Build a Shopping Cart

https://github.com/mschwarzmueller/laravel-shopping-cart-tutorial

Intro & Setup

1. 建立專案
>> php artisan create-project laravel/laravel shopping-cart

2. 建資料庫，設定.env檔案

3. 在resources底下建立layouts, partials, shop, user四個資料夾

4. 在layouts資料夾底下建立master.blade.php
   - 新增程式碼，加上bootstrap, jquery…
   - 把layouts, partials, shop, user四個資料夾放到views資料夾底下

5. shop資料夾底下建立index.blade.php
   - 新增程式碼
   - 新增routes

http://localhost:8000/

安裝自動產生bootstrap程式碼
>> apm install atom-bootstrap3
https://atom.io/packages/atom-bootstrap3
atom-beautify 調整程式碼編排
>> apm install atom-beautify

Product Index View

1. 在partials資料夾底下新建header.blade.php
http://getbootstrap.com/components/
copy “Default navbar” code to header.blade.php

2. 製作header
   - 到master.blade.php include hearder @include('partials.header')
   - 把header.blade.php 清除多餘程式碼

http://localhost:8000/
   - http://fontawesome.io/get-started/ 申請CDN link 加到master.blade.php
   - http://fontawesome.io/icon/user/ 搜尋icon(user, shopping cart)，加到 header.blade.php

Build a Shopping Cart

https://github.com/mschwarzmueller/laravel-shopping-cart-tutorial

Intro & Setup

1. 建立專案
>> php artisan create-project laravel/laravel shopping-cart

2. 建資料庫、設定.env檔案

3. 在resources底下建立layouts, partials, shop, user四個資料夾

4. 在layouts資料夾底下建立master.blade.php
   - 新增程式碼、加上bootstrap, jquery...
   - 把layouts, partials, shop, user四個資料夾放到views資料夾底下

5. shop資料夾底下建立index.blade.php
   - 新增程式碼
   - 新增routes

http://localhost:8000/

安裝自動產生bootstrap程式碼
>> apm install atom-bootstrap3
https://atom.io/packages/atom-bootstrap3
atom-beautify 調整程式碼編排
>> apm install atom-beautify

Product Index View

1. 在partials資料夾底下新建header.blade.php
http://getbootstrap.com/components/
copy "Default navbar" code to header.blade.php

2. 製作header
   - 到master.blade.php include hearder @include('partials.header')
   - 把header.blade.php 清除多餘程式碼

http://localhost:8000/

- http://fontawesome.io/get-started/ 申請CDN link 加到master.blade.php
- http://fontawesome.io/icon/user/ 搜尋icon(user, shopping cart)、加到 header.blade.php

User Sign Up

1. 建立User Migration
   >> php artisan make:migration create_users_table
   - 寫user migration file
   - 修改User.php (model)

2. Sign Up (view)
   - 在views\user底下建立signup.blade.php, signin.blade.php, profile.blade.php
   - 在signup.blade.php 建立UI (view)
   - add new routes

3. Sign Up (controller)
   - >> php artisan make:controller UserController
   - 在UserController.php getSignup(), postSignup() wirte code

4. 微調
   - >> php artisan migrate
   - header.blade.php 調整登入登出的選單

http://localhost:8000/signup

User Sign In

------sign in------
1. 複製signin.blade.php code to signup.blade.php, 修改code (view)

2. add new routes

3. UserController.php getSignin(), postSignin() write code

4. 微調header.blade.php的signin link
------user profile------
5. 複製signin.blade.php code to profile.blade.php, 修改code (view)

6. add new routes

7. UserController.php getProfile() write code
http://localhost:8000/signin

Cart Views

1. 在header.blade.php新增link，讓shopping cart可以顯示目前有幾個 & add new link

2. add new route

3. ProductController.php getCart() write code

4. 在views/shop底下新增shopping-cart.blade.php, write code
http://localhost:8000/shopping-cart

Stripe Payments View

1. shopping-cart.blade.php 新增Checkout的link

2. add new routes

3. ProductController.php getCheckout() write code

4. 在views/shop底下新增checkout.blade.php, write code
   - https://stripe.com/
   - https://stripe.com/docs

用來做信用卡付款的API

5. add new routes
http://localhost:8000/checkout

Stripe Credit Card Verification

1. https://dashboard.stripe.com/account/apikeys
Test Secret Key: sk_test_fMPXxzuYp7KofG5ZTXPOEU73
   - https://stripe.com/docs/libraries PHP套件加到composer.json

```
{
  "require": {
    "stripe/stripe-php": "4.*"
  }
}
```

   - composer update

2. https://stripe.com/docs/stripe.js
   - 把Stripe.js CDN加到checkout.blade.php
   - 在public/src/底下新增js資料夾，在新增checkout.js write code
     - Stripe.setPublishableKey('pk_test_zrmfU1QCrHGivEoDbD6dGDWY');

3. checkout.blade.php 新增code

4. 回到checkout.js新增code

Finishing Touches(Reupload)

1. Cart.php reduceByOne() write code

2. ProductController.php getReductByOne() write code

3. add new routes

4. shop/shopping-cart.blade.php add link
http://localhost:8000/shopping-cart

5. Cart.php reduceByOne() write code

---

6. Cart.php removeItem() write code

7. ProductController.php getRemoveItem() write code

8. add new routes

9. shop/shopping-cart.blade.php add link
http://localhost:8000/shopping-cart

---

10. ProductController.php getReductByOne(), getRemoveItem() add code
判斷商品數為0時，會自動顯示No items，而不會跑出0元和checkout按鈕