Project: OneCarBrakePads.com

Purpose

This is a small project to familiarise yourself with Ruby, Rails and some of the gems that are used at ClickMechanic.

Outline

For this project you're going to make a small ecommerce site selling car parts, similar to existing websites such as http://www.gsfcarparts.com/

To save time, use one specific model of car (eg: a Ford Fiesta MK5 Hatchback 1.6 TDCi) and one part type (eg: brake pads) that you want the site will sell.

User interaction

The typical flow through this site should be:

1. User selects a car using one or more select boxes (remember one model is fine). Submitting the form returns a list of possible part types is displayed (again just one is fine).
2. User selects a part type. A list of parts of the correct type is shown.
3. User select a part from the list, which takes them to the checkout (forget about shopping basket functionality).
4. If the user is not signed in, the checkout redirects to a sign in page (with a link to create an account).
5. On the first checkout page, the user must enter a name and address. Submitting the form should validate that these fields are present and then redirect to a payment page.
6. On the payment page, the user enters their credit card and security code (CVC). Allow submission of this form to always succeed without validating the input. On a successful submission, send an email is sent to the user and a basic confirmation page is shown.

Gems to use

If you have already installed Rails 4.2 you'll need to do 'rails new _4.1.12_ my_car_parts' to create a 4.1.12 project.

rails ~> 4.1.12 – Using the bundled defaults of Coffeescript, jQuery and SASS.

pg – Replaces *sqlite3* in default Gemfile. Not essential but we use postgresql at ClickMechanic

bootstrap-sass – CSS framework

simple_form – Improved form generator that will wrap fields in bootstrap containers and apply bootstrap styles (may need to add *config.default_wrapper* = *:bootstrap* to initializer)

devise – User authentication.

These gems are all well documented online, so they shouldn't be too hard to use

Models / schema

The basic models you will need are:

User
Stores email and encrypted password. Devise includes a good generator for this – see the Devise docs

CarModel
Stores a display name for the car (Make/Model/Year/Engine)

PartType
Stores a display name and any other data you'd like to present on the list of parts.

Order
References a User. Stores an address, and the order total.

OrderPart
References a PartType and an Order, and stores a price.

Notes

Don't spend time styling all of the pages. Try to include one page that looks as good as you can get with the Bootstrap CSS framework, and leave the others unstyled.