= artcategory.image %>

<div class="card-trip-infos">

<div>

<h2><%=artcategory.name%></h2>

</div>

</div>

</div>

<%end%>

</div>

Alternatively, build the page using bootstrap grid and put the categories in their own columns individually.

CSS grid for displaying all the cards

.cards {

display: grid;

grid-template-columns: 1fr 1fr 1fr;

grid-gap: 16px;

}

@media(max-width: 992px) {

.cards {

grid-template-columns: 1fr 1fr;

}

}

@media(max-width: 768px) {

.cards {

grid-template-columns: 1fr;

**Artcategories(also home, garden) show is the same as items index**

<div class=”sales-card”>

<div class=”row”>

<div class=” col-12”>

<div class="card-trip/item-card">

<%=item.image %>

<div class="card-trip-infos">

<div>

<h2><%=item.name%></h2>

<p><%=item.detailed\_description%><p>

</div>

<h2 class="card-trip-pricing">item.price</h2>

</div>

</div>

<div class=sales-button btn-btn-ghost>

link\_to “purchase this item/add to cart”, payment\_path

</div>

</div>

</div>

</div>

**Stripe and payment**

rails new \

--webpack \

--database postgresql \

-m https://raw.githubusercontent.com/lewagon/rails-templates/master/devise.rb \

teddies\_shop

cd teddies\_shop

rails g model Category name:string

rails g model Teddy sku:string name:string category:references photo\_url:string

rails db:migrate

go to seeds.rb, create some seeds then rails db:seed

rails g controller teddies

# config/routes.rb

devise\_for :users

root 'teddies#index'

resources :teddies, only: [:index, :show]

in teddies controller

skip\_before\_action :authenticate\_user!

def index

@teddies = Teddy.all

end

def show

@teddy = Teddy.find(params[:id])

End

gem 'money-rails'

bundle

rails g money\_rails:initializer

Set default currency

# config/initializers/money.rb

# https://github.com/RubyMoney/money#deprecation

Money.locale\_backend = :currency

MoneyRails.configure do |config|

config.default\_currency = :eur # or :zar, :usd, etc.

# [...]

end

### ADD PRICE TO TEDDIES

rails g migration AddPriceToTeddies

Edit migration

add\_monetize :teddies, :price, currency: { present: false }

rails db:migrate

Enable monetize

# app/models/teddy.rb

class Teddy < ApplicationRecord

# [...]

monetize :price\_cents

end

**then in seeds rb, set the price. Rails db:seed**

**displaying price, just add in the following code or similar code.**

<!-- app/views/teddies/index.html.erb -->

<p>Amount: <%= humanized\_money\_with\_symbol(teddy.price) %></p>

<!-- app/views/teddies/show.html.erb -->

<p>Amount: <%= humanized\_money\_with\_symbol(@teddy.price) %></p>

Now go to stripe.com, create an account and have a name.

# Gemfile

gem 'stripe'

bundle install – in terminal

### LETS SECURE OUR STRIPE CREDENTIALS!

Add stripe initializer

# config/initializers/stripe.rb

Rails.configuration.stripe = {

publishable\_key: ENV['STRIPE\_PUBLISHABLE\_KEY'],

secret\_key: ENV['STRIPE\_SECRET\_KEY']

}

Stripe.api\_key = Rails.configuration.stripe[:secret\_key]

Add your keys to .env file.

# .env

STRIPE\_PUBLISHABLE\_KEY=pk\_\*\*H

STRIPE\_SECRET\_KEY=sk\_\*\*f

We get our keys from stripe when we create an account.

rails g model Order state:string teddy\_sku:string amount:monetize checkout\_session\_id:string user:references teddy:references

Edit migration to remove currency

t.monetize :amount, currency: { present: false }

Don't forget to run the migration

rails db:migrate

Enable monetize

# app/models/order.rb

monetize :amount\_cents

Add relation to User model

# app/models/user.rb

has\_many :orders

### ROUTES & CONTROLLER

Add orders controller

rails g controller orders

Edit routes

# config/routes.rb

resources :orders, only: [:show, :create]

### EDIT TEDDY PAGE

Add a form to create an Order

# app/views/teddies/show.html.erb

<%= form\_tag orders\_path do %>

<%= hidden\_field\_tag 'teddy\_id', @teddy.id %>

<%= submit\_tag 'Purchase', class: 'btn btn-primary' %>

<% end %>

### EDIT ORDERS CONTROLLER

# app/controllers/orders\_controller.rb

def create

teddy = Teddy.find(params[:teddy\_id])

order = Order.create!(teddy: teddy, teddy\_sku: teddy.sku, amount: teddy.price, state: 'pending', user: current\_user)

session = Stripe::Checkout::Session.create(

payment\_method\_types: ['card'],

line\_items: [{

name: teddy.sku,

images: [teddy.photo\_url],

amount: teddy.price\_cents,

currency: 'eur',

quantity: 1

}],

success\_url: order\_url(order),

cancel\_url: order\_url(order)

)

order.update(checkout\_session\_id: session.id)

redirect\_to new\_order\_payment\_path(order)

end

## PAYMENTS

### ROUTES & CONTROLLER

### rails g controller payments

Edit routes

# config/routes.rb

resources :orders, only: [:show, :create] do

resources :payments, only: :new

end

### EDIT PAYMENT CONTROLLER

# app/controllers/payments\_controller.rb

class PaymentsController < ApplicationController

def new

@order = current\_user.orders.where(state: 'pending').find(params[:order\_id])

end

end

**PAYMENT FORM**

<!-- app/views/payments/new.html.erb -->

<div class="container">

<div class="row py-4">

<div class="col-md-3"><%= image\_tag @order.teddy.photo\_url, width: '100%' %></div>

<div class="col">

<h1>Checkout summary</h1>

<p>Purchase of <strong><%= @order.teddy.name %></strong> for <strong><%= humanized\_money\_with\_symbol @order.amount %></strong></p>

<button id="pay" class="btn btn-primary">Pay</button>

<!-- # Commented for the lecture, remove comments.

<script src="https://js.stripe.com/v3/"></script>

<script>

const paymentButton = document.getElementById('pay');

paymentButton.addEventListener('click', () => {

const stripe = Stripe('<%= ENV['STRIPE\_PUBLISHABLE\_KEY'] %>');

stripe.redirectToCheckout({

sessionId: '<%= @order.checkout\_session\_id %>'

});

});

</script> -->

</div>

</div>

</div>

### EDIT ORDERS CONTROLLER

# app/controllers/orders\_controller.rb

def show

@order = current\_user.orders.find(params[:id])

end

### EDIT ORDER PAGE

<!-- app/views/orders/show.html.erb -->

<div class="container">

<div class="row py-4">

<div class="col-md-3"><%= image\_tag @order.teddy.photo\_url, width: '100%' %></div>

<div class="col">

<h1>Order #<%= @order.id %> <small><span class="badge badge-primary"><%= @order.state %></span></small></h1>

<ul class="list-unstyled">

<li><strong>ISSUED ON</strong> <%= @order.created\_at.to\_date %></li>

<li><strong>TEDDY</strong> <%= @order.teddy.name %> [<%= @order.teddy\_sku %>]</li>

<li><strong>TOTAL</strong> <%= humanized\_money\_with\_symbol @order.amount %></li>

</li>

</ul>

<p>Estimated delivery in <strong>2 business days</strong></p>

</div>

</div>

</div>

## HOW TO UPDATE THE ORDER?

Right now, the order is still pending 🤔

## USING [STRIPE WEBHOOKS](https://stripe.com/docs/webhooks)!

Let Stripe tell us when the payment is done.

## ADD [STRIPE EVENT](https://github.com/integrallis/stripe_event) GEM

# Gemfile

gem 'stripe\_event'

bundle install

## CREATE A WEBHOOK ENDPOINT

# config/routes.rb

mount StripeEvent::Engine, at: '/stripe-webhooks'

Go on your Stripe dashboard to [create a webhook](https://dashboard.stripe.com/test/webhooks)

And add your webhook URL endpoint: <HOST URL>/stripe-webhooks

(You can use [ngrok](https://ngrok.com/) to test it locally)

## WEBHOOK SECRET KEY

Add the Stripe webhook 'Signing secret' in your .env file...

STRIPE\_WEBHOOK\_SECRET\_KEY=whsec\_\*\*\*\*

... and in your stripe.rb initializer.

Rails.configuration.stripe = {

# ...

signing\_secret: ENV['STRIPE\_WEBHOOK\_SECRET\_KEY']

}

# ...

StripeEvent.signing\_secret = Rails.configuration.stripe[:signing\_secret]

# config/initializers/stripe.rb

StripeEvent.configure do |events|

events.subscribe 'checkout.session.completed', StripeCheckoutSessionService.new

end

# app/services/stripe\_checkout\_session\_service.rb

class StripeCheckoutSessionService

def call(event)

order = Order.find\_by(checkout\_session\_id: event.data.object.id)

order.update(state: 'paid')

end

end

## TESTING

# Standard credit card

4242 4242 4242 4242

# Credit card with 3D Secure 2

4000 0000 0000 3220

More cards: <https://stripe.com/docs/testing#cards>