GROUP PROJECT FRONT SHEET

Qualification	**BTEC Level 5 HND Diploma in Computing**		
Unit number and title	WEBG301 - Project Web		
Submission date		**Date Received 1st submission**	
Re-submission Date		**Date Received 2nd submission**	
Student Name		**Student ID**	
Class		**Assessor name**	Nguyen Dinh Tran Long

Student declaration

I certify that the assignment submission is entirely my own work and I fully understand the consequences of plagiarism. I understand that making a false declaration is a form of malpractice.

Student's signature	

Grade

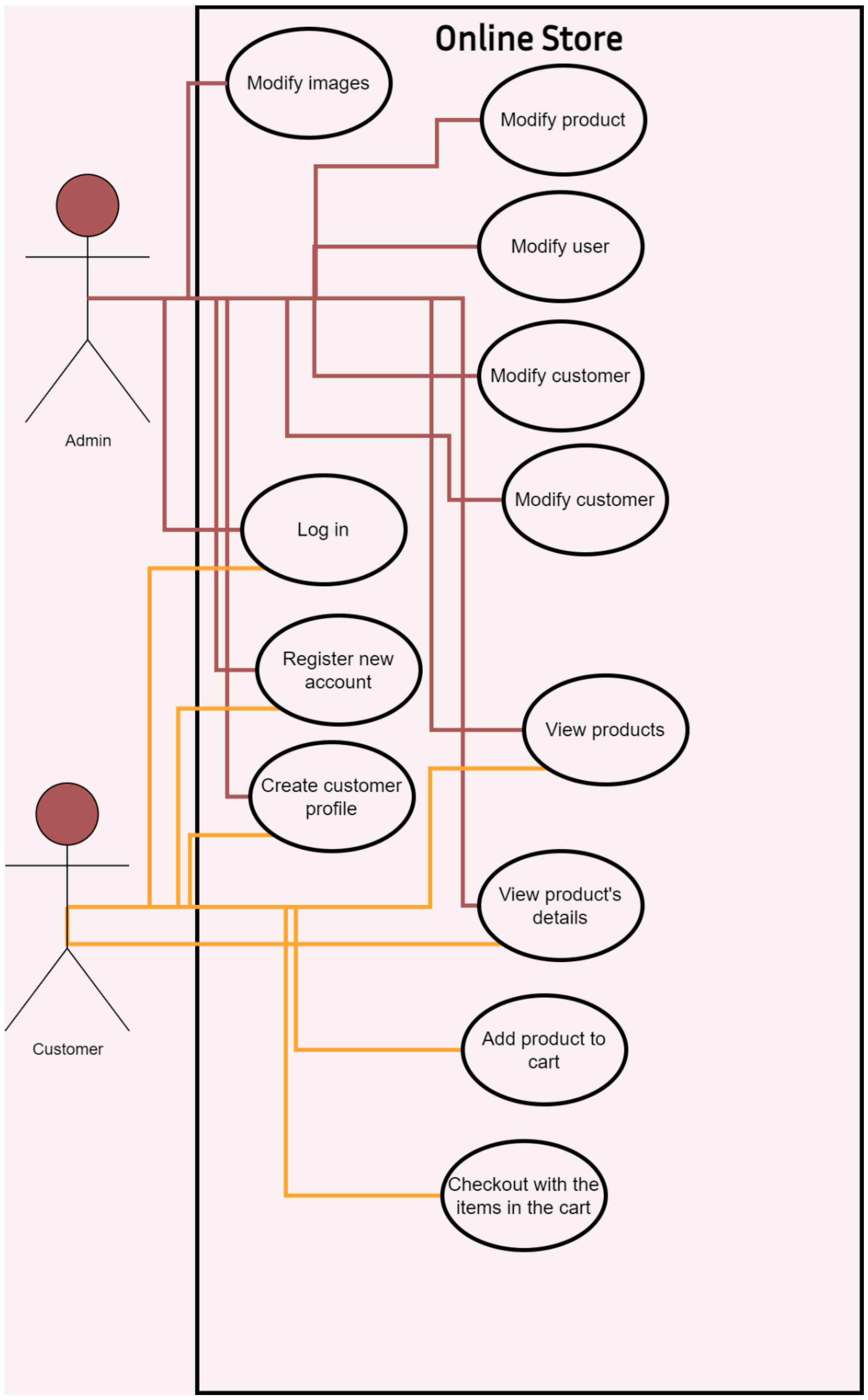
❒Summative Feedback:		❒Resubmission Feedback:
Grade:	Assessor Signature:	Date:
Signature & Date:		

Part 1 – Users' requirements

I. User Stories template

No	User story	Functional/Nonfunctional
1	As a customer, I want to be able to create an account so I can put products into a cart and checkout with them	Functional
2	As a customer, I want to be directed to the login page if I happen to click on the "Cart" button when I am not logged in	Functional
3	As a customer, when I click on the "Checkout" button, I want a receipt to be generated based on the items I have added into my cart and the information on my account	Functional
4	As a customer, when I hover over the picture of a product, I want brief information about the product to be displayed	Functional
5	As a customer, when I click on a product, I want to be taken to a detail page that presents all of the product's information along with a "Add to cart" button	Functional
6	As the operation manager, I want the website to have aesthetics that suit the store's theme, which is artistic hand-made crafts	Non-Functional
7	As a system admin, I want additional links to appear on the navigation bar when an admin is logged into the website	Functional
8	As a system admin, I want to be able to perform CRUD on different information of the database without having to directly interact with the database	Functional
9	As the operation manager, I want the system to encrypt the login information when a new user is created in order to protect the information of everyone	Non-Functional

II. Use case diagram

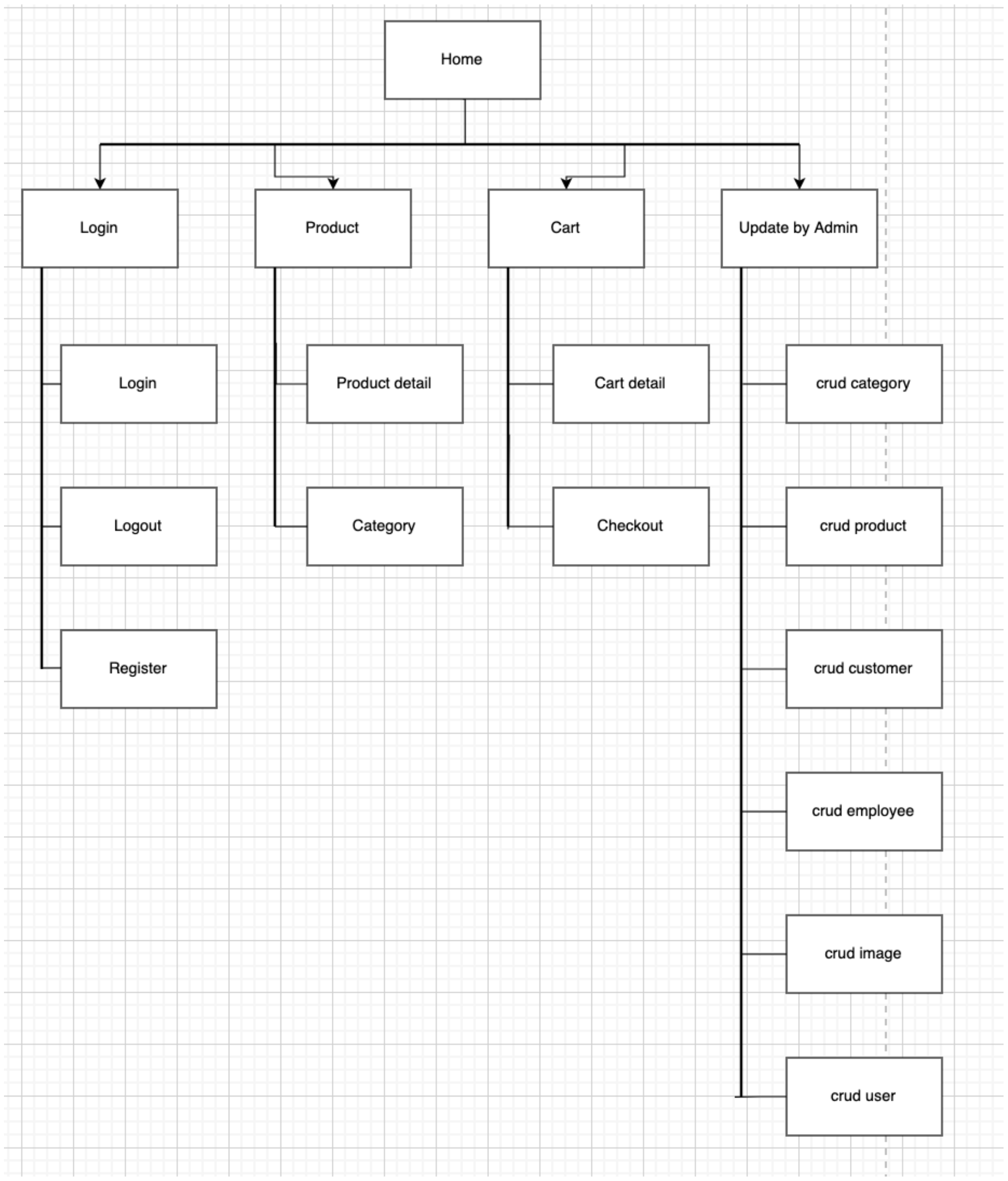

Online Store
Modify images
Modify product
Modify user
Modify customer
Modify customer
Admin
Log in
Register new account
Create customer profile
View products
View product's details
Add product to cart
Checkout with the items in the cart
Customer

Part 2 – System Design

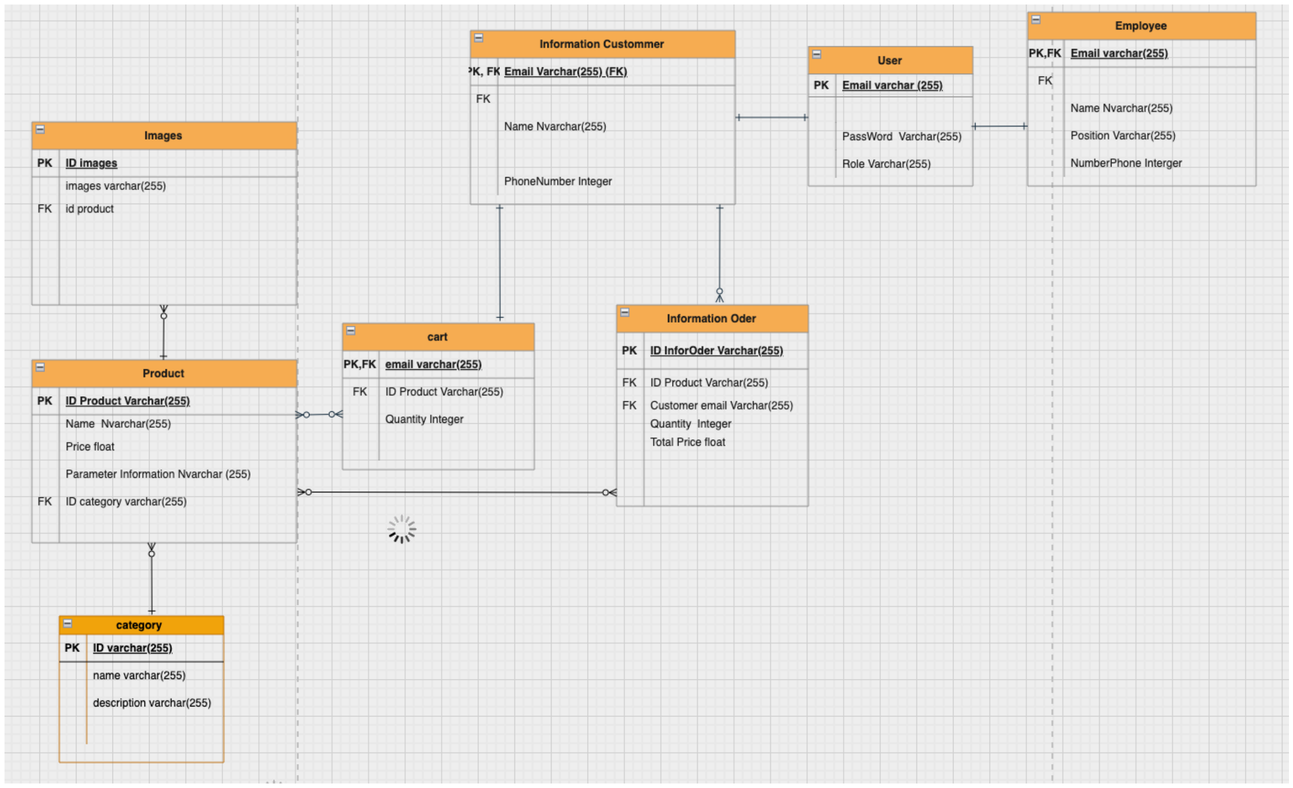
I. Site map

The project sitemap starts from the home page (Home), which is the main website for users to access and manipulate functionality. On the home page, users will branch out to other functional websites including: Login, Product, Cart, and the admin-specific function page, which is the Update by admin page.

- Login page: This is the page that helps users to login. There will be two types of decentralized accounts for User and Admin. In login there will be login, logout and register branches for newly registered users.
- Product page: contains information, general images of all products of the website. This page will guide users to access the Product detail which contains detailed information of each product. There will also be a Category so that users can filter products by the corresponding category.
- Cart page: When the user chooses to add to the cart, the product will be included in the Cart detail where the product information the user chooses to buy. Inside there will be a Checkout page to output the invoice of the order.
- Update by Admin page: special page for Admin users. The page will branch into many different functional pages that have crud functions for each entity of the project such as: product, user, employee, etc.



II. Entity Relationship Diagram

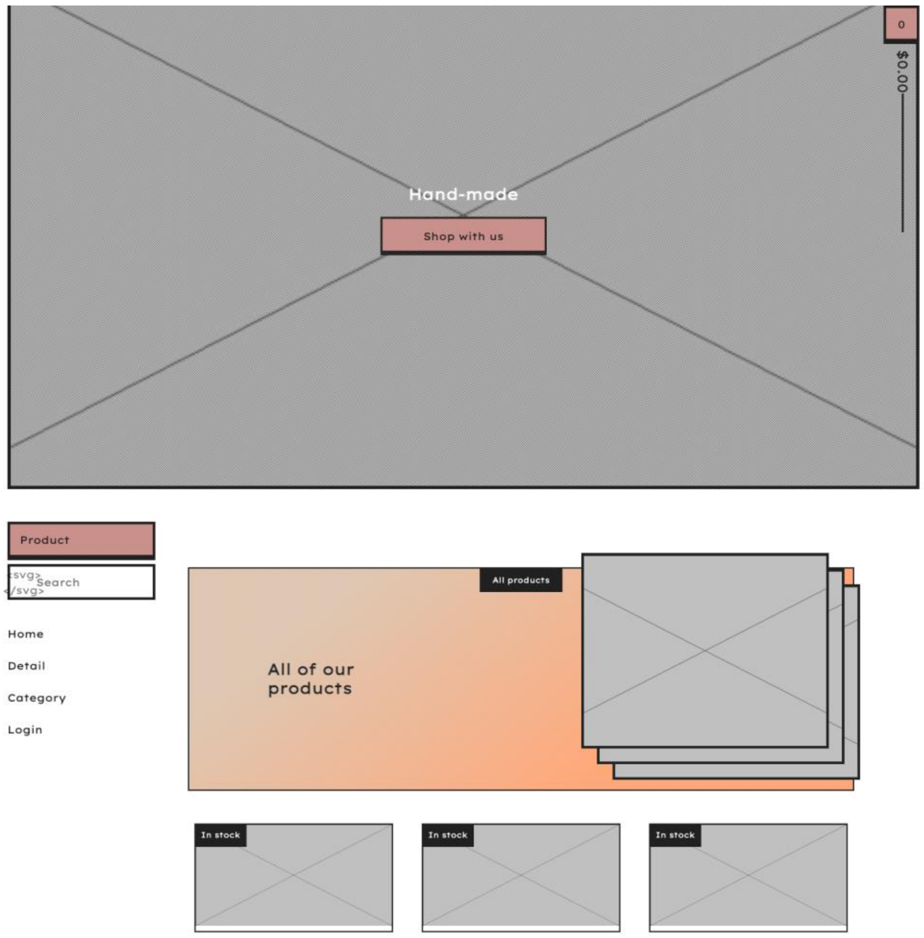


Relation 1	Relation 2	Type of relationship	Reason
User	Employee	One to One	One user account for one employee and one employee has only one user account
User	Customer	One to One	One user account for one customer and one customer has only one user account
Customer	Order Infomation	One to many	A customer has many different Orders and an Order of only one Customer
Customer	Cart	One to One	One customer for one cart and one cart created by only one Customer
Product	Cart	Many to many	One product is in multiple carts and one cart can contain many products
Product	Order Infomation	Many to many	One product is in multiple Order Infomation and one Order Infomation can contain many products.

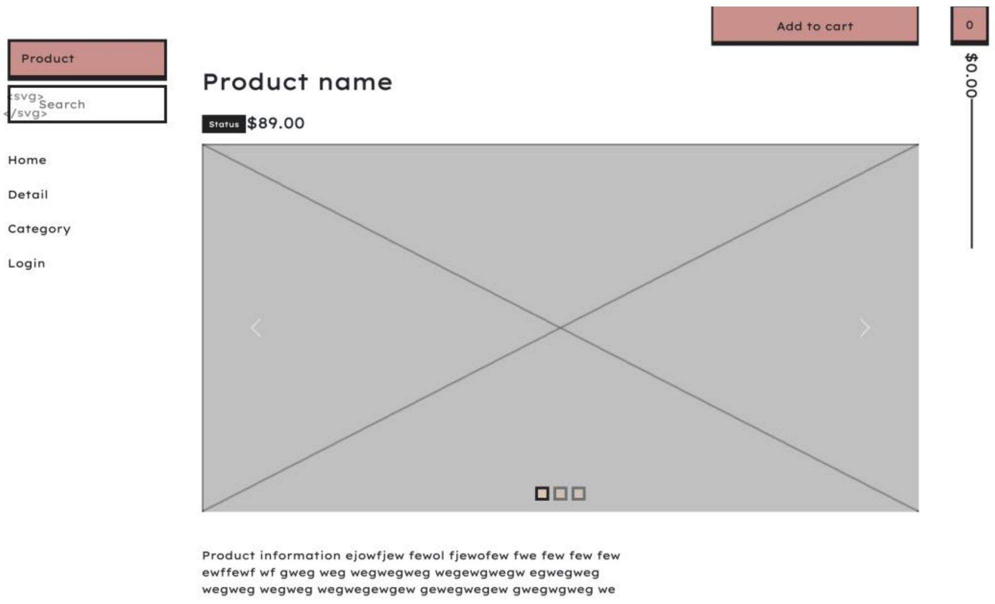
Product	Category	Many to One	A Product has only one category, but a category can contain many products
Product	Image	One to Many	A product has many photos, but one photo is only of that product

III. Wireframes

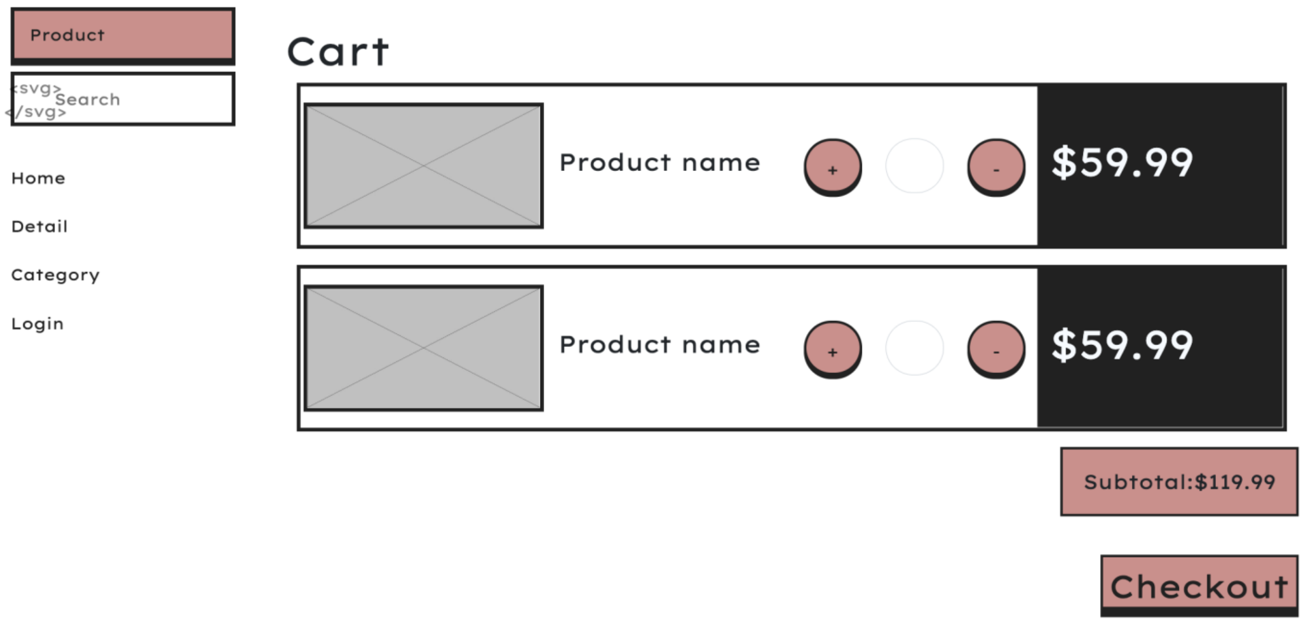
Home Page:



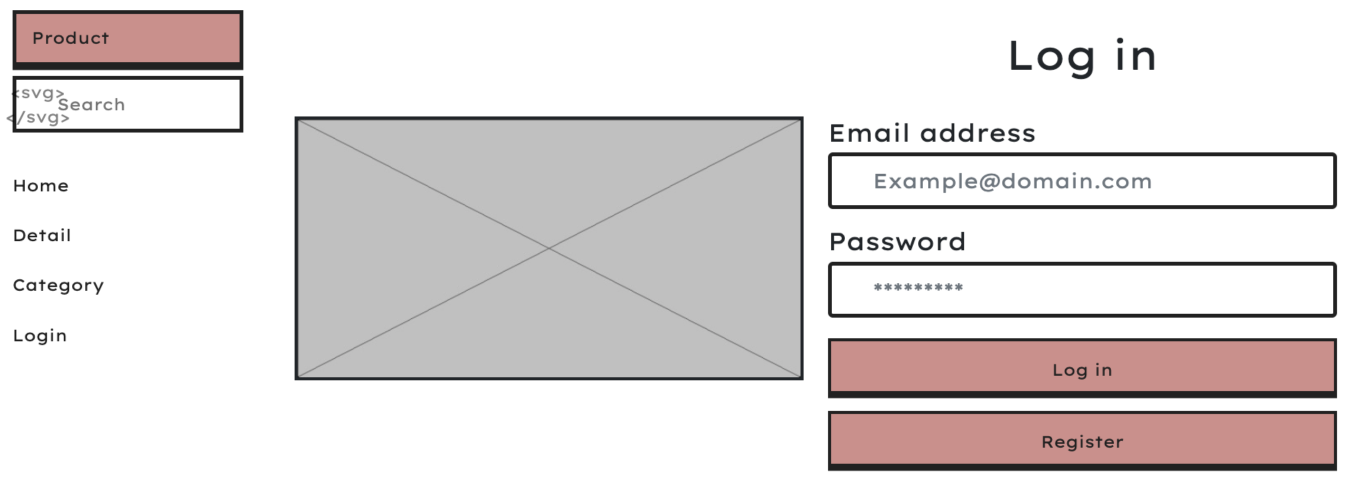
Product detail:


Add to cart
0
$0.00
Product
<svg>Search</svg>
Product name
Status $89.00
Home
Detail
Category
Login
Product information ejowfjew fewol fjewofew fwe few few few ewffewf wf gweg weg wegwegweg wegewgwegw egwegweg wegweg wegweg wegwegewgew gewegwegew gwegwgweg we

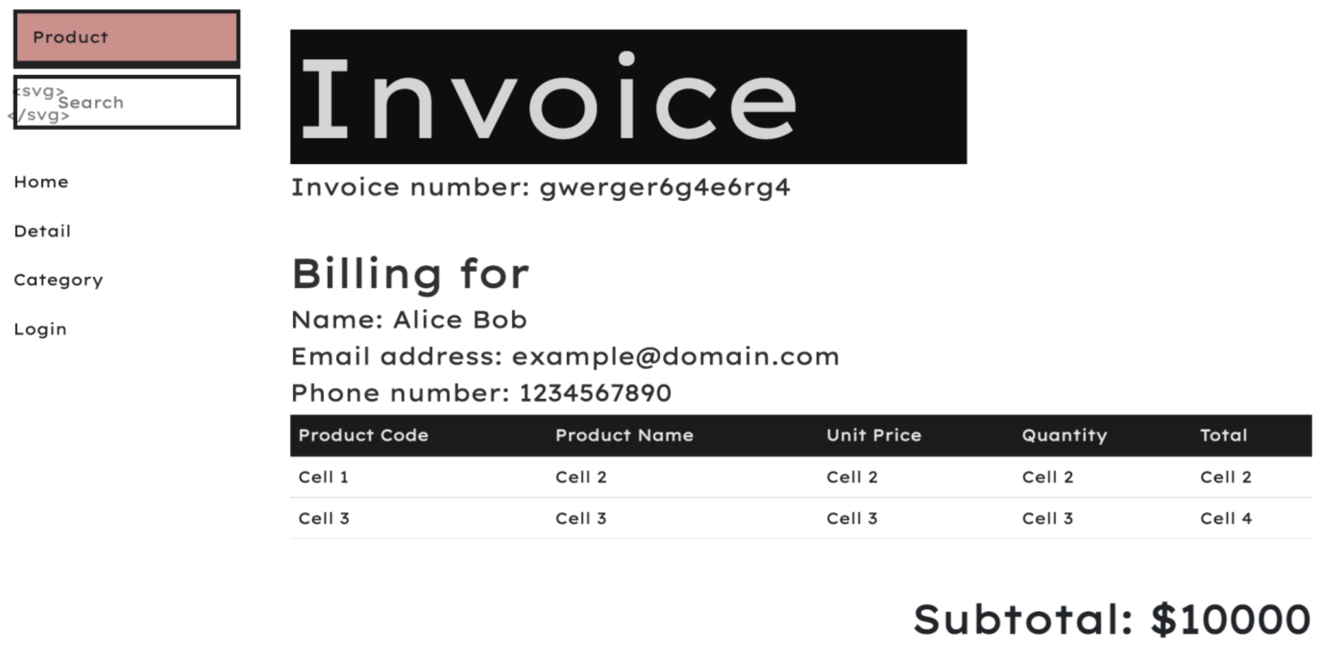
Cart:


Product
<svg>Search</svg>
Cart
Home
Detail
Category
Login
Product name
+
-
$59.99
Product name
+
-
$59.99
Subtotal:$119.99
Checkout

Login:



Checkout:



Invoice

Invoice number: gwerger6g4e6rg4

Billing for

Name: Alice Bob

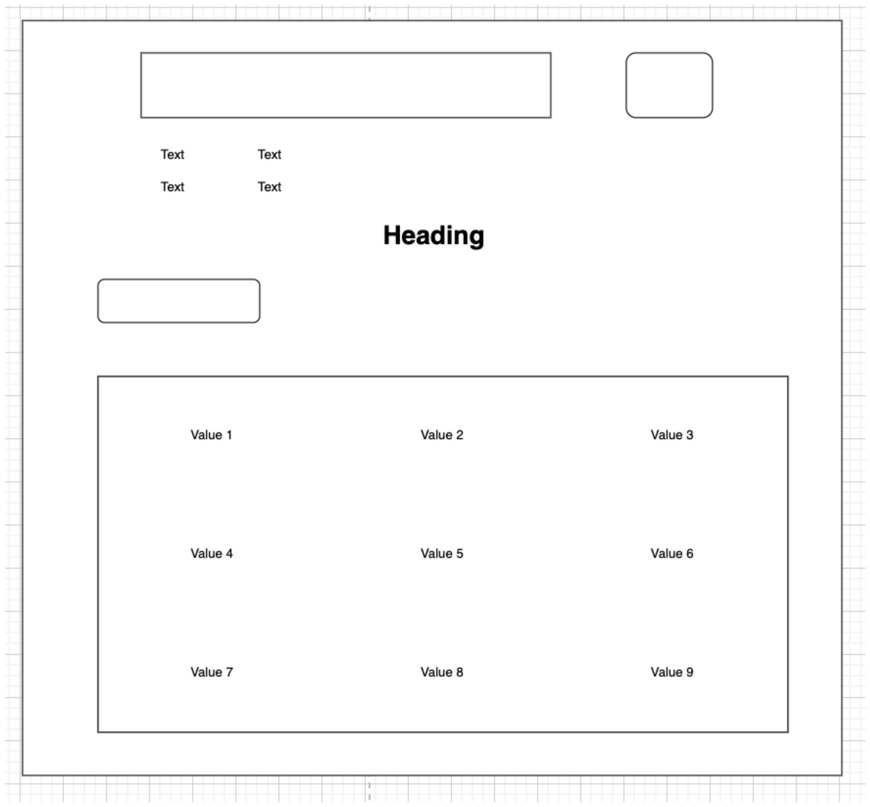
Email address: example@domain.com

Phone number: 1234567890

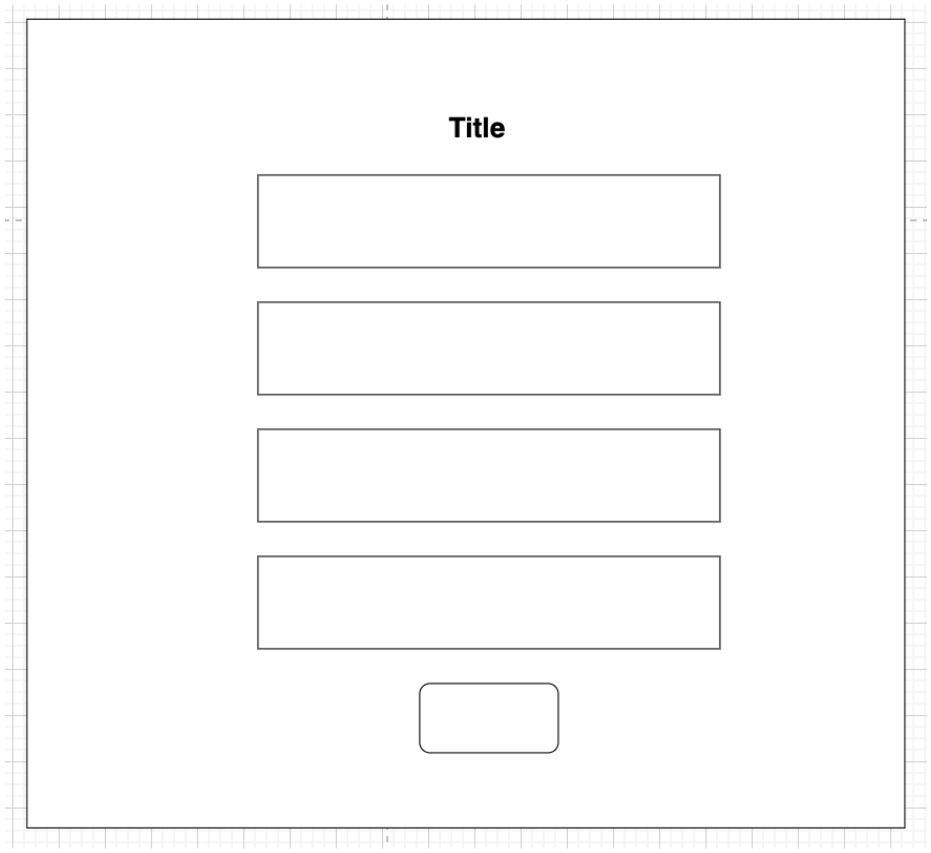
Product Code	Product Name	Unit Price	Quantity	Total
Cell 1	Cell 2	Cell 2	Cell 2	Cell 2
Cell 3	Cell 3	Cell 3	Cell 3	Cell 4

Subtotal: $10000

Product list:



Product add, edit



Part 3 – System Implementation

I. Source Code

1. The MVC Pattern

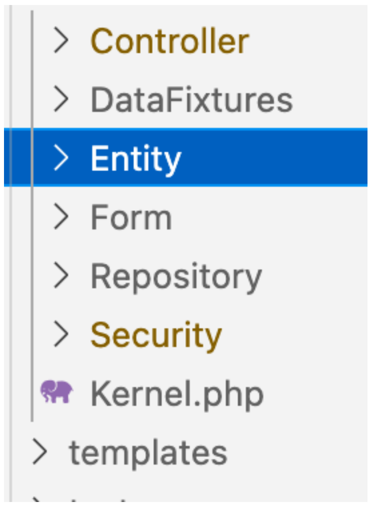
Symfony is based on the classic web design pattern known as the MVC architecture, which consists of three levels:

- The Model represents the information on which the application operates--its business logic.
- The View renders the model into a web page suitable for interaction with the user.
- The Controller responds to user actions and invokes changes on the model or view as appropriate.

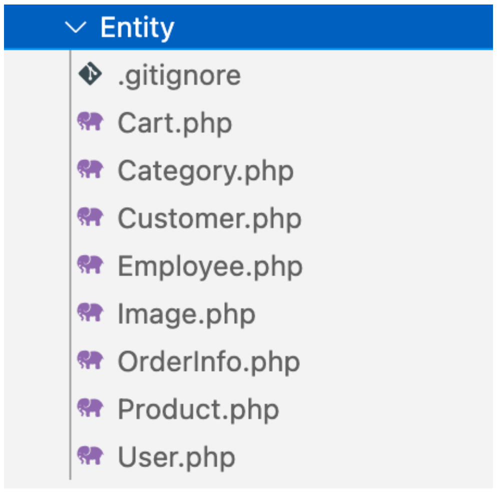
(Symfony, 2022)

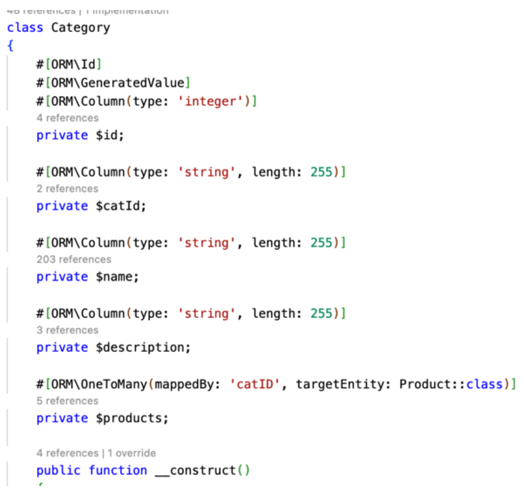
2. How it was applied in the writer project

In the design of the writer used Symfony Framework technology to be able to apply the MCV model in the design. Specifically, the writer has created 3 main MCV entities including Model (Entity), Controller, View (Template).



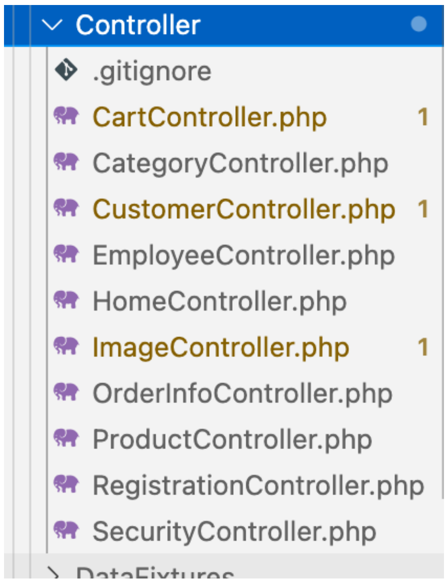
In Entity or Model: this is the main directory containing the important components of each table in the design. Entity includes entity files including: Cart, Category, Product, Customer, Employee, Image, User.And each file contains the elements corresponding to each table. For example, File Category.php will contain: id, CatId, name, description.



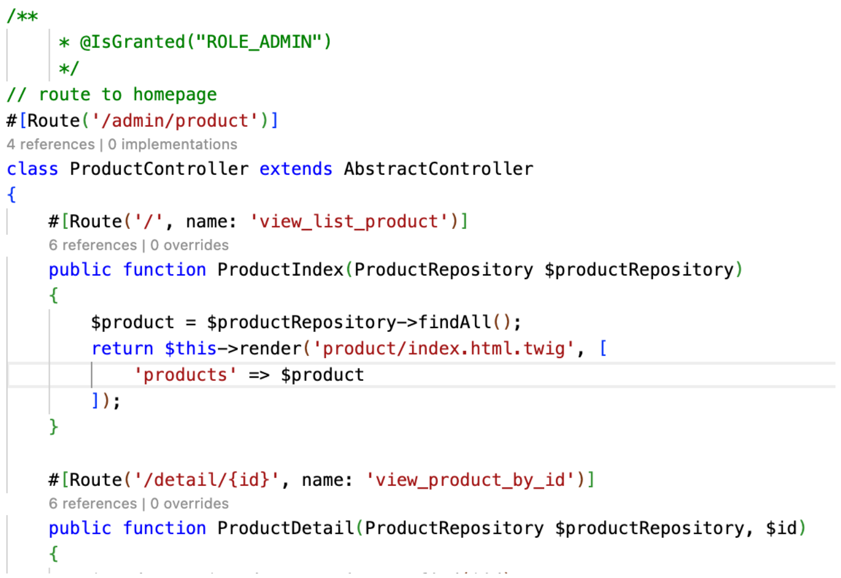


```
class Category
{
    #[ORM\Id]
    #[ORM\GeneratedValue]
    #[ORM\Column(type: 'integer')]
    4 references
    private $id;

    #[ORM\Column(type: 'string', length: 255)]
    2 references
    private $catId;

    #[ORM\Column(type: 'string', length: 255)]
    203 references
    private $name;

    #[ORM\Column(type: 'string', length: 255)]
    3 references
    private $description;

    #[ORM\OneToMany(mappedBy: 'catID', targetEntity: Product::class)]
    5 references
    private $products;

    4 references | 1 override
    public function __construct()
```

In the Controller is a folder containing all the main function files of the design including: add, edit, delete, create, search, soft, show detail, show list. The functions will be delegated to each object in the Controller.

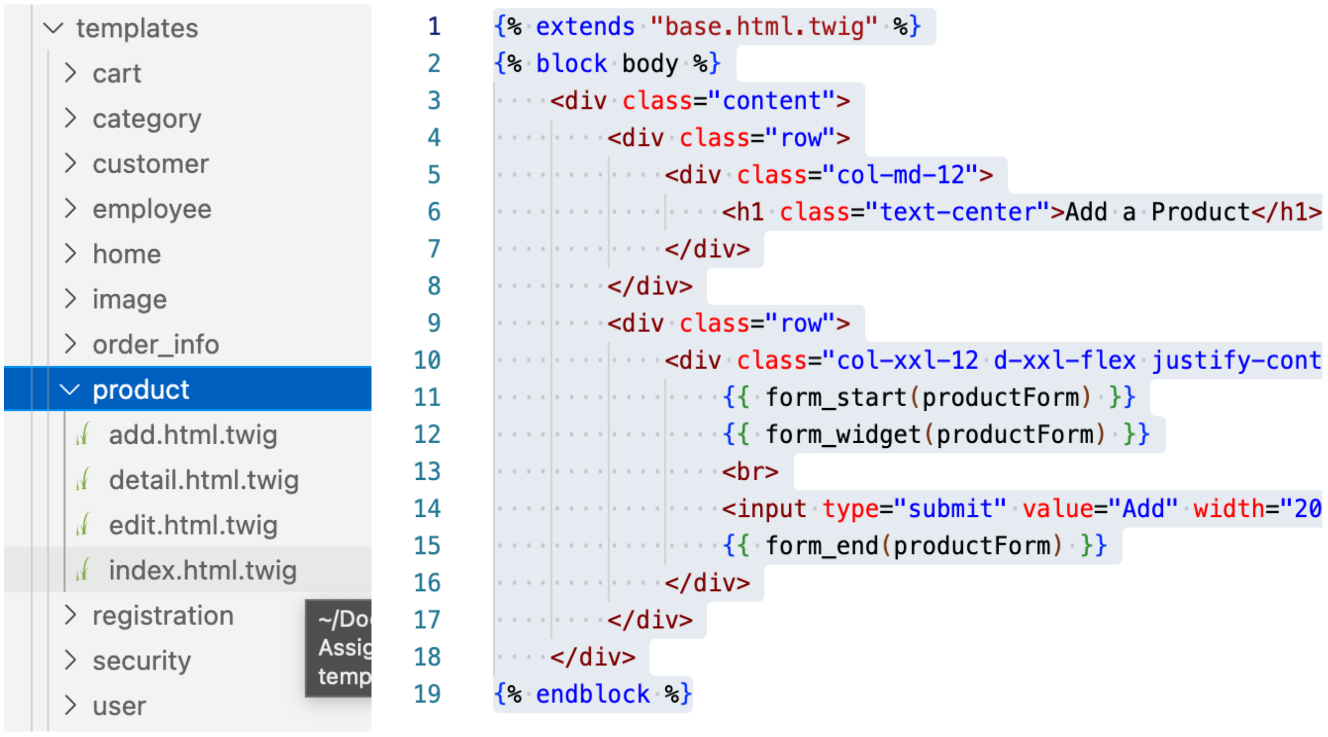


Controller
.gitignore
CartController.php 1
CategoryController.php
CustomerController.php 1
EmployeeController.php
HomeController.php
ImageController.php 1
OrderInfoController.php
ProductController.php
RegistrationController.php
SecurityController.php
DataFixtures



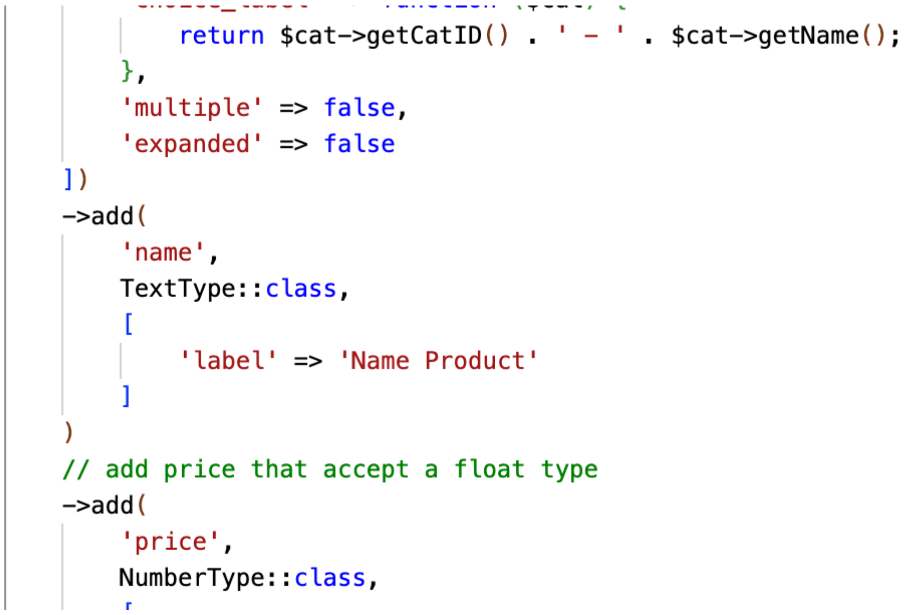
```
/**
    * @IsGranted("ROLE_ADMIN")
    */
// route to homepage
#[Route('/admin/product')]
4 references | 0 implementations
class ProductController extends AbstractController
{
    #[Route('/', name: 'view_list_product')]
    6 references | 0 overrides
    public function ProductIndex(ProductRepository $productRepository)
    {
        $product = $productRepository->findAll();
        return $this->render('product/index.html.twig', [
            'products' => $product
        ]);
    }

    #[Route('/detail/{id}', name: 'view_product_by_id')]
    6 references | 0 overrides
    public function ProductDetail(ProductRepository $productRepository, $id)
    {
```

In View or template will be html.twig files to create interfaces for users to manipulate. In general, each folder in the View setting of different entities will have 4 basic files: add.html.twig, edit.html.twig, detail.html.twig, and index.html.twig. Also other special folders will have special files.



```
{% extends "base.html.twig" %}
{% block body %}
    <div class="content">
        <div class="row">
            <div class="col-md-12">
                <h1 class="text-center">Add a Product</h1>
            </div>
        </div>
        <div class="row">
            <div class="col-xxl-12 d-xxl-flex justify-cont
                {{ form_start(productForm) }}
                {{ form_widget(productForm) }}
                <br>
                <input type="submit" value="Add" width="20
                {{ form_end(productForm) }}
            </div>
        </div>
    </div>
{% endblock %}
```

Above is how the writer applied the MCV model to his design. In addition to the above 3 main components in the design, the writer also uses DataFixture folders to create data, Form to create constraints, etc.

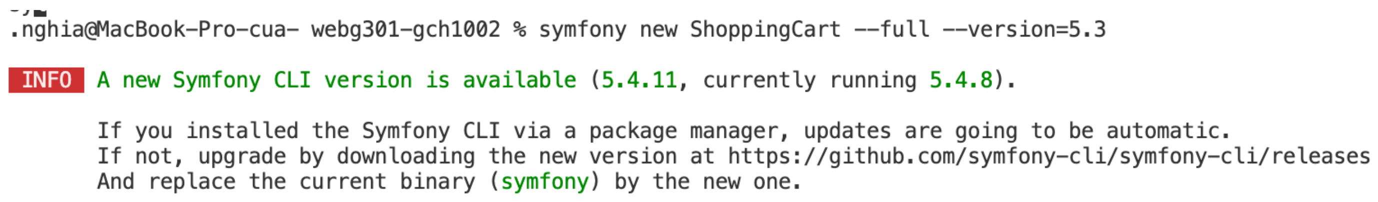


```
            return $cat->getCatID() . ' - ' . $cat->getName();
        },
        'multiple' => false,
        'expanded' => false
    ])
    ->add(
        'name',
        TextType::class,
        [
            'label' => 'Name Product'
        ]
    )
    // add price that accept a float type
    ->add(
        'price',
        NumberType::class,
```

3. Step-by-step how to develop a Symfony project

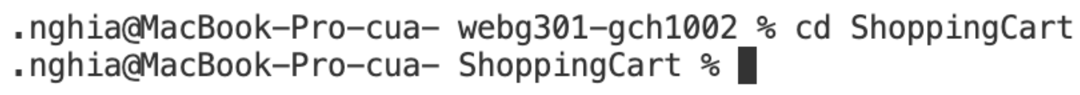
The writer will demonstrate how to create and develop a Symfony project by following these steps: (Completed Symfony, XAMPP and composer installation).

- Step1: Create a new Symfony version 5.3 project with the command " symfony new ShoppingCart --full --version=5.3 " in the terminal panel of VS code.



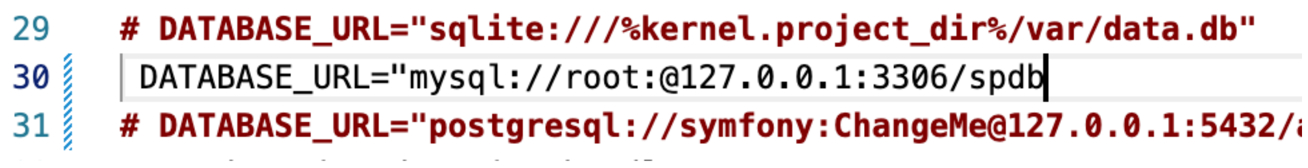
```
.nghia@MacBook-Pro-cua- webg301-gch1002 % symfony new ShoppingCart --full --version=5.3

 INFO  A new Symfony CLI version is available (5.4.11, currently running 5.4.8).

       If you installed the Symfony CLI via a package manager, updates are going to be automatic.
       If not, upgrade by downloading the new version at https://github.com/symfony-cli/symfony-cli/releases
       And replace the current binary (symfony) by the new one.
```

- Step2: After successfully creating a new project, we proceed to use the " cd ShoppingCart " command to enter the newly created project.



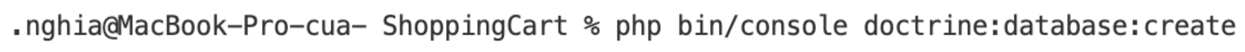
```
.nghia@MacBook-Pro-cua- webg301-gch1002 % cd ShoppingCart
.nghia@MacBook-Pro-cua- ShoppingCart %
```

- Step3: Start XAMPP and install the data base in the file ".env" . In this step you find line 31 in ".env" put comment code, then uncomment code line 30 and replace with "DATABASE_URL="mysql://root:@127.0.0.1:3306/spdb" where "spdb" is the database name of the project.



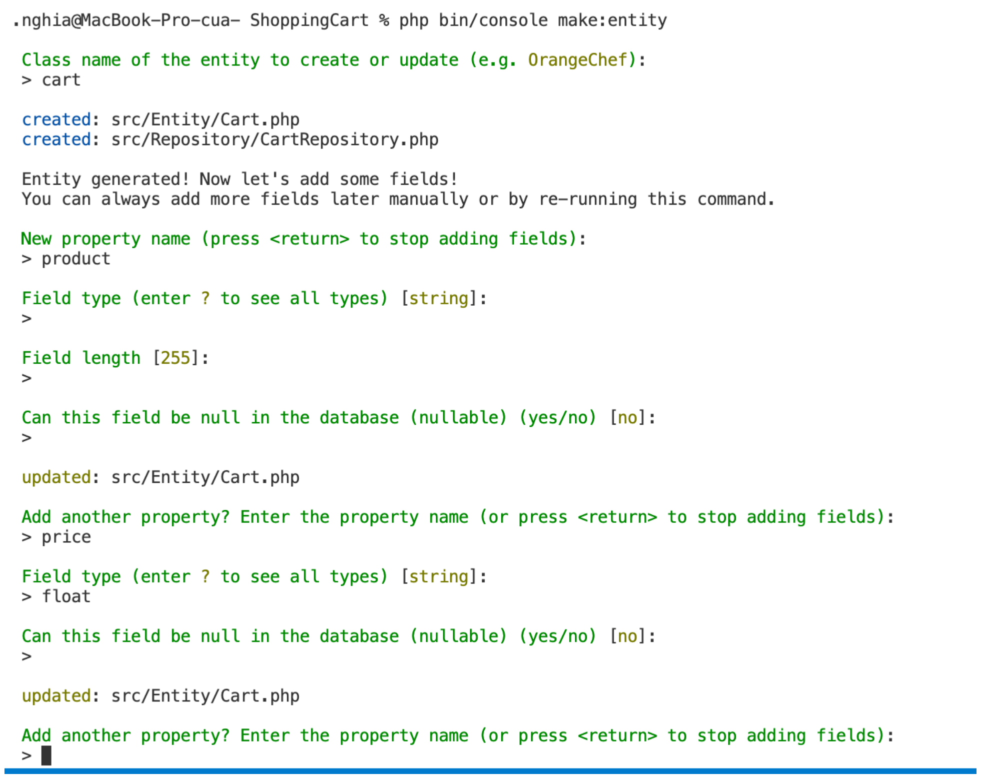
```
29   # DATABASE_URL="sqlite:///%kernel.project_dir%/var/data.db"
30   DATABASE_URL="mysql://root:@127.0.0.1:3306/spdb
31   # DATABASE_URL="postgresql://symfony:ChangeMe@127.0.0.1:5432/
```

- Step4: After setting up the database name and connection port, we continue to create the database with the command " php bin/console doctrine:database:create "



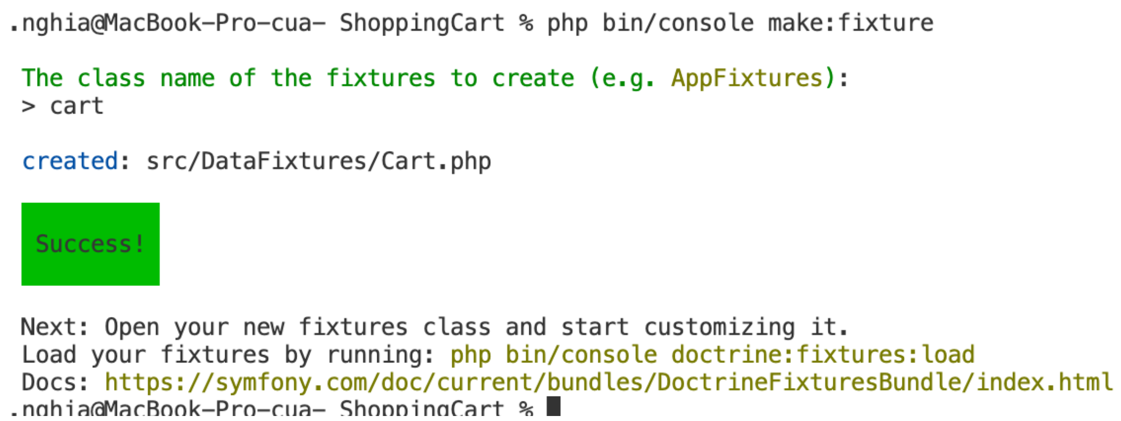
```
.nghia@MacBook-Pro-cua- ShoppingCart % php bin/console doctrine:database:create
```

- Step5: Then we continue to create entity " php bin/console make:entity", then fill in the properties, type, length for the class you instantiate.



```
.nghia@MacBook-Pro-cua- ShoppingCart % php bin/console make:entity

 Class name of the entity to create or update (e.g. OrangeChef):
 > cart

 created: src/Entity/Cart.php
 created: src/Repository/CartRepository.php

 Entity generated! Now let's add some fields!
 You can always add more fields later manually or by re-running this command.

 New property name (press <return> to stop adding fields):
 > product

 Field type (enter ? to see all types) [string]:
 >

 Field length [255]:
 >

 Can this field be null in the database (nullable) (yes/no) [no]:
 >

 updated: src/Entity/Cart.php

 Add another property? Enter the property name (or press <return> to stop adding fields):
 > price

 Field type (enter ? to see all types) [string]:
 > float

 Can this field be null in the database (nullable) (yes/no) [no]:
 >

 updated: src/Entity/Cart.php

 Add another property? Enter the property name (or press <return> to stop adding fields):
 >
```

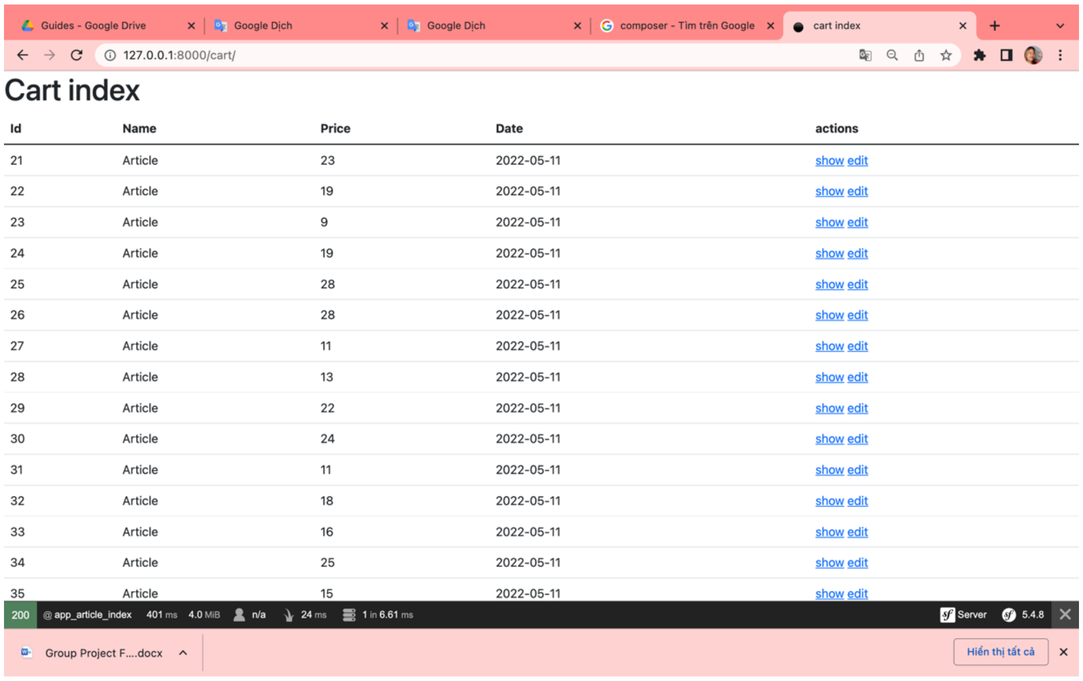
- Step6: Create a migration to generate SQL script. Enter “php bin/console make:migration”.
- Step7: Run migration to sync database with code. Enter “php bin/console doctrine:migrations:migrate”.
- Step8: then you can add data to the database using Fixture. Before that, proceed to create Fixture “composer require orm-fixtures --dev” and "php bin/console make:fixture"



```
.nghia@MacBook-Pro-cua- ShoppingCart % php bin/console make:fixture

 The class name of the fixtures to create (e.g. AppFixtures):
 > cart

 created: src/DataFixtures/Cart.php

 Success!

 Next: Open your new fixtures class and start customizing it.
 Load your fixtures by running: php bin/console doctrine:fixtures:load
 Docs: https://symfony.com/doc/current/bundles/DoctrineFixturesBundle/index.html
.nghia@MacBook-Pro-cua- ShoppingCart %
```

- Step9: To create functions in the controller you can enter the code manually in the controller file or generate CRUD automatically with the command "php bin/console make:crud".

- Step10: Finally, run the project with the command "symfony serve" link "http://127.0.0.1:8000/cart/"



127.0.0.1:8000/cart/

Cart index

Id	Name	Price	Date	actions
21	Article	23	2022-05-11	show edit
22	Article	19	2022-05-11	show edit
23	Article	9	2022-05-11	show edit
24	Article	19	2022-05-11	show edit
25	Article	28	2022-05-11	show edit
26	Article	28	2022-05-11	show edit
27	Article	11	2022-05-11	show edit
28	Article	13	2022-05-11	show edit
29	Article	22	2022-05-11	show edit
30	Article	24	2022-05-11	show edit
31	Article	11	2022-05-11	show edit
32	Article	18	2022-05-11	show edit
33	Article	16	2022-05-11	show edit
34	Article	25	2022-05-11	show edit
35	Article	15	2022-05-11	show edit

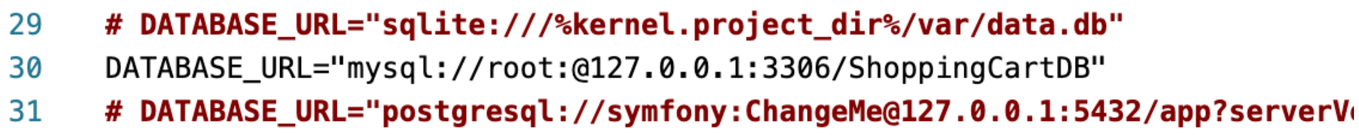
Guides - Google Drive
Google Dịch
Google Dịch
composer - Tìm trên Google
cart index
200 @app_article_index 401 ms 4.0 MiB n/a 24 ms 1 in 6.61 ms Server 5.4.8
Group Project F....docx
Hiển thị tất cả

Above are the basic steps to create a project using Symfony.

4. Extra important codes

4.1 .env

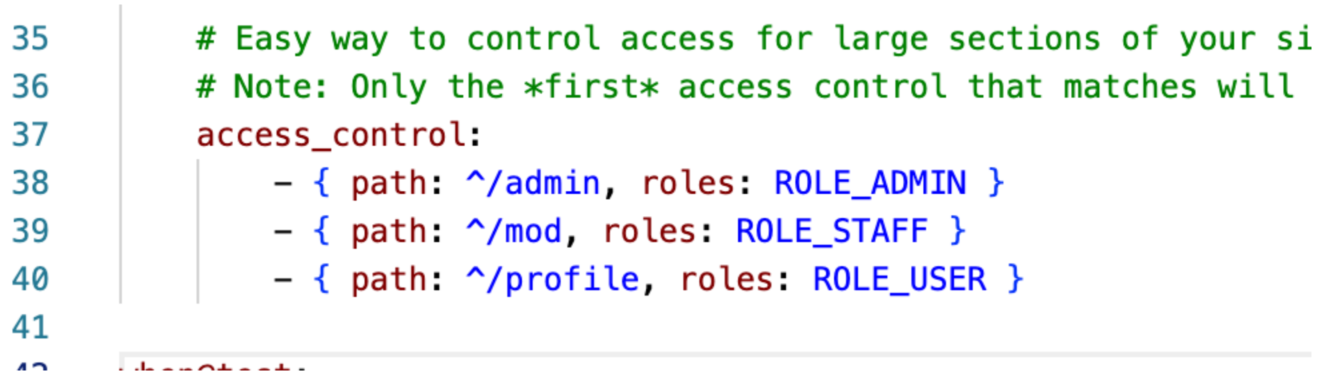
".env" is the setup data file for the project. The project is currently using database technology provided by XAMPP. The ".env" setup steps include commenting the line code 31 and uncommenting the line 30 code and replacing it with "DATABASE_URL="mysql://root:@127.0.0.1:3306/ database name". "3306" connection port can be changed depending on the user's needs, for example: "3307", "3308", etc. This change is meant to create a path with the database name and its connection port.



```
29    # DATABASE_URL="sqlite:///%kernel.project_dir%/var/data.db"
30    DATABASE_URL="mysql://root:@127.0.0.1:3306/ShoppingCartDB"
31    # DATABASE_URL="postgresql://symfony:ChangeMe@127.0.0.1:5432/app?serverV
```

4.2 Security.yaml

"security.yaml" is in "config/packages/security.yaml". Changes to this file include setting up roles for the users of the design. In this design are Admin, Staff and User. Change made at line 37 "access_control:" and added the following roles:

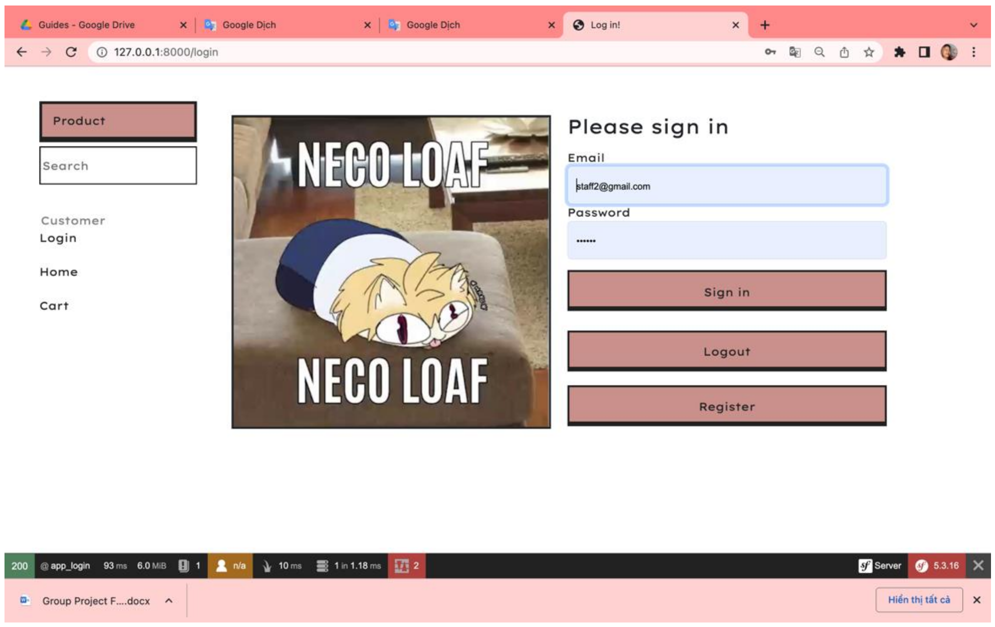


```
    # Easy way to control access for large sections of your si
    # Note: Only the *first* access control that matches will
    access_control:
        - { path: ^/admin, roles: ROLE_ADMIN }
        - { path: ^/mod, roles: ROLE_STAFF }
        - { path: ^/profile, roles: ROLE_USER }
```

This change plays the role of setting roles and decentralizing objects.

II. Web screenshots

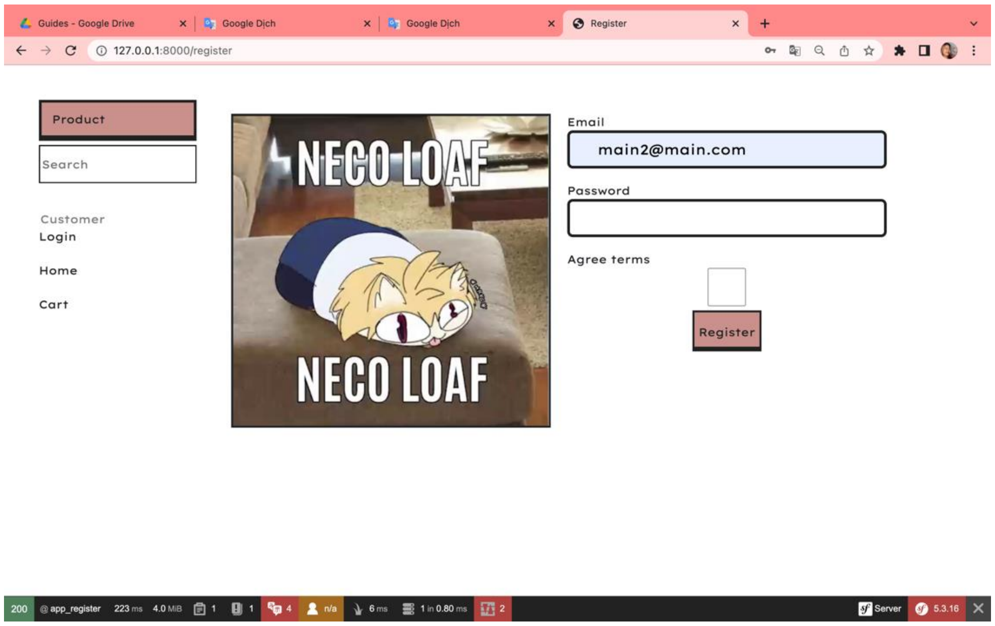
Login: In the login page, the user will proceed to log in to the account with the previous account. Otherwise they can choose register to create a user account. Upon successful login, the user will be redirected to the Home page.



Email: staff2@gmail.com

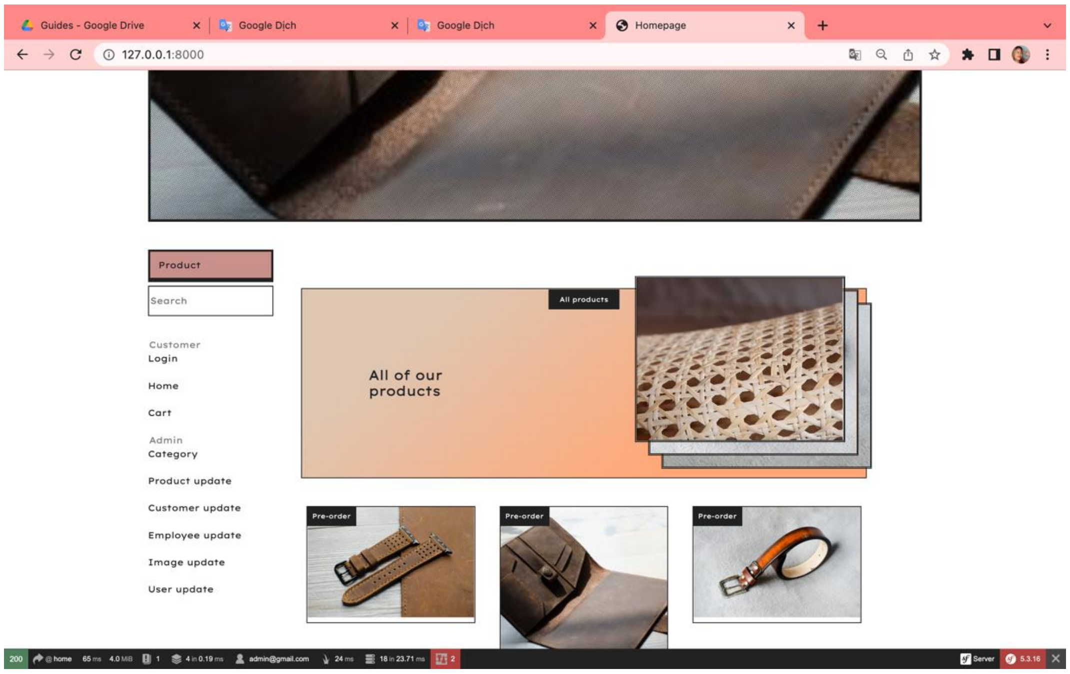
Password: ••••••

Register: Register allows you to register an account with the role of user, you just need to enter information

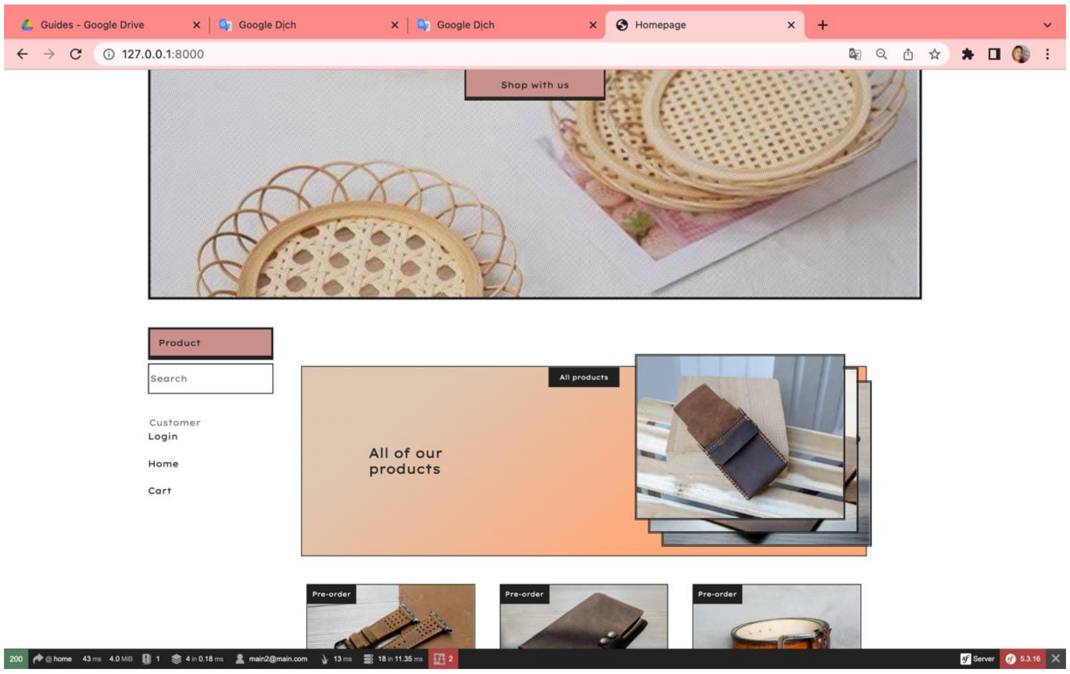


Home page: The home page will include products with functions for each user object. In addition, there will be an additional search function by product name.

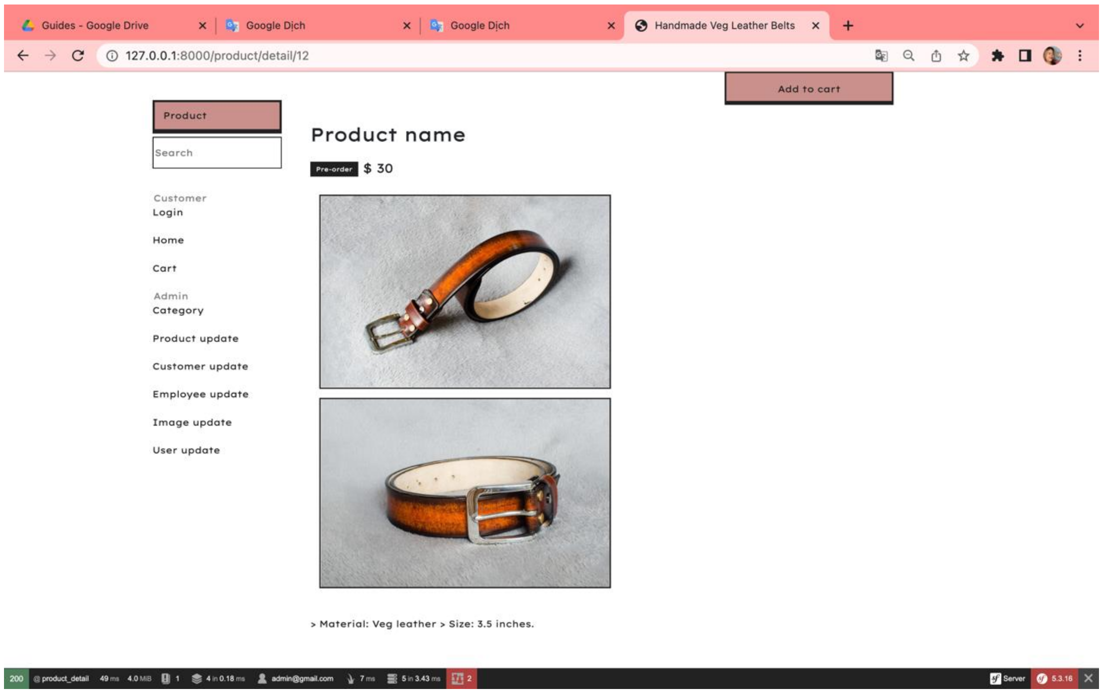
- Home page with Admin:



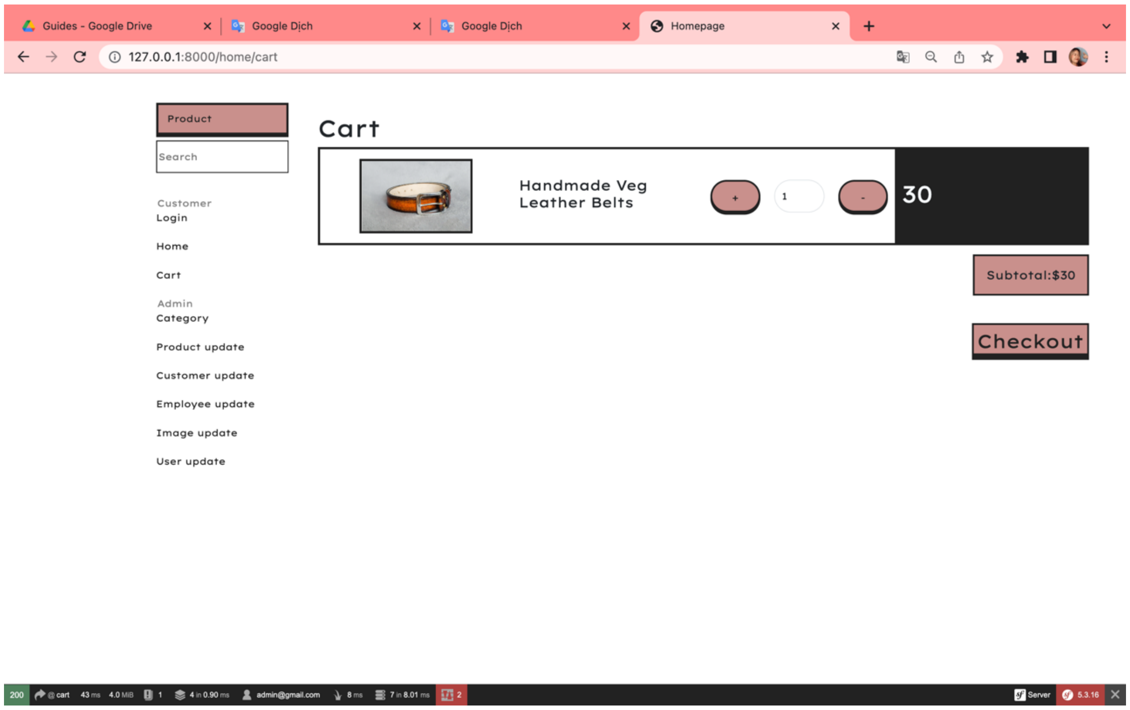
- Home page with User:

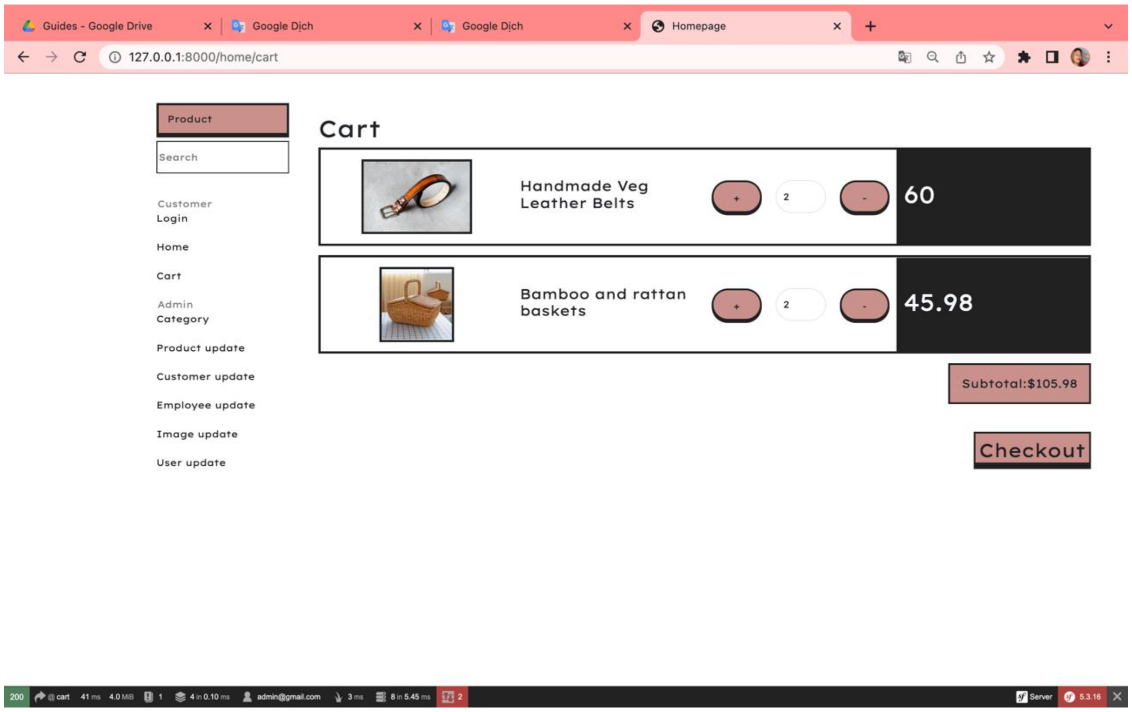


Product detail: By clicking on any product, you will be taken to the product detail page. The page will have product photos, product information and add to cart functionality.



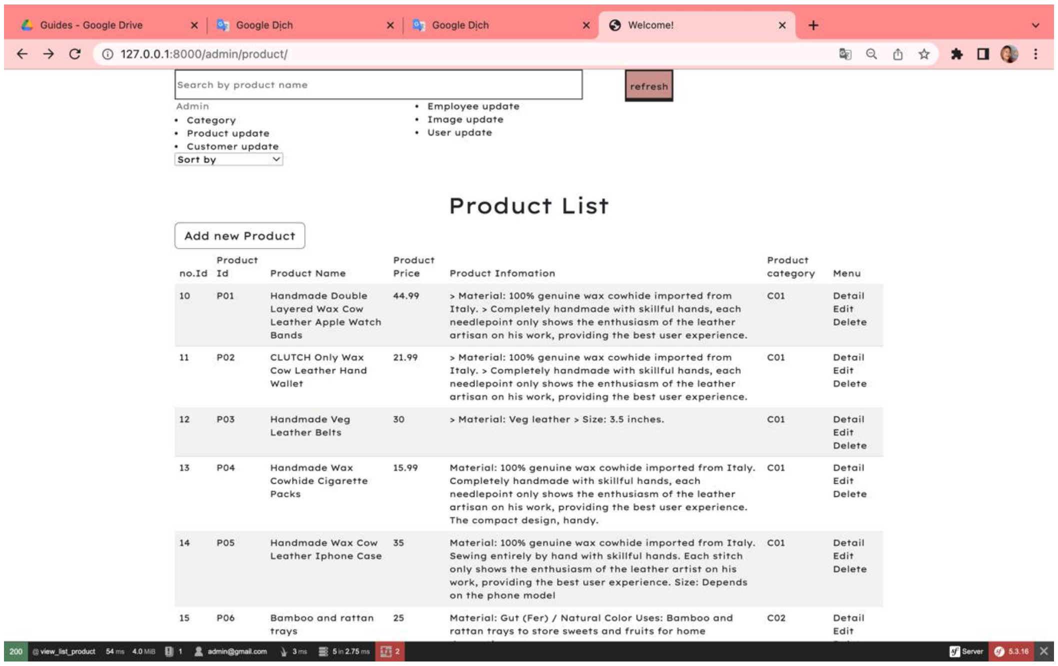
Cart: Cart is generated when you click create cart. At this time, there will be the product you have added in the cart. The website will display product photos, quantity increase and decrease, price and total price. You can also add multiple products to the same cart.


Guides - Google Drive
Google Dịch
Google Dịch
Homepage
127.0.0.1:8000/home/cart
Product
Search
Customer
Login
Home
Cart
Admin
Category
Product update
Customer update
Employee update
Image update
User update
Cart
Handmade Veg Leather Belts
+
1
-
30
Subtotal:$30
Checkout


Guides - Google Drive
Google Dịch
Google Dịch
Homepage
127.0.0.1:8000/home/cart
Product
Search
Customer
Login
Home
Cart
Admin
Category
Product update
Customer update
Employee update
Image update
User update
Cart
Handmade Veg Leather Belts
+
2
-
60
Bamboo and rattan baskets
+
2
-
45.98
Subtotal:$105.98
Checkout

CRUD:

- Product index contains all information of the product in the home page including ID, name, price, information. in which it also contains the functions of adding, editing, deleting and viewing product details.



127.0.0.1:8000/admin/product/

Search by product name

refresh

Admin
- Category
- Product update
- Customer update
- Employee update
- Image update
- User update

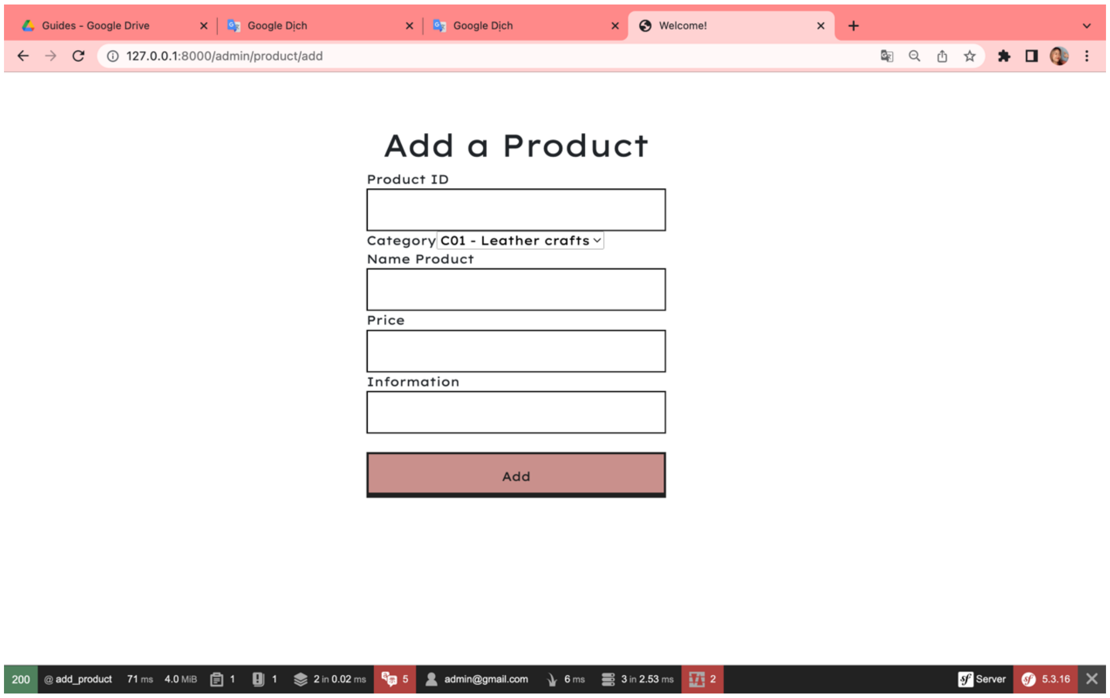
Sort by

Product List

Add new Product

no.Id	Product Id	Product Name	Product Price	Product Infomation	Product category	Menu
10	P01	Handmade Double Layered Wax Cow Leather Apple Watch Bands	44.99	> Material: 100% genuine wax cowhide imported from Italy. > Completely handmade with skillful hands, each needlepoint only shows the enthusiasm of the leather artisan on his work, providing the best user experience.	C01	Detail Edit Delete
11	P02	CLUTCH Only Wax Cow Leather Hand Wallet	21.99	> Material: 100% genuine wax cowhide imported from Italy. > Completely handmade with skillful hands, each needlepoint only shows the enthusiasm of the leather artisan on his work, providing the best user experience.	C01	Detail Edit Delete
12	P03	Handmade Veg Leather Belts	30	> Material: Veg leather > Size: 3.5 inches.	C01	Detail Edit Delete
13	P04	Handmade Wax Cowhide Cigarette Packs	15.99	Material: 100% genuine wax cowhide imported from Italy. Completely handmade with skillful hands, each needlepoint only shows the enthusiasm of the leather artisan on his work, providing the best user experience. The compact design, handy.	C01	Detail Edit Delete
14	P05	Handmade Wax Cow Leather Iphone Case	35	Material: 100% genuine wax cowhide imported from Italy. Sewing entirely by hand with skillful hands. Each stitch only shows the enthusiasm of the leather artist on his work, providing the best user experience. Size: Depends on the phone model	C01	Detail Edit Delete
15	P06	Bamboo and rattan trays	25	Material: Gut (Fer) / Natural Color Uses: Bamboo and rattan trays to store sweets and fruits for home	C02	Detail Edit

- Add product: Admin can add a new product including attributes: product ID, category, name, price, infomation



Guides - Google Drive | Google Dịch | Google Dịch | Welcome!

127.0.0.1:8000/admin/product/add

Add a Product

Product ID

Category C01 - Leather crafts

Name Product

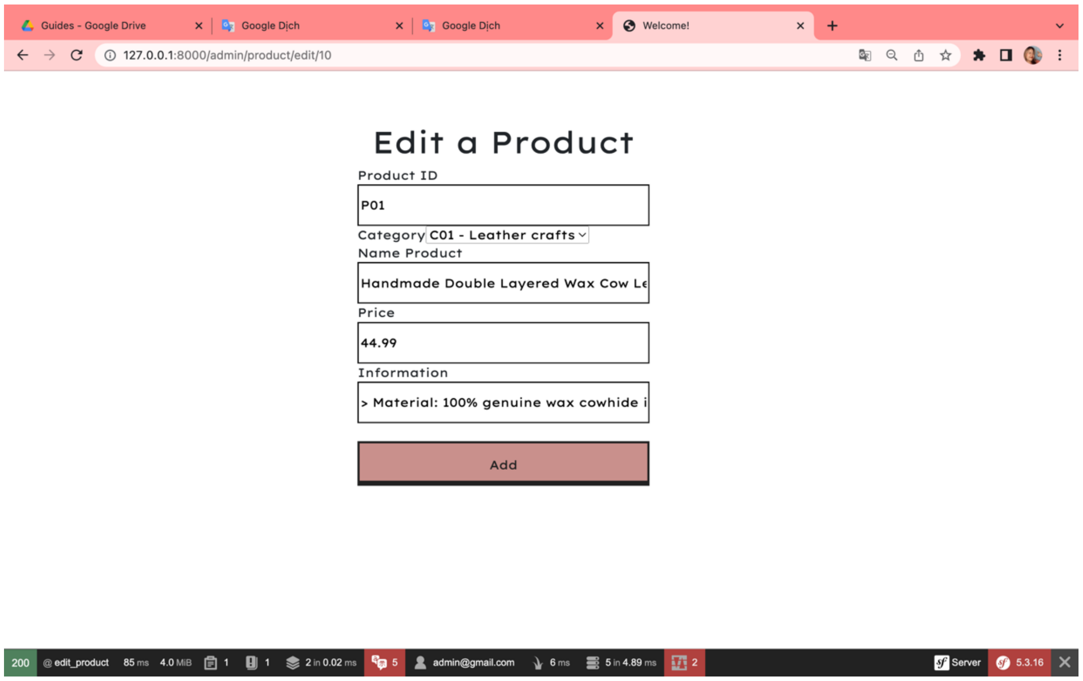
Price

Information

Add

200 @ add_product 71 ms 4.0 MiB 1 1 2 in 0.02 ms 5 admin@gmail.com 6 ms 3 in 2.53 ms 2 Server 5.3.16

- Edit product: Just like with add, admins can also modify the product information at their discretion.



Guides - Google Drive | Google Dịch | Google Dịch | Welcome!

127.0.0.1:8000/admin/product/edit/10

Edit a Product

Product ID: P01

Category C01 - Leather crafts

Name Product: Handmade Double Layered Wax Cow Le

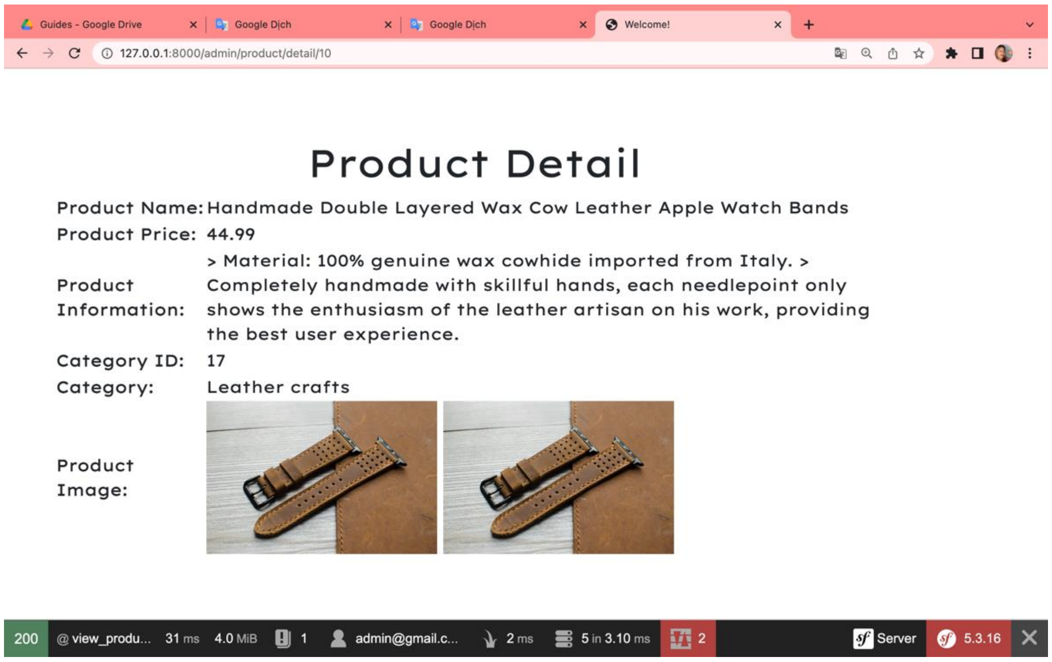
Price: 44.99

Information: > Material: 100% genuine wax cowhide i

Add

200 @ edit_product 85 ms 4.0 MiB 1 1 2 in 0.02 ms 5 admin@gmail.com 6 ms 5 in 4.89 ms 2 Server 5.3.16

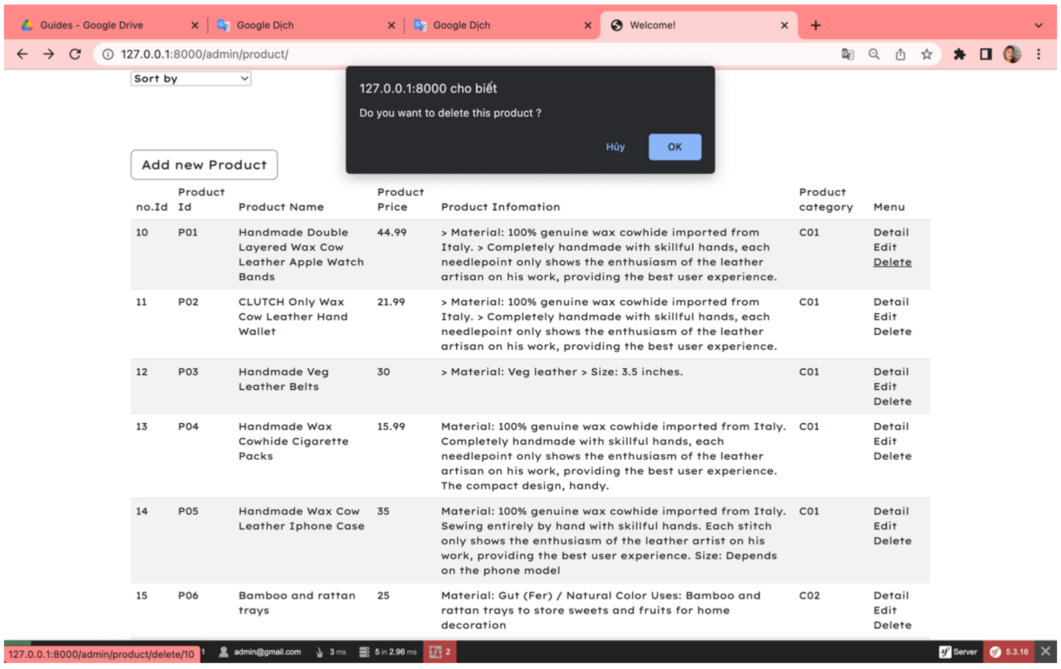
- Detail product: Contains product-specific information and product photos


Guides - Google Drive
Google Dịch
Google Dịch
Welcome!
127.0.0.1:8000/admin/product/detail/10
200
@ view_produ... 31 ms 4.0 MiB 1 admin@gmail.c... 2 ms 5 in 3.10 ms 2
Server 5.3.16

Product Detail

Product Name:	Handmade Double Layered Wax Cow Leather Apple Watch Bands
Product Price:	44.99
Product Information:	> Material: 100% genuine wax cowhide imported from Italy. > Completely handmade with skillful hands, each needlepoint only shows the enthusiasm of the leather artisan on his work, providing the best user experience.
Category ID:	17
Category:	Leather crafts
Product Image:	

- Delete product: You need to select delete in the product table to delete the product. The website will issue a notice whether you want to delete the product or not. Click ok to continue deleting and cancel to cancel the operation.


Guides - Google Drive
Google Dịch
Google Dịch
Welcome!
127.0.0.1:8000/admin/product/
Sort by
Hủy
OK
Add new Product
127.0.0.1:8000/admin/product/delete/10
admin@gmail.com 3 ms 5 in 2.96 ms 2
Server 5.3.16

127.0.0.1:8000 cho biết

Do you want to delete this product ?

no.Id	Product Id	Product Name	Product Price	Product Infomation	Product category	Menu
10	P01	Handmade Double Layered Wax Cow Leather Apple Watch Bands	44.99	> Material: 100% genuine wax cowhide imported from Italy. > Completely handmade with skillful hands, each needlepoint only shows the enthusiasm of the leather artisan on his work, providing the best user experience.	C01	Detail Edit Delete
11	P02	CLUTCH Only Wax Cow Leather Hand Wallet	21.99	> Material: 100% genuine wax cowhide imported from Italy. > Completely handmade with skillful hands, each needlepoint only shows the enthusiasm of the leather artisan on his work, providing the best user experience.	C01	Detail Edit Delete
12	P03	Handmade Veg Leather Belts	30	> Material: Veg leather > Size: 3.5 inches.	C01	Detail Edit Delete
13	P04	Handmade Wax Cowhide Cigarette Packs	15.99	Material: 100% genuine wax cowhide imported from Italy. Completely handmade with skillful hands, each needlepoint only shows the enthusiasm of the leather artisan on his work, providing the best user experience. The compact design, handy.	C01	Detail Edit Delete
14	P05	Handmade Wax Cow Leather Iphone Case	35	Material: 100% genuine wax cowhide imported from Italy. Sewing entirely by hand with skillful hands. Each stitch only shows the enthusiasm of the leather artist on his work, providing the best user experience. Size: Depends on the phone model	C01	Detail Edit Delete
15	P06	Bamboo and rattan trays	25	Material: Gut (Fer) / Natural Color Uses: Bamboo and rattan trays to store sweets and fruits for home decoration	C02	Detail Edit Delete

Part 4 – Conclusion

I. Advantages of website

The website meets the requirements set by the lecturer. The design has handled all possible functions well and the interface is finished to a decent level.

- User operation is smooth, friendly, easy to use and does not generate minor errors during use.
- The design has outlined a popular shopping cart website structure, meeting the needs of use.
- The design has decentralized permissions for different roles like user and admin. User will not have the right to use special Admin functions.
- The manageability and crud of the design is complete and can help admin manage it easily.

II. Disadvantages of website

Design some issues that the design team needs to pay attention to.

- The interface design has not been streamlined in terms of location and functions are not really impressed with users. The interface of add and edit is sketchy and unremarkable.
- The design doesn't handle payment and delivery functionality yet. However, the function has been omitted by the lecturer and does not affect the outcome of the project.
- Products are not diverse enough in quantity and category.

III. Lesson learnt

Through the project, the design team and writers gained a lot of experience in web design and how to use Symfony Framework in web creation. The group has united and created opportunities to help each other complete the work. The ability to work in groups is also improved and enhanced. Regarding the project, the team members were able to handle the errors that arose during the web building process. The basic functions of the web such as crud, search, and soft are well understood by the members. Through this project, the team gained a lot of useful experience in future web projects.

IV. Future improvements

Based on the weaknesses, the team proposes future improvements as follows:

- Improve the interface in the near future to suit the needs of customers.
- Add validation attributes to input information so that the information can be most accurate and minimize the risk.
- Add payment and delivery functionality to the site in the near future.
- Upgrade database system for products and users.

Part 5 – Appendix

I. Group member list + role

Member	Role	Group dedication (1-10)
Trinh Duc Anh (leader)	Code implementation: interface (frontend), User function, Login page	10
Dao Trong Nghia	Code implementation: Admin functions, decentralization, product information search.	10
Nguyen Thu Ha	Execute the code: crud orderinfo, cart and checkout	10

II. Link to code on GitHub

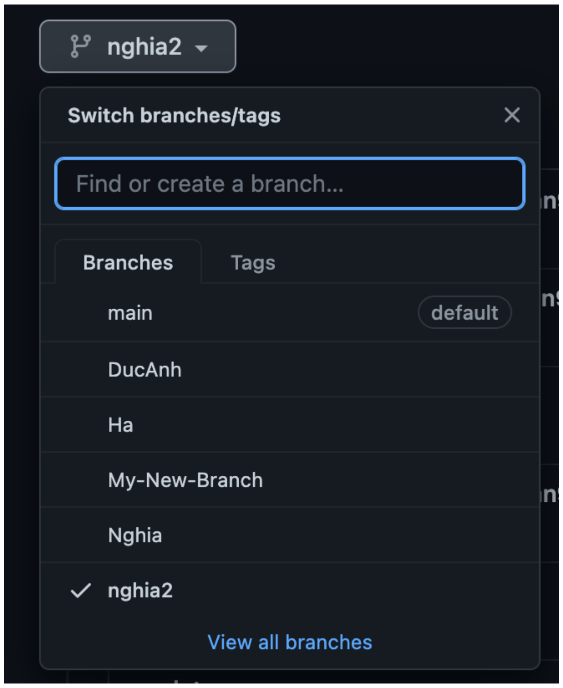
Link: github.com/wildman9x/GCH1002-Web301-Assignment-ShoppingCart

Branch: Trinh Duc Anh :DucAnh

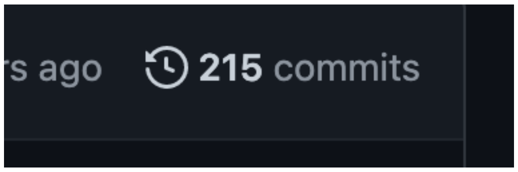
Branch: Dao Trong Nghia: Nghia2

Branch: Nguyen Thu Ha: Ha

Branch Main: Main



Commit: 215 Commits


rs ago 215 commits