INTERNATIONAL UNIVERSITY – HCMVNU **Group: 3**

FACULTY OF COMPUTER SCIENCE AND ENGINEERING *Class: Friday theory*

Principle of database managements *Instructor: Nguyen Thi Thuy Loan*

Topic 10: E-Commerce website for automotive parts

ID	Name	Phone number	Contribution percentage
ITITIU19228	Trần Nguyễn Thương Trường (leader)	0917085937	100%
ITITIU19173	Trần Hữu Nhật	0342958368	100%
ITITIU19225	Nguyễn Trung Trực	0911680270	100%
ITITIU19181	Lê Quốc Phong	072243023	100%
ITITIU19201	Trần Thanh Sơn	0916407532	100%

TABLE OF CONTENTS

I. INTRODUCTION

1.1. Requrement of topic

- Under this project, you can develop a standard e-commerce website that displays products to be sold. Users must create account be able to select the products they want to buy and add them to cart. Then able to make checkout the their orders and payments via a secure payment authenticate gateway.

1.2. Requiement of proposal & report

- Student informations.
- Topic informations.
- Detailed description of the topic.
- Implementation plan.
- Ouline of midterm project.
- ERD digram of database system.

1.3. Action workflows

- Generate ideas → plan implementation → build a roadmap → divide specific tasks for each member → development → production.
- Monitor progress, evaluate and check after each phase.
- Building user interface and experience for client (UI/UX).
- Handling logics and interactions action of data in server.
- Initialize the database (CRUD diagram) & connect to server.
- Data is communicated between client and server through API.
- Backend-side is where data can be written and readed from the database.
- Presentations slide & report: All member.
- Frontend - Design team: Trường, Nhật, Phong.
- Backend - Database team: Trường, Trực, Sơn.

1.4. Why we choose the topic

- Nowadays, the e-commerce market is becoming a part of everyday life. With just a few simple steps, through our personal phone or laptop, we can easily search and buy our favorite products.
- The scope and functionality of the website is quite extensive, requiring a combination of programming knowledge to create it ((OOP, Database, API, Algorithm, UI/UX, Responsive web...). Therefore, we can both practice and improve our programming knowledge in the process of website development.
- Famous and popular e-commerce websites such as: “Amazon, Shopee, Tiki, Lazada, Alibaba …”. They are all designed with beautiful interfaces, great experiences to attract consumers.

1.5. Planning

- We think about all the business rules and scenarios that will lead to the project's outcome.
- Almost all businesses now use online services to modernize.
- System scope of use of the website is not limited to age and gender.

- ERD will be in the database. Information fields will be added when the user selects the product.
- Elegant design will please the viewer and regularly updated on the website to increase the number of users.
- But the first and important goal here is to design how the idea, roadmap and workflow go through the system instead of focusing on the front-end and back-end.
- Back to the system, business processes and website operation will be applied as follows:
  - When users register for an account and are entering information on the page, they can shop online by searching for their favorite products and adding to their shopping cart.
  - The products will have illustrations, product names, prices. Products added to the cart will be added to the waiting list. And the user has the right to confirm the purchase or remove it from their list. After purchasing and choosing to buy the product, the user will complete the information and pay at the "checkout-page".
  - A list of discounted products and remaining quantities are also displayed according to the online time frame, along with the top selling products and recommended products are also displayed.
  - Users can easily edit personal information and public delivery address at "account-page".
  - With sellers, they will see the total of all pending and canceled orders, the total of all products in their store. Sellers will add products to their store at the “add-product page”. In addition, all information or addresses about the seller is also public at the "shop-profile page".
  - Seller and user can chat directly with each other.
  - Admin has the highest authority in the system, they can see the total number of orders, total number of users and new registrations, total products, seller list, user list, can add-remove-edit users, add-remove-edit seller. Every detail is represented as a visual chart.
  - Admin can chat directly with the seller.
  - In addition, the website system is supplemented with multi-language mode, dark / light mode.

II. ANALYSIS

2.1 Description of topic

- Our platform name is e-shopee. In this project, we referened to shopee.vn website as a main focus studying. However, interface and relationship database is minimization and customizing to be appropriate.
- Feather:
  - Multi authenticate account.
  - Verify users through mail code.
  - Online chat messages with seller and admin.
  - Discounting on real time.
  - Dark/Light mode inside.

    - Multi languages (English, Vietnamese, Japanese, Spanish, Chinese, Indian, Russia…)

- Languages & tools:
    - Frontend: HTML, CSS, SCSS, JavaScript, TypeScript, Vue…
    - Backend: Laravel, PHP, TypeScript, Blade..
    - Database managements: PostgreSQL, SQL…
    - Support tools: XAMPP, phpMyAdmin, postgreDB, Docker…
    - Docs tools: Postman, Swagger, Draw.io, Github…
- The website system have **3-role** view: Buyer - Seller – Admin in Frontend-side equal to MVC parttern in Backend-side.
- **<u>Buyer role:</u>**

✓ Register/Login-page:
  - Start with new account for buyer.
  - Register with username or email and verification email code.
  - Reset password through email if forgot.

✓ Home-page:
  - Menu of all kinds of products.
  - Search product.
  - Add product to cart.
  - Flash sales in discounts list.
  - List of hint products today.
  - List of discounted products & romotion time remaining.
  - List of top best-selling products.

✓ Product-details-page
  - Show a complete display of product details that the user wants to see
  - Consist (product image, name, sizes, category of, color, quantities, star-votes come-from, descriptions, comments of other users, represention shop preview…).
  - Summary preview about owner shop of this products.

✓ Cart-page:
  - View, Update, Select, Delete all added products in the cart.
  - Display prices, preview image, description, name, quantity of products.
  - Total bill before checkout.

✓ Checkout-page:
  - Setup all information of account profile (name, email, phone number, payment method, credit card details, address, postal code…) before checkout if they're empty.
  - Choose a delivery service and make a note on delivery.

✓ Account-page:
  - Setup, update, delete all information of account profile (name, email, phone number, payment method, credit card details, address, postal code…)
  - Authenticate user.

✓ About-page:
  - Informations & description of project and group.

- ✓ Contact-page:
    - Address details, hotline and email to main company.
    - Send contact messages (active in future).
- ✓ Sitemap-page:
    - Google-map address.
- ✓ Privacy legal & FAQ.
- ✓ Services & contact.

- **<u>Seller role</u>**
  - ✓ Dashboard-page:
    - Visualize view all informations of seller controler.
  - ✓ Orders-page:
    - Display the list of all orders (sort by time).
  - ✓ Product-page:
    - Filling shop information (if empty) before add new categories.
    - Create and update products information (sizes, prices, status, quantities...).
    - List all products on sale and available in the shop.
    - Filter and find products by names, categories, quantities...
  - ✓ Notifications-page:
    - Display all notifications from customer and system.
  - ✓ Shop-preview:
    - Display shop profile (avatar, cover, name, address, total products, review…)
    - Edit shop profile details.
  - ✓ Shop-profile-preview:
    - Display shop profile when public.
  - ✓ Advertising-page:
    - Manage advertising.
  - ✓ Chat-page (active in future):
    - Chat with admin.
    - Chat with customer.
- **<u>Admin role</u>**
  - ✓ Login-page:
    - Login by admin email/password and verify through email.
    - Reset new password through email if forgot.
  - ✓ Dashboard-page:
    - Profile information of admin.
    - Manage customer list and seller list.
    - Total products, total orders, total sales, total new users.
    - Add, Update, Delete any users or sellers.
    - Searching products by categories.
    - View and join advertisements (active in future).
    - Chat with seller (active in future).

2.2 Entities relationship diagram (ERD)

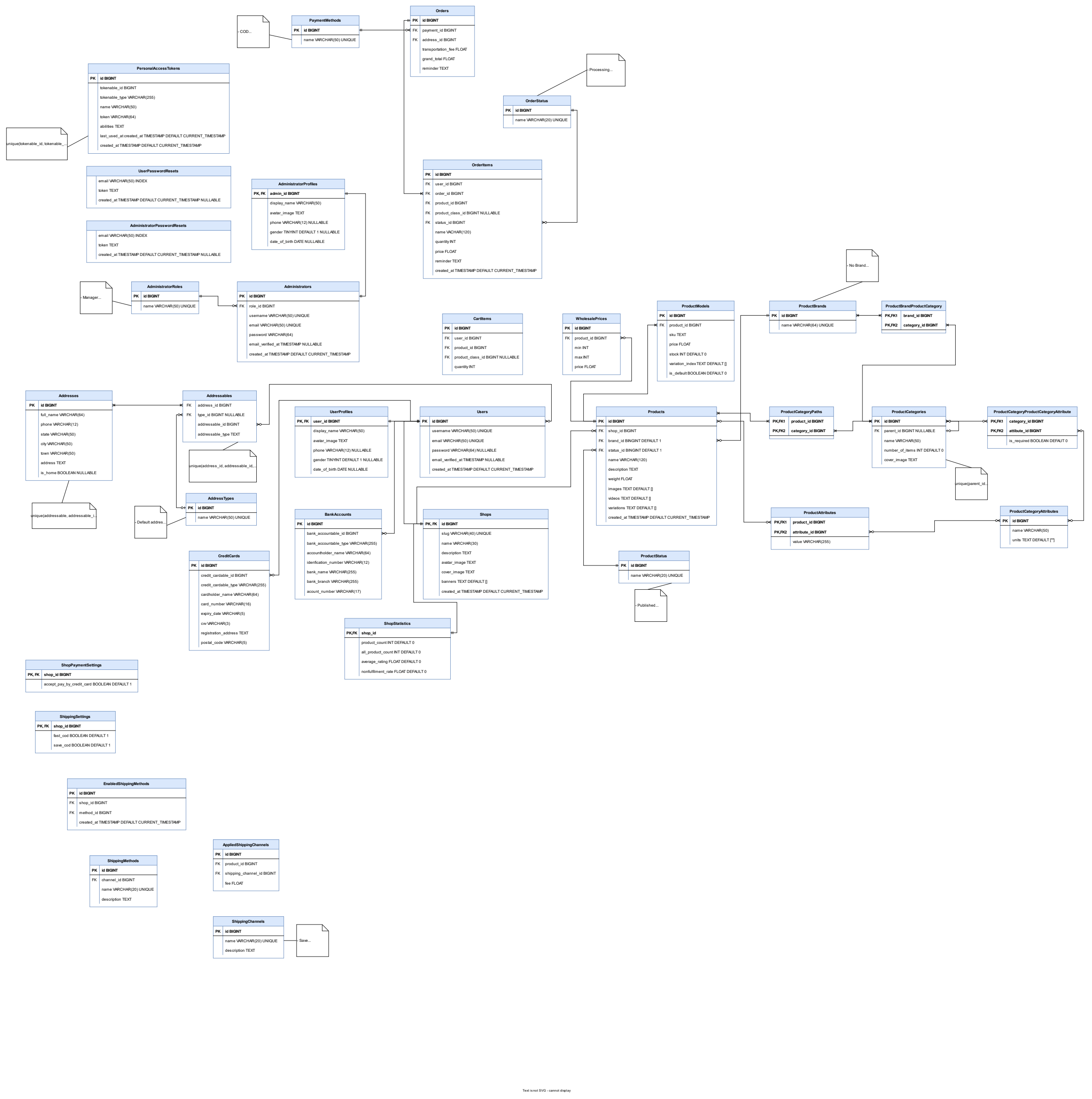
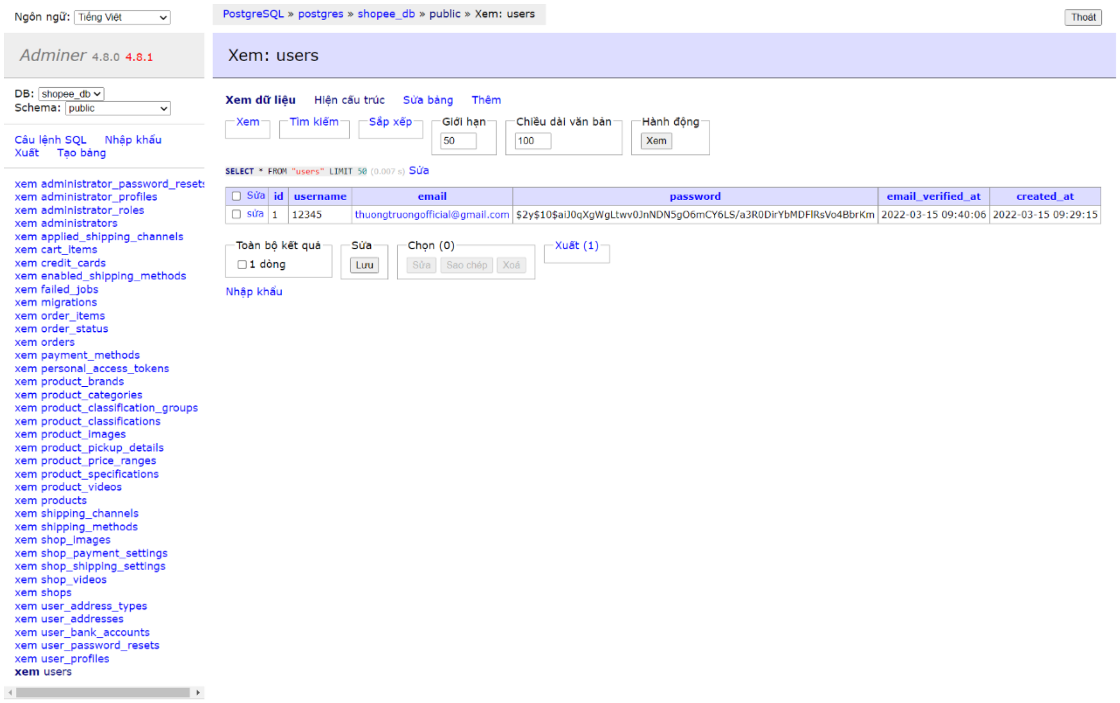


Figure 2.2 Database entities relationship diagram

III. IMPLEMENTATION ON POSTGRESQL

3.1 Create the database



Ngôn ngữ: Tiếng Việt

PostgreSQL » postgres » shopee_db » public » Xem: users

Thoát

Adminer 4.8.0 4.8.1

Xem: users

DB: shopee_db
Schema: public

Câu lệnh SQL Nhập khẩu
Xuất Tạo bảng

xem administrator_password_resets
xem administrator_profiles
xem administrator_roles
xem administrators
xem applied_shipping_channels
xem cart_items
xem credit_cards
xem enabled_shipping_methods
xem failed_jobs
xem migrations
xem order_items
xem order_status
xem orders
xem payment_methods
xem personal_access_tokens
xem product_brands
xem product_categories
xem product_classification_groups
xem product_classifications
xem product_images
xem product_pickup_details
xem product_price_ranges
xem product_specifications
xem product_videos
xem products
xem shipping_channels
xem shipping_methods
xem shop_images
xem shop_payment_settings
xem shop_shipping_settings
xem shop_videos
xem shops
xem user_address_types
xem user_addresses
xem user_bank_accounts
xem user_password_resets
xem user_profiles
xem users

Xem dữ liệu Hiện cấu trúc Sửa bảng Thêm

Xem | Tìm kiếm | Sắp xếp | Giới hạn: 50 | Chiều dài văn bản: 100 | Hành động: Xem

```
SELECT * FROM "users" LIMIT 50 (0.007 s) Sửa
```

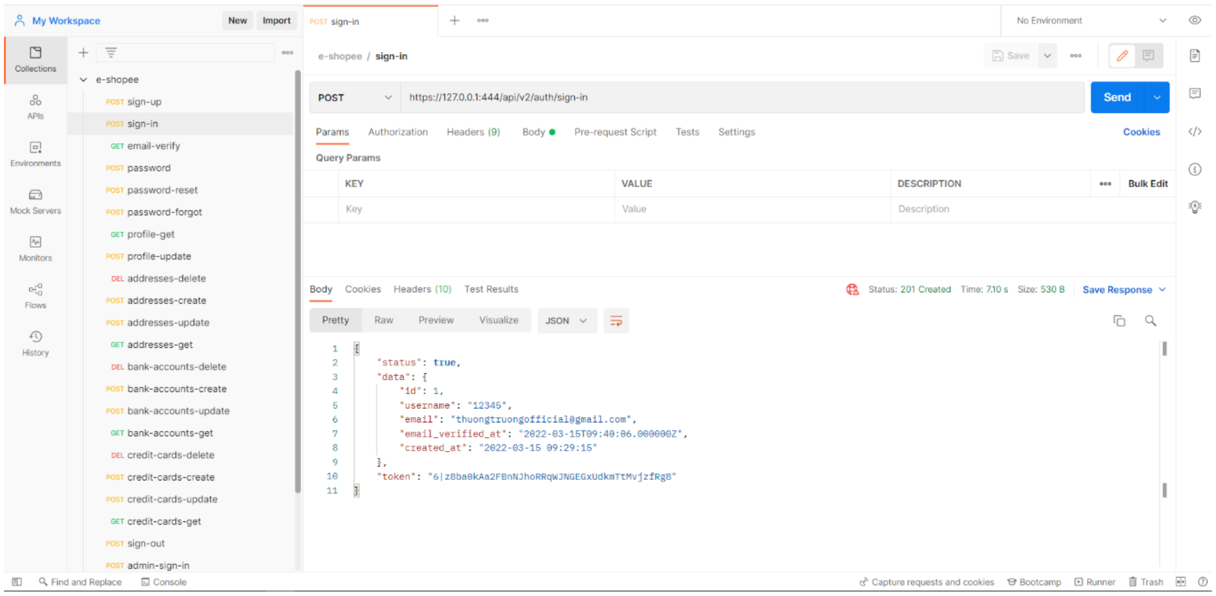
Sửa	id	username	email	password	email_verified_at	created_at
sửa	1	12345	thuongtruongofficial@gmail.com	$2y$10$aiJ0qXgWgLtwv0JnNDN5gO6mCY6LS/a3R0DirYbMDFlRsVo4BbrKm	2022-03-15 09:40:06	2022-03-15 09:29:15

Toàn bộ kết quả: 1 dòng | Sửa: Lưu | Chọn (0): Sửa Sao chép Xoá | Xuất (1)

Nhập khẩu

Figure 3.1 Database query in postgreDB playground

3.2 Input data & testing



My Workspace New Import

e-shopee
POST sign-up
POST sign-in
GET email-verify
POST password
POST password-reset
POST password-forgot
GET profile-get
POST profile-update
DEL addresses-delete
POST addresses-create
POST addresses-update
GET addresses-get
DEL bank-accounts-delete
POST bank-accounts-create
POST bank-accounts-update
GET bank-accounts-get
DEL credit-cards-delete
POST credit-cards-create
POST credit-cards-update
GET credit-cards-get
POST sign-out
POST admin-sign-in

e-shopee / sign-in

POST https://127.0.0.1:444/api/v2/auth/sign-in

Params Authorization Headers (9) Body Pre-request Script Tests Settings

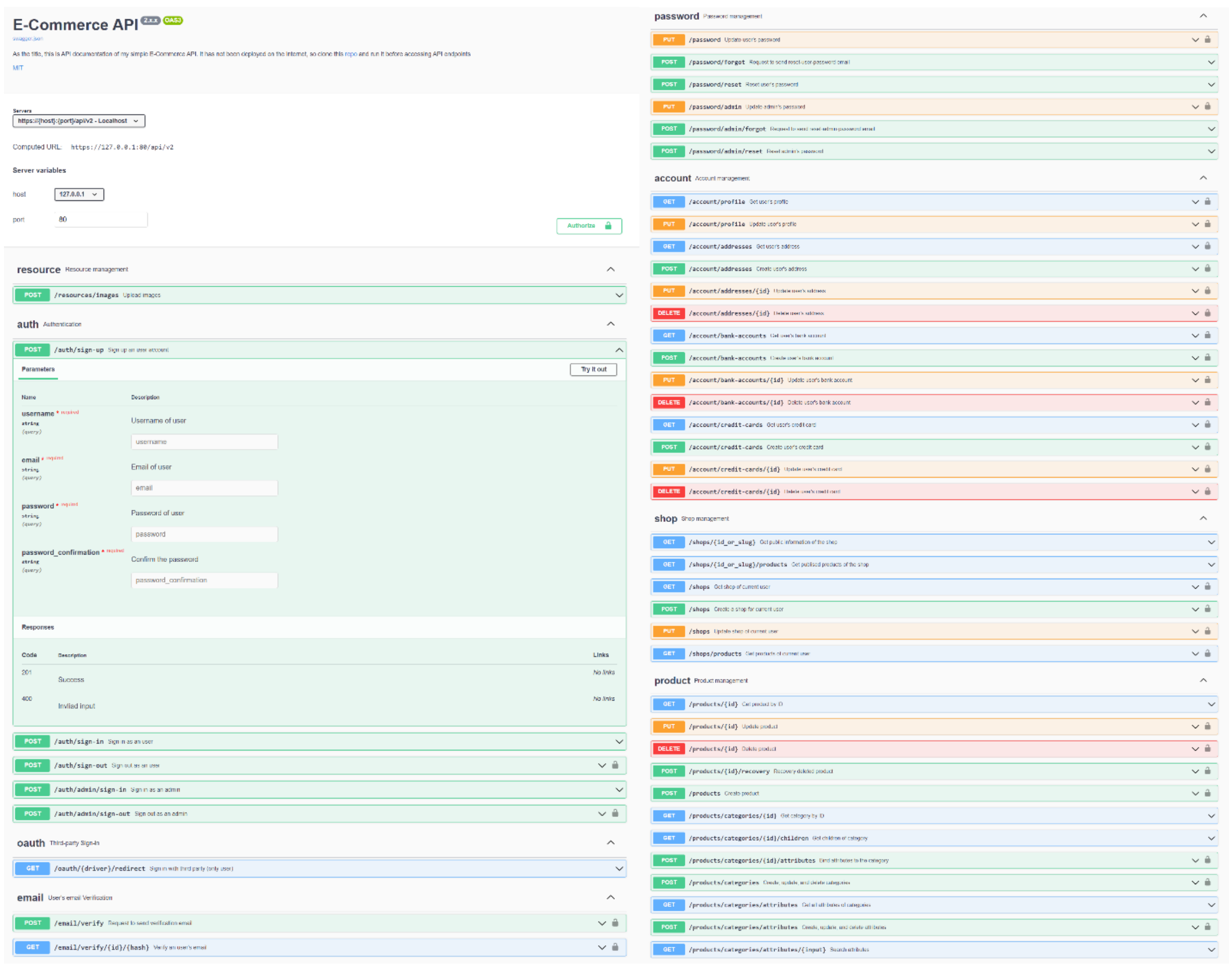
Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Cookies Headers (10) Test Results — Status: 201 Created Time: 7.10 s Size: 530 B

```
{
    "status": true,
    "data": {
        "id": 1,
        "username": "12345",
        "email": "thuongtruongofficial@gmail.com",
        "email_verified_at": "2022-03-15T09:40:06.000000Z",
        "created_at": "2022-03-15 09:29:15"
    },
    "token": "6|z8ba0kAa2FBnNJhoRRqWJNGEGxUdkmTtMvjzfRg8"
}
```

Figure 3.2a Testing API on Postman



E-Commerce API 2.x.x OAS3
swagger.json

As the title, this is API documentation of my simple E-Commerce API. It has not been deployed on the internet, so clone this repo and run it before accessing API endpoints

MIT

Servers
https://{host}:{port}/api/v2 - Localhost

Computed URL: `https://127.0.0.1:80/api/v2`

Server variables

host 127.0.0.1

port 80

resource Resource management
- POST `/resources/images` Upload images

auth Authentication
- POST `/auth/sign-up` Sign up an user account

Parameters

Name	Description
username * required string (query)	Username of user username
email * required string (query)	Email of user email
password * required string (query)	Password of user password
password_confirmation * required string (query)	Confirm the password password_confirmation

Responses

Code	Description	Links
201	Success	No links
400	Invlied input	No links

- POST `/auth/sign-in` Sign in as an user
- POST `/auth/sign-out` Sign out as an user
- POST `/auth/admin/sign-in` Sign in as an admin
- POST `/auth/admin/sign-out` Sign out as an admin

oauth Third-party Sign-In
- GET `/oauth/{driver}/redirect` Sign in with third party (only user)

email User's email Verification
- POST `/email/verify` Request to send verification email
- GET `/email/verify/{id}/{hash}` Verify an user's email

password Password management
- PUT `/password` Update user's password
- POST `/password/forgot` Request to send reset-user-password email
- POST `/password/reset` Reset user's password
- PUT `/password/admin` Update admin's password
- POST `/password/admin/forgot` Request to send reset-admin-password email
- POST `/password/admin/reset` Reset admin's password

account Account management
- GET `/account/profile` Get user's profile
- PUT `/account/profile` Update user's profile
- GET `/account/addresses` Get user's address
- POST `/account/addresses` Create user's address
- PUT `/account/addresses/{id}` Update user's address
- DELETE `/account/addresses/{id}` Delete user's address
- GET `/account/bank-accounts` Get user's bank account
- POST `/account/bank-accounts` Create user's bank account
- PUT `/account/bank-accounts/{id}` Update user's bank account
- DELETE `/account/bank-accounts/{id}` Delete user's bank account
- GET `/account/credit-cards` Get user's credit card
- POST `/account/credit-cards` Create user's credit card
- PUT `/account/credit-cards/{id}` Update user's credit card
- DELETE `/account/credit-cards/{id}` Delete user's credit card

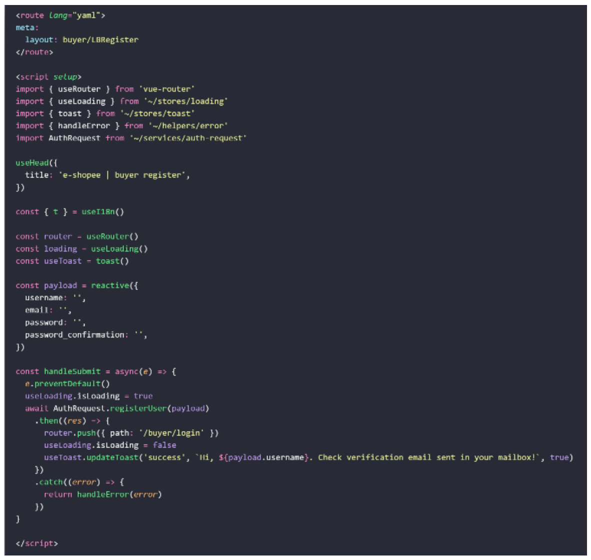
shop Shop management
- GET `/shops/{id_or_slug}` Get public information of the shop
- GET `/shops/{id_or_slug}/products` Get publised products of the shop
- GET `/shops` Get shop of current user
- POST `/shops` Create a shop for current user
- PUT `/shops` Update shop of current user
- GET `/shops/products` Get products of current user

product Product management
- GET `/products/{id}` Get product by ID
- PUT `/products/{id}` Update product
- DELETE `/products/{id}` Delete product
- POST `/products/{id}/recovery` Recovery deleted product
- POST `/products` Create product
- GET `/products/categories/{id}` Get category by ID
- GET `/products/categories/{id}/children` Get children of category
- POST `/products/categories/{id}/attributes` Bind attributes to the category
- POST `/products/categories` Create, update, and delete categories
- GET `/products/categories/attributes` Get all attributes of categories
- POST `/products/categories/attributes` Create, update, and delete attributes
- GET `/products/categories/attributes/{input}` Search attributes

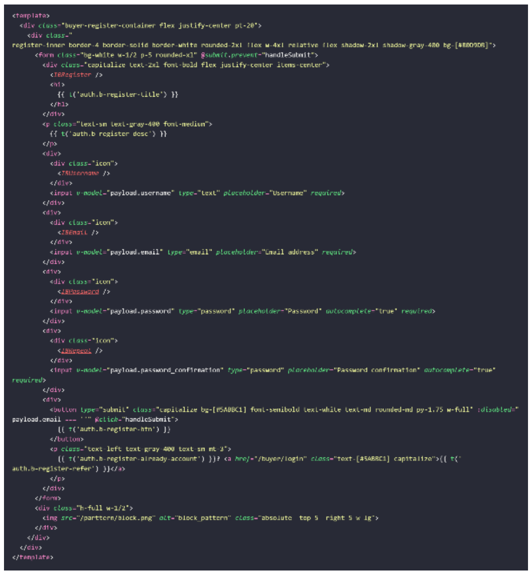
Figure 3.2b Testing API on Swagger

IV. CONNECT

4.1 Create interface *(this is sample, see more in source code)*

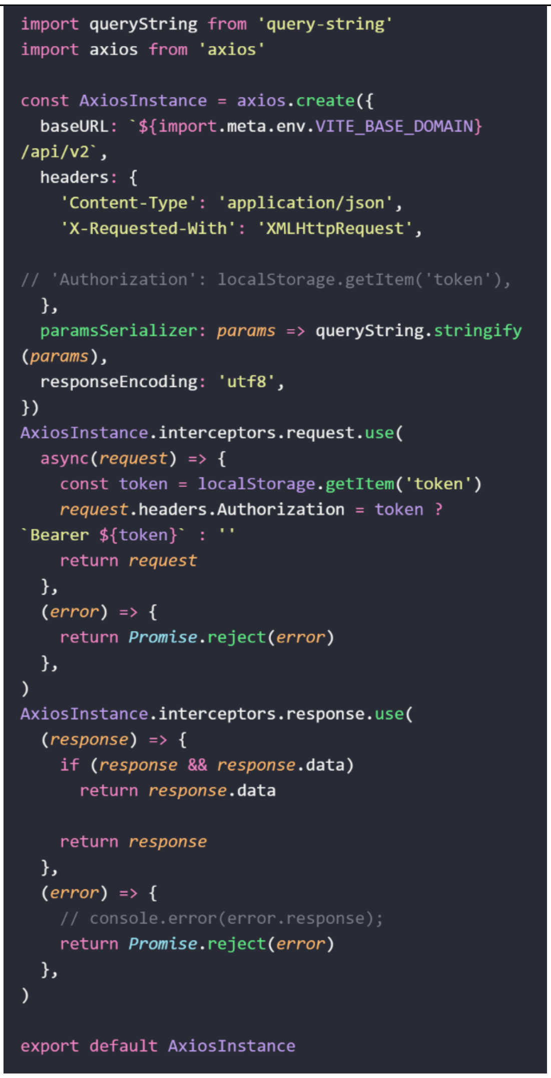


```
<route lang="yaml">
meta:
  layout: buyer/LBRegister
</route>

<script setup>
import { useRouter } from 'vue-router'
import { useLoading } from '~/stores/loading'
import { toast } from '~/stores/toast'
import { handleError } from '~/helpers/error'
import AuthRequest from '~/services/auth-request'

useHead({
  title: 'e-shopee | buyer register',
})

const { t } = useI18n()

const router = useRouter()
const loading = useLoading()
const useToast = toast()

const payload = reactive({
  username: '',
  email: '',
  password: '',
  password_confirmation: '',
})

const handleSubmit = async(e) => {
  e.preventDefault()
  useLoading.isLoading = true
  await AuthRequest.registerUser(payload)
    .then((res) => {
      router.push({ path: '/buyer/login' })
      useLoading.isLoading = false
      useToast.updateToast('success', `Hi, ${payload.username}. Check verification email sent in your mailbox!`, true)
    })
    .catch((error) => {
      return handleError(error)
    })
}

</script>
```



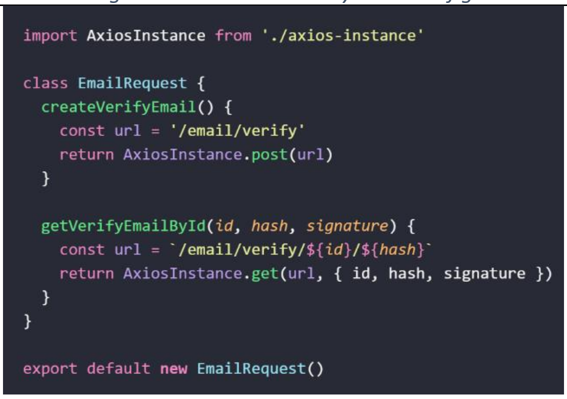
```
<template>
  <div class="buyer-register-container flex justify-center pt-20">
    <div class="
register-inner border-4 border-solid border-white rounded-2xl flex w-4xl relative flex shadow-2xl shadow-gray-400 bg-[#B8D9D8]">
      <form class="bg-white w-1/2 p-5 rounded-xl" @submit.prevent="handleSubmit">
        <div class="capitalize text-2xl font-bold flex justify-center items-center">
          <I8Register />
          <h1>
            {{ t('auth.b-register-title') }}
          </h1>
        </div>
        <p class="text-sm text-gray-400 font-medium">
          {{ t('auth.b register desc') }}
        </p>
        <div>
          <div class="icon">
            <I8Username />
          </div>
          <input v-model="payload.username" type="text" placeholder="Username" required>
        </div>
        <div>
          <div class="icon">
            <I8Email />
          </div>
          <input v-model="payload.email" type="email" placeholder="Email address" required>
        </div>
        <div>
          <div class="icon">
            <I8Password />
          </div>
          <input v-model="payload.password" type="password" placeholder="Password" autocomplete="true" required>
        </div>
        <div>
          <div class="icon">
            <I8Repeat />
          </div>
          <input v-model="payload.password_confirmation" type="password" placeholder="Password confirmation" autocomplete="true"
required>
        </div>
        <div>
          <button type="submit" class="capitalize bg-[#5ABBC1] font-semibold text-white text-md rounded-md py-1.75 w-full" :disabled="
payload.email === ''" @click="handleSubmit">
            {{ t('auth.b-register-btn') }}
          </button>
          <p class="text-left text-gray-400 text-sm mt-3">
            {{ t('auth.b-register-already-account') }}? <a href="/buyer/login" class="text-[#5ABBC1] capitalize">{{ t('
auth.b-register-refer') }}</a>
          </p>
        </div>
      </form>
      <div class="h-full w-1/2">
        <img src="/parttern/block.png" alt="block_pattern" class="absolute  top 5  right 5 w lg">
      </div>
    </div>
  </div>
</template>
```

4.2 Connect API by Axios



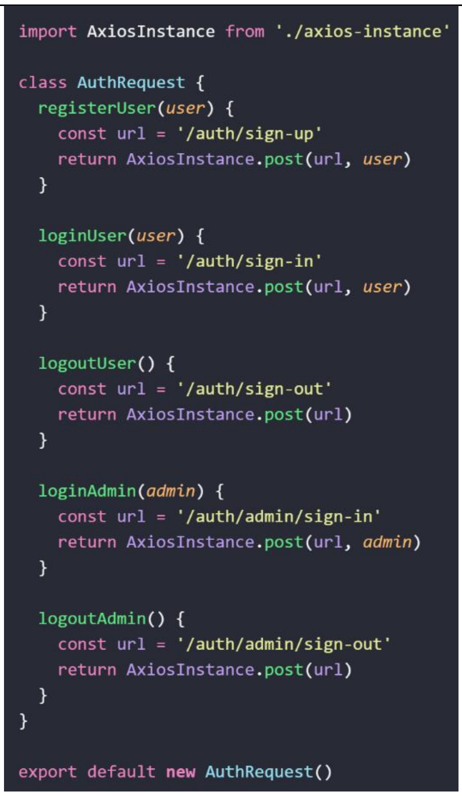
```
import queryString from 'query-string'
import axios from 'axios'

const AxiosInstance = axios.create({
  baseURL: `${import.meta.env.VITE_BASE_DOMAIN}
/api/v2`,
  headers: {
    'Content-Type': 'application/json',
    'X-Requested-With': 'XMLHttpRequest',

// 'Authorization': localStorage.getItem('token'),
  },
  paramsSerializer: params => queryString.stringify
(params),
  responseEncoding: 'utf8',
})
AxiosInstance.interceptors.request.use(
  async(request) => {
    const token = localStorage.getItem('token')
    request.headers.Authorization = token ?
`Bearer ${token}` : ''
    return request
  },
  (error) => {
    return Promise.reject(error)
  },
)
AxiosInstance.interceptors.response.use(
  (response) => {
    if (response && response.data)
      return response.data

    return response
  },
  (error) => {
    // console.error(error.response);
    return Promise.reject(error)
  },
)

export default AxiosInstance
```

Figure 4.2a Connect API by axios config



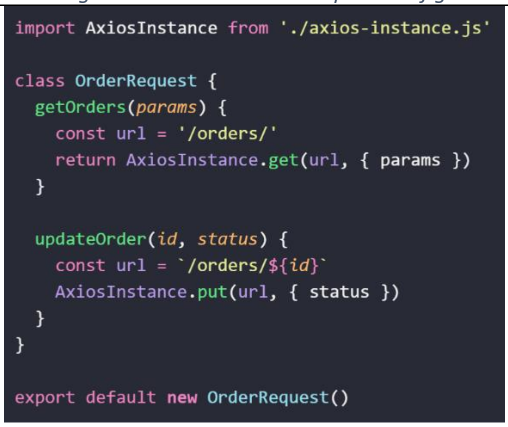
```
import AxiosInstance from './axios-instance'

class EmailRequest {
  createVerifyEmail() {
    const url = '/email/verify'
    return AxiosInstance.post(url)
  }

  getVerifyEmailById(id, hash, signature) {
    const url = `/email/verify/${id}/${hash}`
    return AxiosInstance.get(url, { id, hash, signature })
  }
}

export default new EmailRequest()
```

Figure 4.2b Email request config



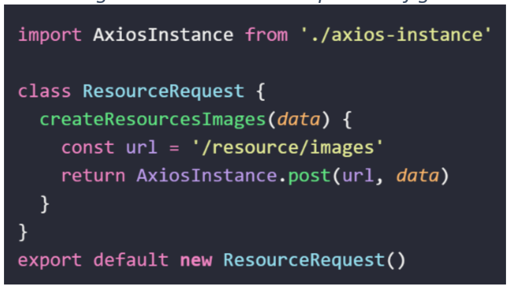
```
import AxiosInstance from './axios-instance'

class AuthRequest {
  registerUser(user) {
    const url = '/auth/sign-up'
    return AxiosInstance.post(url, user)
  }

  loginUser(user) {
    const url = '/auth/sign-in'
    return AxiosInstance.post(url, user)
  }

  logoutUser() {
    const url = '/auth/sign-out'
    return AxiosInstance.post(url)
  }

  loginAdmin(admin) {
    const url = '/auth/admin/sign-in'
    return AxiosInstance.post(url, admin)
  }

  logoutAdmin() {
    const url = '/auth/admin/sign-out'
    return AxiosInstance.post(url)
  }
}

export default new AuthRequest()
```

Figure 4.2d Authenticate request config



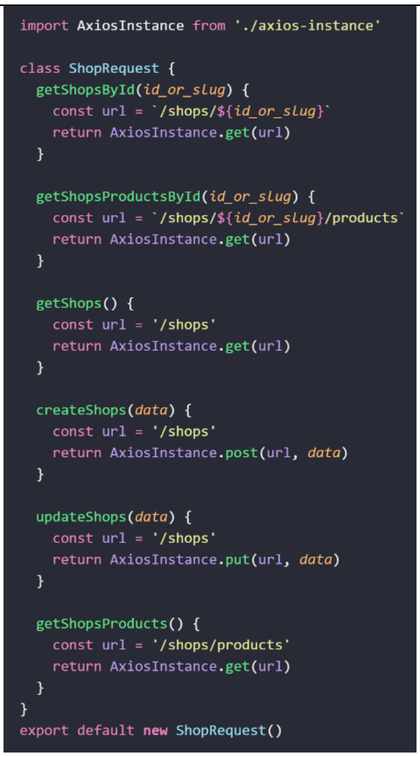
```
import AxiosInstance from './axios-instance.js'

class OrderRequest {
  getOrders(params) {
    const url = '/orders/'
    return AxiosInstance.get(url, { params })
  }

  updateOrder(id, status) {
    const url = `/orders/${id}`
    AxiosInstance.put(url, { status })
  }
}

export default new OrderRequest()
```

Figure 4.2e Password request config



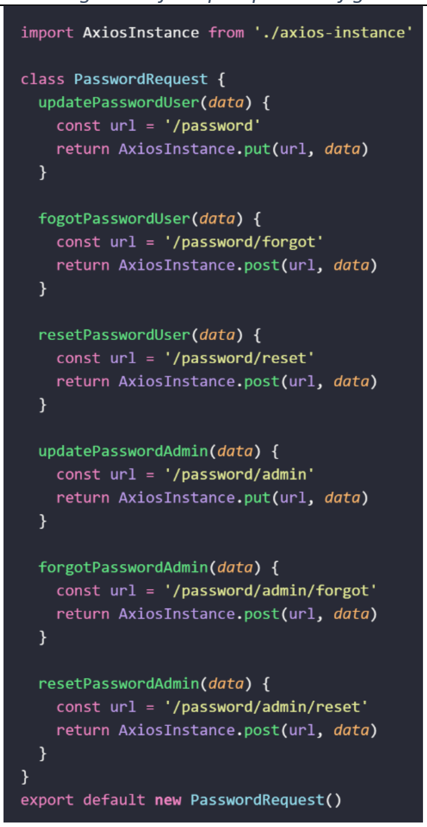
```
import AxiosInstance from './axios-instance'

class ResourceRequest {
  createResourcesImages(data) {
    const url = '/resource/images'
    return AxiosInstance.post(url, data)
  }
}
export default new ResourceRequest()
```

Figure 4.2c Resource request



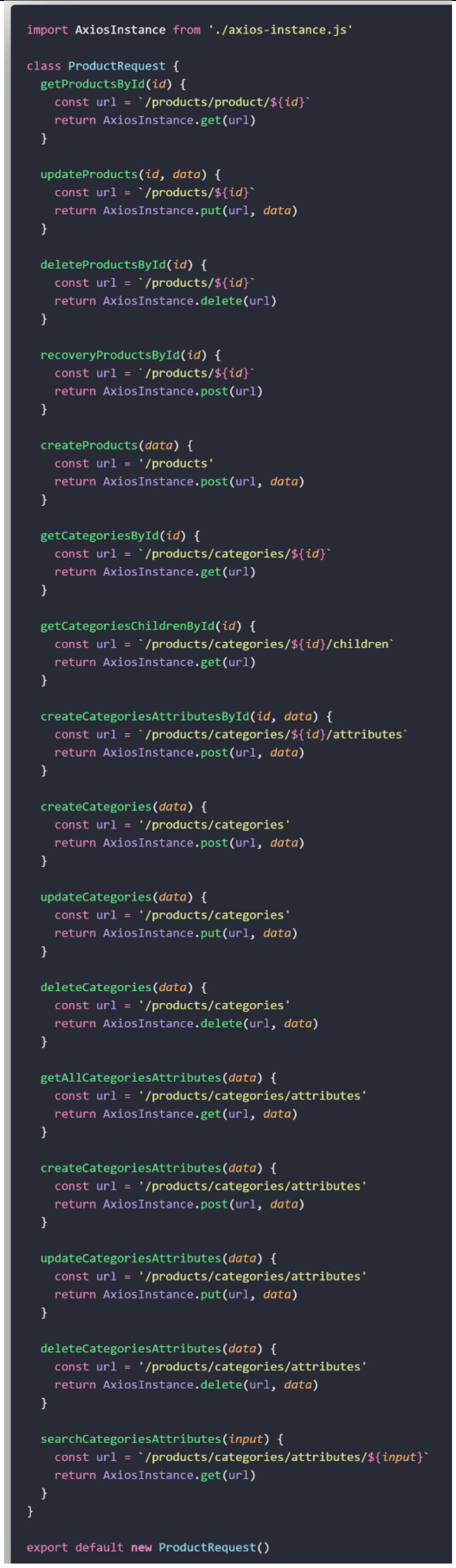
```
import AxiosInstance from './axios-instance'

class ShopRequest {
  getShopsById(id_or_slug) {
    const url = `/shops/${id_or_slug}`
    return AxiosInstance.get(url)
  }

  getShopsProductsById(id_or_slug) {
    const url = `/shops/${id_or_slug}/products`
    return AxiosInstance.get(url)
  }

  getShops() {
    const url = '/shops'
    return AxiosInstance.get(url)
  }

  createShops(data) {
    const url = '/shops'
    return AxiosInstance.post(url, data)
  }

  updateShops(data) {
    const url = '/shops'
    return AxiosInstance.put(url, data)
  }

  getShopsProducts() {
    const url = '/shops/products'
    return AxiosInstance.get(url)
  }
}
export default new ShopRequest()
```

Figure 4.2f Shop request config



```
import AxiosInstance from './axios-instance'

class PasswordRequest {
  updatePasswordUser(data) {
    const url = '/password'
    return AxiosInstance.put(url, data)
  }

  fogotPasswordUser(data) {
    const url = '/password/forgot'
    return AxiosInstance.post(url, data)
  }

  resetPasswordUser(data) {
    const url = '/password/reset'
    return AxiosInstance.post(url, data)
  }

  updatePasswordAdmin(data) {
    const url = '/password/admin'
    return AxiosInstance.put(url, data)
  }

  forgotPasswordAdmin(data) {
    const url = '/password/admin/forgot'
    return AxiosInstance.post(url, data)
  }

  resetPasswordAdmin(data) {
    const url = '/password/admin/reset'
    return AxiosInstance.post(url, data)
  }
}
export default new PasswordRequest()
```

Figure 3.1 Password request config



```
import AxiosInstance from './axios-instance.js'

class ProductRequest {
  getProductsById(id) {
    const url = `/products/product/${id}`
    return AxiosInstance.get(url)
  }

  updateProducts(id, data) {
    const url = `/products/${id}`
    return AxiosInstance.put(url, data)
  }

  deleteProductsById(id) {
    const url = `/products/${id}`
    return AxiosInstance.delete(url)
  }

  recoveryProductsById(id) {
    const url = `/products/${id}`
    return AxiosInstance.post(url)
  }

  createProducts(data) {
    const url = '/products'
    return AxiosInstance.post(url, data)
  }

  getCategoriesById(id) {
    const url = `/products/categories/${id}`
    return AxiosInstance.get(url)
  }

  getCategoriesChildrenById(id) {
    const url = `/products/categories/${id}/children`
    return AxiosInstance.get(url)
  }

  createCategoriesAttributesById(id, data) {
    const url = `/products/categories/${id}/attributes`
    return AxiosInstance.post(url, data)
  }

  createCategories(data) {
    const url = '/products/categories'
    return AxiosInstance.post(url, data)
  }

  updateCategories(data) {
    const url = '/products/categories'
    return AxiosInstance.put(url, data)
  }

  deleteCategories(data) {
    const url = '/products/categories'
    return AxiosInstance.delete(url, data)
  }

  getAllCategoriesAttributes(data) {
    const url = '/products/categories/attributes'
    return AxiosInstance.get(url, data)
  }

  createCategoriesAttributes(data) {
    const url = '/products/categories/attributes'
    return AxiosInstance.post(url, data)
  }

  updateCategoriesAttributes(data) {
    const url = '/products/categories/attributes'
    return AxiosInstance.put(url, data)
  }

  deleteCategoriesAttributes(data) {
    const url = '/products/categories/attributes'
    return AxiosInstance.delete(url, data)
  }

  searchCategoriesAttributes(input) {
    const url = `/products/categories/attributes/${input}`
    return AxiosInstance.get(url)
  }
}

export default new ProductRequest()
```

Figure 3.1 Database quey playground 1

V. PRODUCTION

5.1 Overview

I. Buyer interface

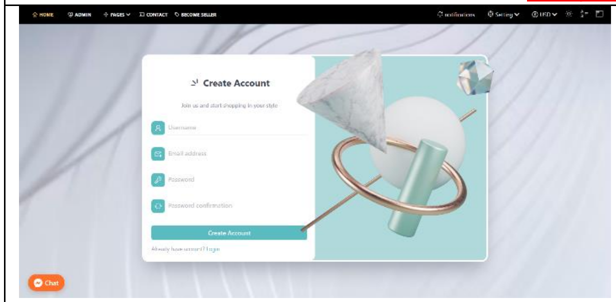


Figure 5.1.1a Register page for buyer

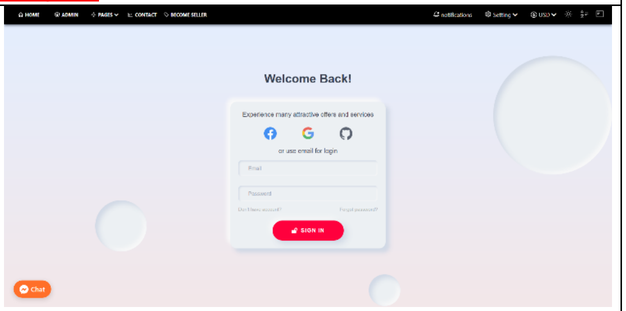


Figure 5.1.1b Login page for buyer

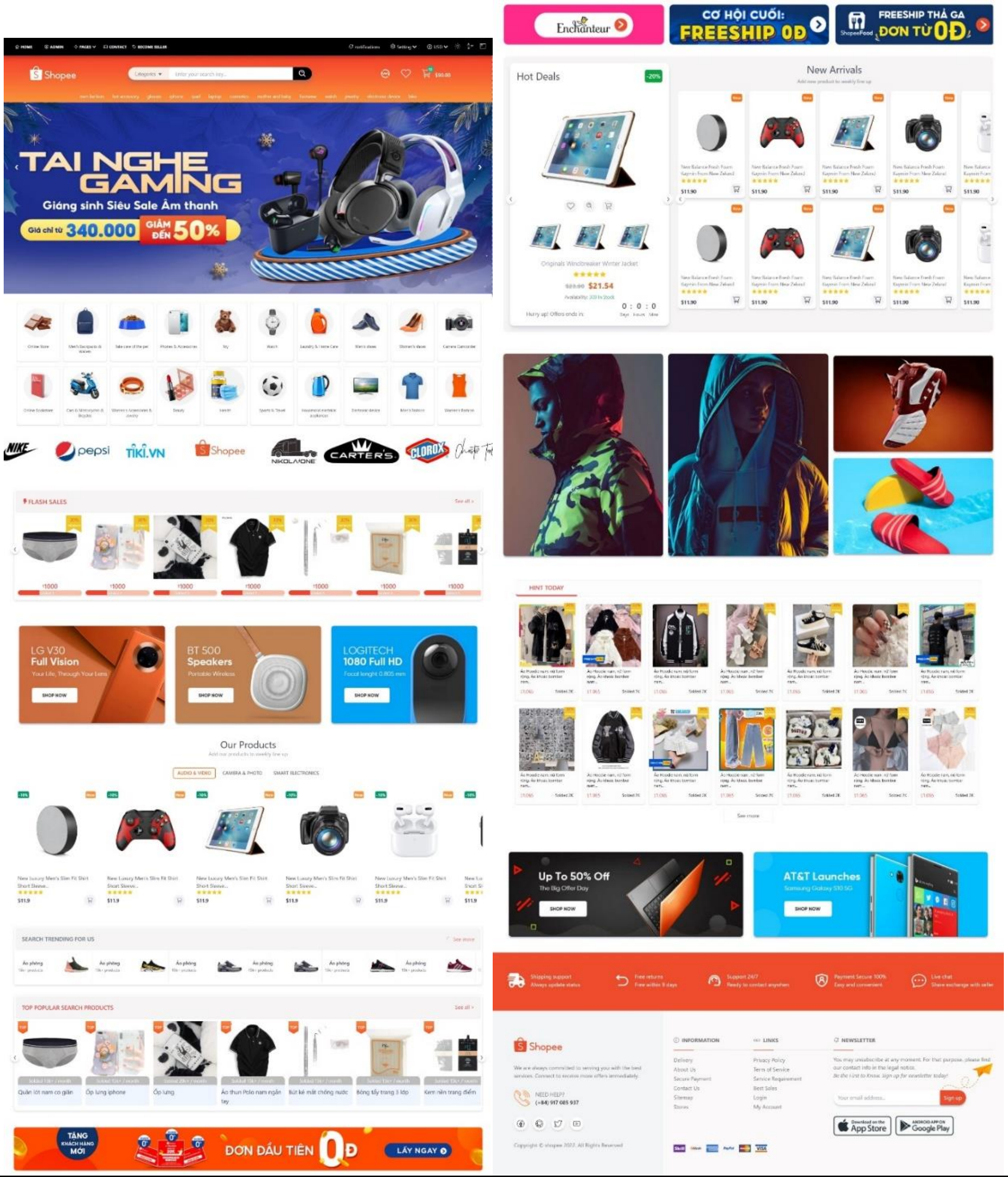
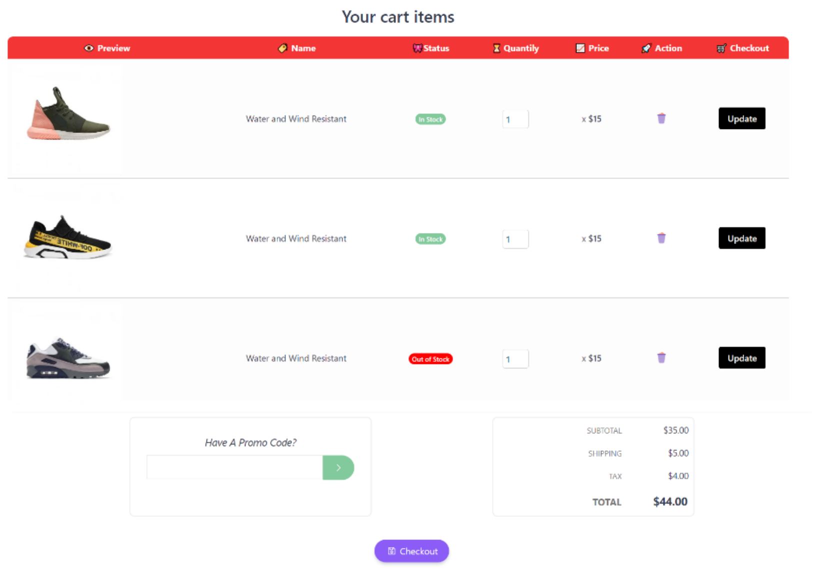


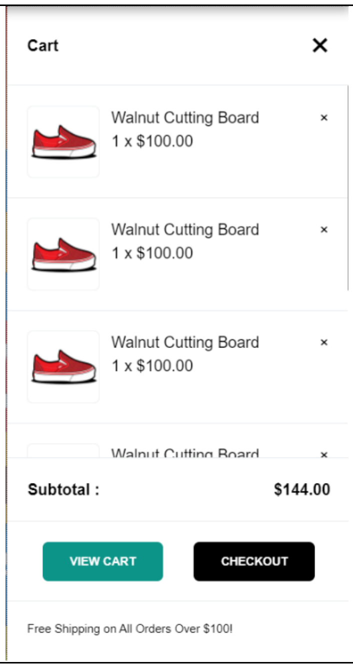
Figure 5.1.1c Home page for buyer


Your cart items

Preview	Name	Status	Quantity	Price	Action	Checkout
	Water and Wind Resistant	In Stock	1	x $15		Update
	Water and Wind Resistant	In Stock	1	x $15		Update
	Water and Wind Resistant	Out of Stock	1	x $15		Update

Have A Promo Code?

SUBTOTAL	$35.00
SHIPPING	$5.00
TAX	$4.00
TOTAL	$44.00


Cart

Walnut Cutting Board
1 x $100.00

Walnut Cutting Board
1 x $100.00

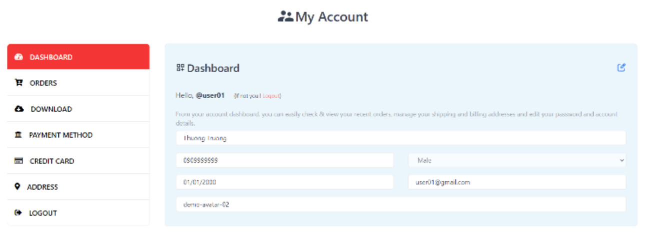
Walnut Cutting Board
1 x $100.00

Walnut Cutting Board

Subtotal : $144.00

Free Shipping on All Orders Over $100!

Figure 5.1.1d&e Cart page details and cart popup preview


My Account

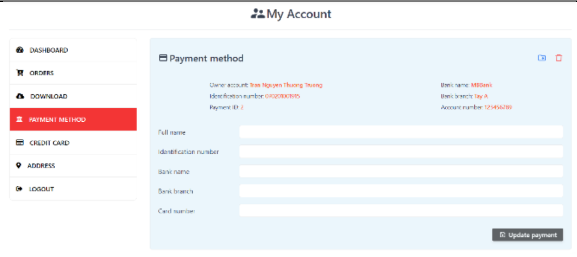
Dashboard

Hello, **@user01** (If not you ! Logout)

From your account dashboard. you can easily check & view your recent orders, manage your shipping and billing addresses and edit your password and account details.

Thuong Truong
0909999999 | Male
01/01/2000 | user01@gmail.com
demo-avatar-02

Figure 5.1.1f Account profile dashboard

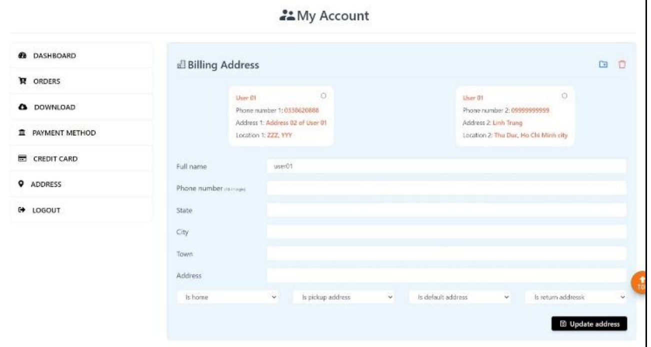

My Account

Payment method

Owner account: Tran Nguyen Thuong Truong
Identification number: 070201001915
Payment ID: 2
Bank name: MBBank
Bank branch: Tay A
Account number: 123456789

Full name
Identification number
Bank name
Bank branch
Card number

Figure 5.1.1g Account payment method


My Account

Billing Address

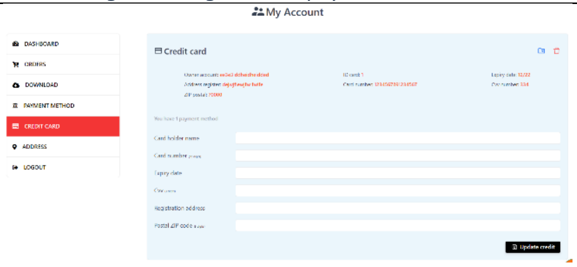
User 01
Phone number 1: 0338620888
Address 1: Address 02 of User 01
Location 1: ZZZ, YYY

User 01
Phone number 2: 09999999999
Address 2: Linh Trung
Location 2: Thu Duc, Ho Chi Minh city

Full name: user01
Phone number
State
City
Town
Address

Is home | Is pickup address | Is default address | Is return address

Figure 5.1.1h Account address details


My Account

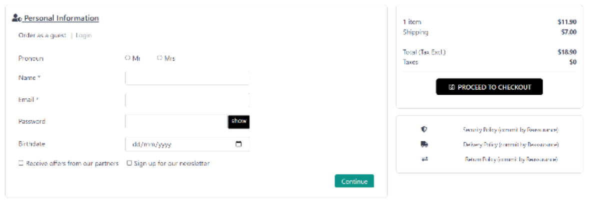
Credit card

Owner account: ee0e3 ddhsidhsidded
Address register: dejwjfiewjfw fwife
ZIP postal: 70000
ID card: 1
Card number: 1234567891234567
Expiry date: 12/22
Cvv number: 334

You have 1 payment method

Card holder name
Card number
Expiry date
Cvv
Registration address
Postal ZIP code

Figure 5.1.1i Account credit card details


Personal Information

Order as a guest | Login

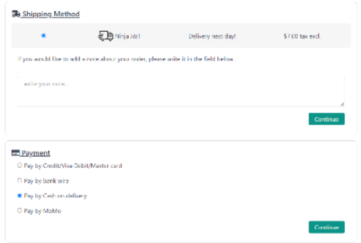
Pronoun ○ Mr ○ Mrs
Name *
Email *
Password
Birthdate dd/mm/yyyy

☐ Receive offers from our partners ☐ Sign up for our newsletter

1 item	$11.90
Shipping	$7.00
Total (Tax Excl.)	$18.90
Taxes	$0

Security Policy (commit by Reassurance)
Delivery Policy (commit by Reassurance)
Return Policy (commit by Reassurance)

Figure 5.1.1j Checkout page


Shipping Method

● Ninja J&T | Delivery next day! | $7.00 tax excl.

If you would like to add a note about your order, please write it in the field below.

write your note..

Payment

○ Pay by Credit/Visa Debit/Master card
○ Pay by bank wire
[✓ Pay by Cash on delivery]
○ Pay by MoMo

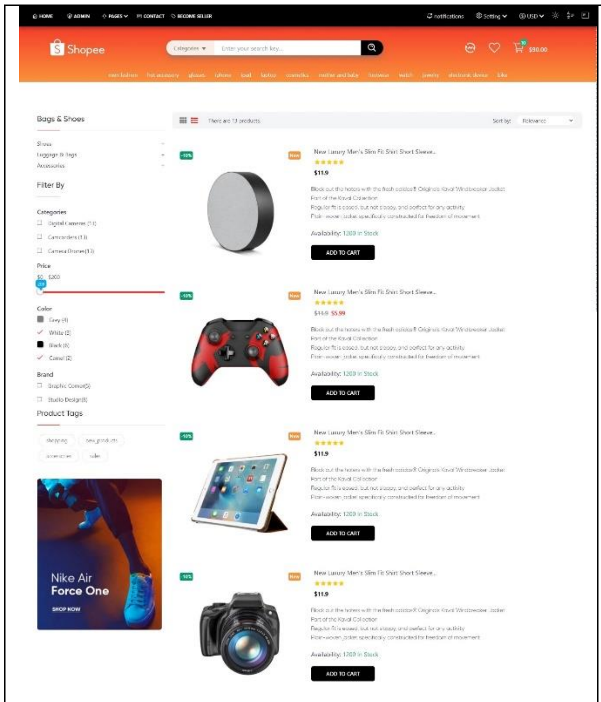


Figure 5.1.1k Filter page in flow mode

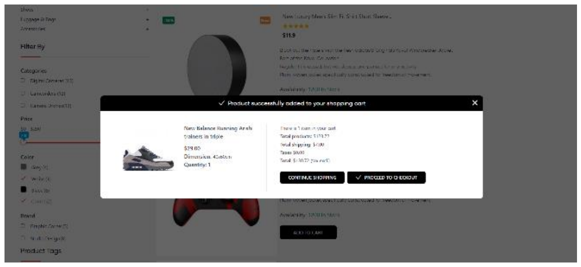


Figure 5.1.1m Add to cart popup

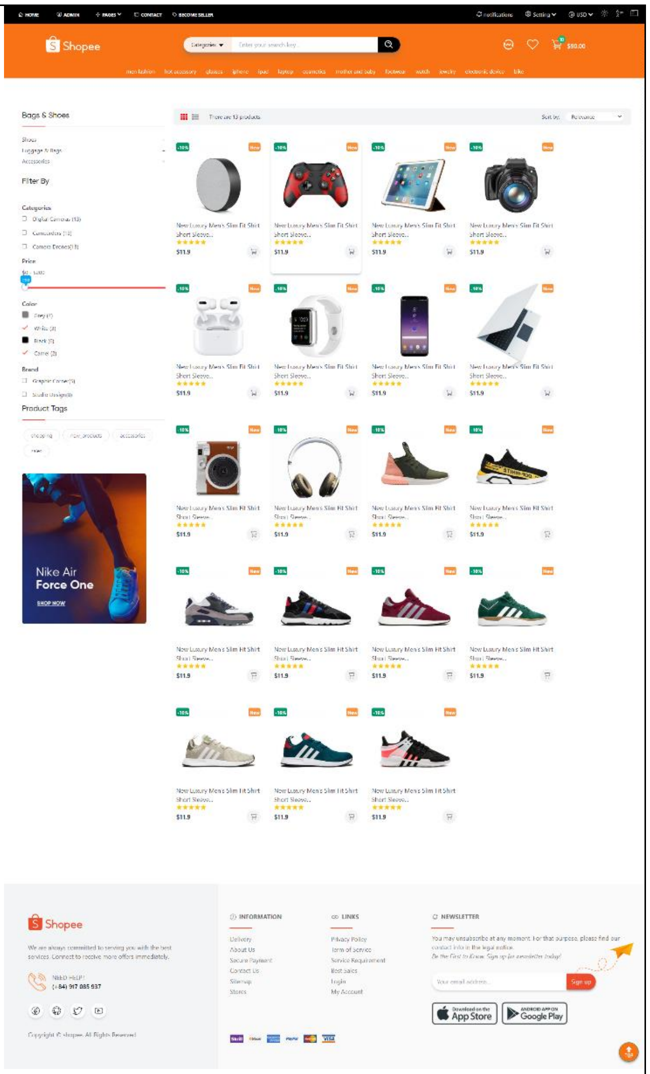


Figure 5.1.1l Filter page in grid mode

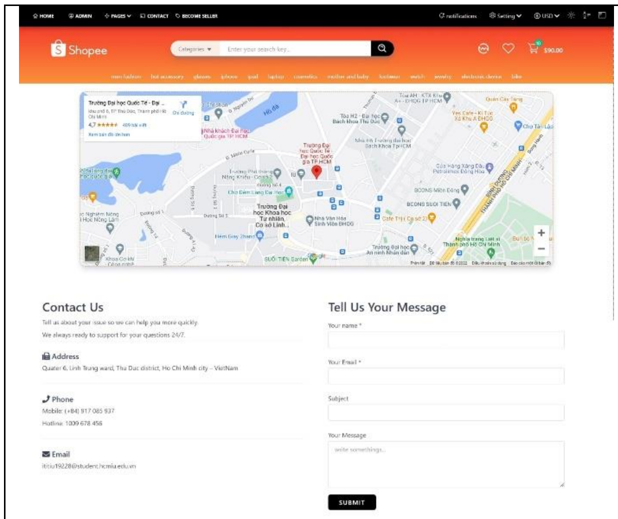


Figure 5.1.1m Contact page

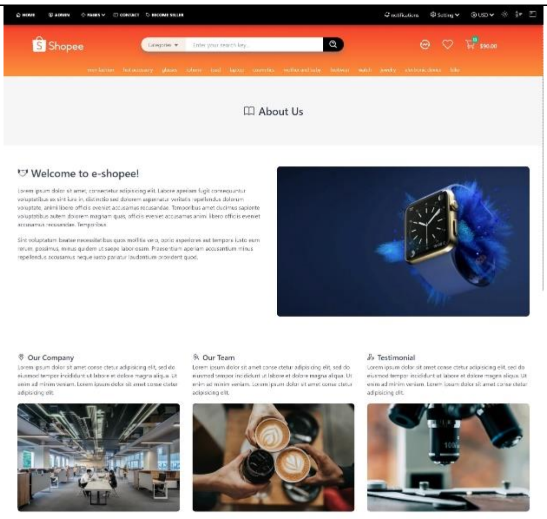
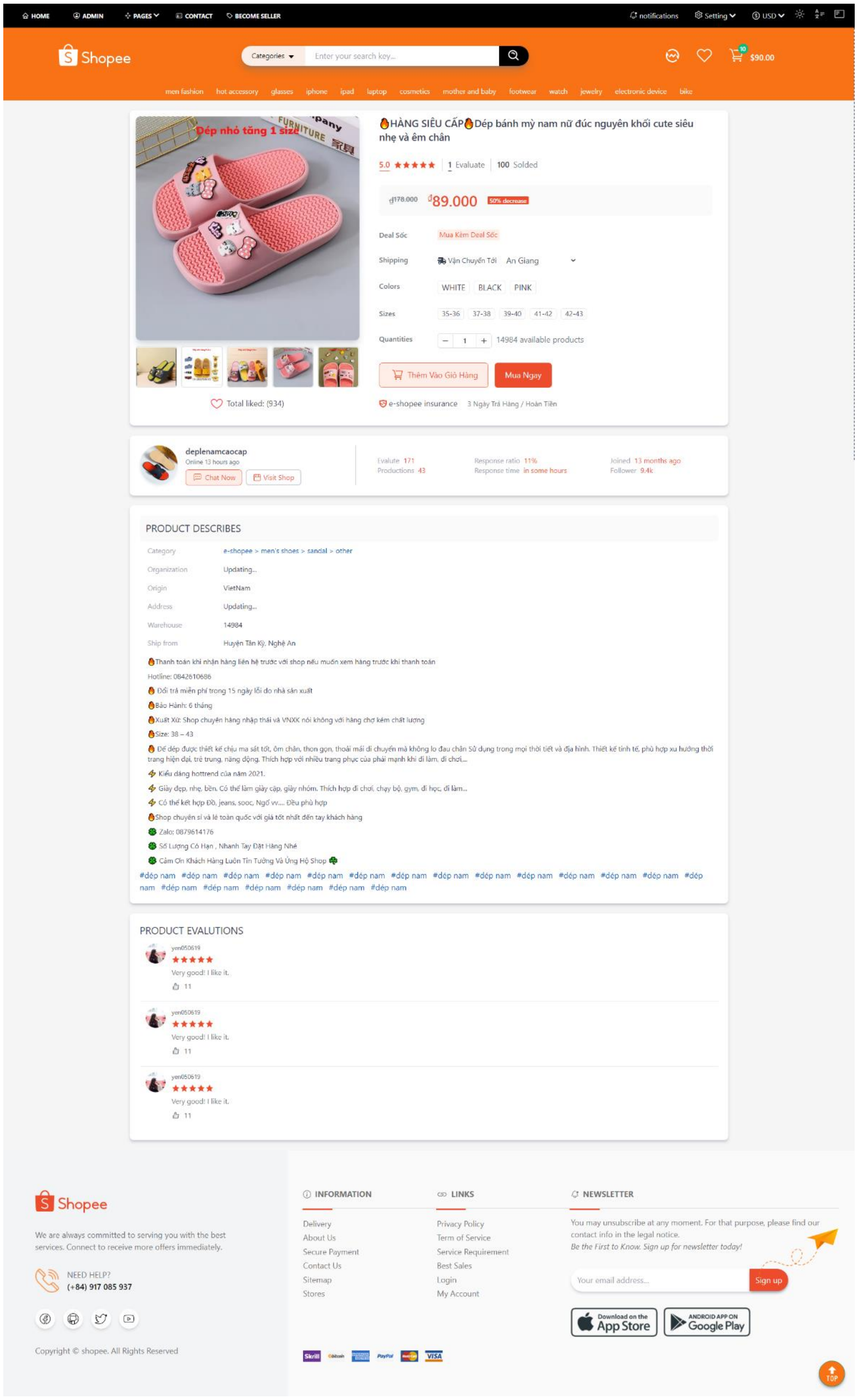


Figure 5.1.1n About page



🔥HÀNG SIÊU CẤP🔥Dép bánh mỳ nam nữ đúc nguyên khối cute siêu nhẹ và êm chân

5.0 ★★★★★ | 1 Evaluate | 100 Solded

₫178.000 ₫89.000 50% decrease

Deal Sốc	Mua Kèm Deal Sốc
Shipping	Vận Chuyển Tới An Giang
Colors	WHITE BLACK PINK
Sizes	35-36 37-38 39-40 41-42 42-43
Quantities	1 14984 available products

Total liked: (934)

e-shopee insurance 3 Ngày Trả Hàng / Hoàn Tiền

deplenamcaocap
Online 13 hours ago

Evalute 171
Productions 43
Response ratio 11%
Response time in some hours
Joined 13 months ago
Follower 9.4k

PRODUCT DESCRIBES

Category	e-shopee > men's shoes > sandal > other
Organization	Updating...
Origin	VietNam
Address	Updating...
Warehouse	14984
Ship from	Huyện Tân Kỳ, Nghệ An

🔥Thanh toán khi nhận hàng liên hệ trước với shop nếu muốn xem hàng trước khi thanh toán

Hotline: 0842610686

🔥 Đổi trả miễn phí trong 15 ngày lỗi do nhà sản xuất

🔥Bảo Hành: 6 tháng

🔥Xuất Xứ: Shop chuyên hàng nhập thái và VNXK nói không với hàng chợ kém chất lượng

🔥Size: 38 – 43

🔥 Đế dép được thiết kế chịu ma sát tốt, ôm chân, thon gọn, thoải mái di chuyển mà không lo đau chân Sử dụng trong mọi thời tiết và địa hình. Thiết kế tinh tế, phù hợp xu hướng thời trang hiện đại, trẻ trung, năng động. Thích hợp với nhiều trang phục của phái mạnh khi đi làm, đi chơi,...

⚡ Kiểu dáng hottrend của năm 2021.

⚡ Giày đẹp, nhẹ, bền. Có thể làm giày cặp, giày nhóm. Thích hợp đi chơi, chạy bộ, gym, đi học, đi làm...

⚡ Có thể kết hợp Đồ, jeans, sooc, Ngố vv.... Đều phù hợp

🔥Shop chuyên sỉ và lẻ toàn quốc với giá tốt nhất đến tay khách hàng

🍀 Zalo: 0879614176

🍀 Số Lượng Có Hạn , Nhanh Tay Đặt Hàng Nhé

🍀 Cảm Ơn Khách Hàng Luôn Tin Tưởng Và Ủng Hộ Shop 🍀

#dép nam #dép nam #dép nam #dép nam #dép nam #dép nam #dép nam #dép nam #dép nam #dép nam #dép nam #dép nam #dép nam #dép nam #dép nam #dép nam #dép nam #dép nam #dép nam #dép nam

PRODUCT EVALUTIONS

yen050619
★★★★★
Very good! I like it.
11

yen050619
★★★★★
Very good! I like it.
11

yen050619
★★★★★
Very good! I like it.
11

We are always committed to serving you with the best services. Connect to receive more offers immediately.

NEED HELP?
(+84) 917 085 937

Copyright © shopee. All Rights Reserved

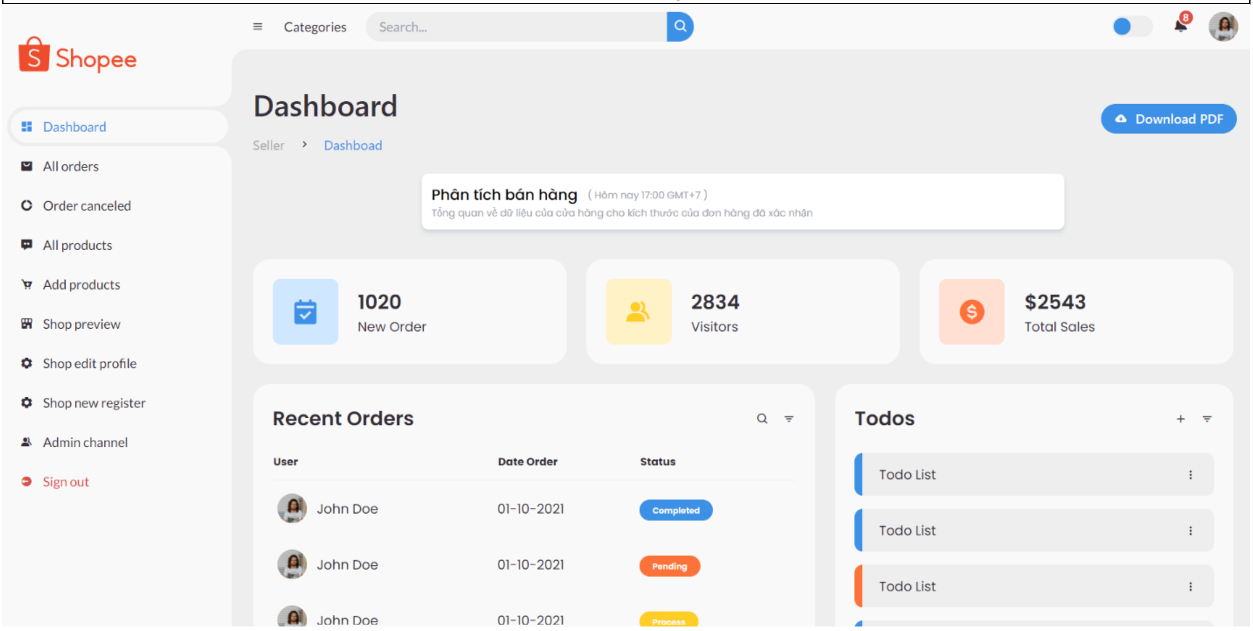
INFORMATION
Delivery
About Us
Secure Payment
Contact Us
Sitemap
Stores

LINKS
Privacy Policy
Term of Service
Service Requirement
Best Sales
Login
My Account

NEWSLETTER
You may unsubscribe at any moment. For that purpose, please find our contact info in the legal notice.
Be the First to Know. Sign up for newsletter today!
Your email address...

Figure 5.1.1o Product details page

II. Seller interface



Shopee

Categories | Search...

Dashboard

Download PDF

Seller > Dashboard

- Dashboard
- All orders
- Order canceled
- All products
- Add products
- Shop preview
- Shop edit profile
- Shop new register
- Admin channel
- Sign out

Phân tích bán hàng (Hôm nay 17:00 GMT+7)
Tổng quan về dữ liệu của cửa hàng cho kích thước của đơn hàng đã xác nhận

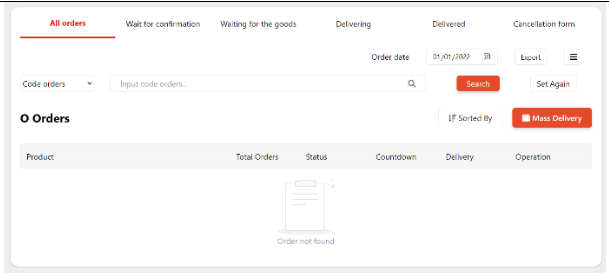
1020 New Order | 2834 Visitors | $2543 Total Sales

Recent Orders

User	Date Order	Status
John Doe	01-10-2021	Completed
John Doe	01-10-2021	Pending
John Doe	01-10-2021	Process

Todos
- Todo List
- Todo List
- Todo List

Figure 5.1.2a Seller dashboard page



All orders | Wait for confirmation | Waiting for the goods | Delivering | Delivered | Cancellation form

Order date 01/01/2022 | Export

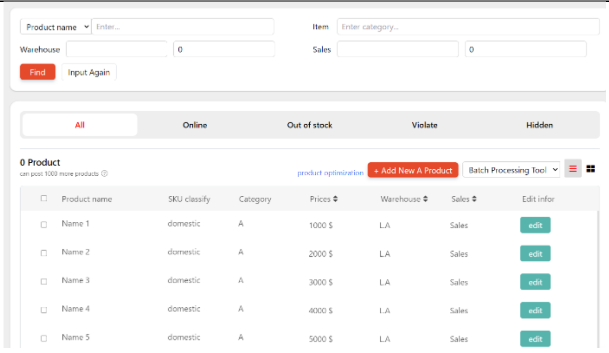
Code orders | Input code orders... | Search | Set Again

0 Orders | Sorted By | Mass Delivery

Product	Total Orders	Status	Countdown	Delivery	Operation

Order not found

Figure 5.1.2b All orders page



Product name: Enter... | Item: Enter category...
Warehouse: 0 | Sales: 0
Find | Input Again

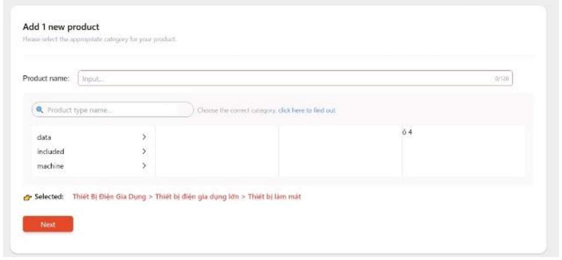
All | Online | Out of stock | Violate | Hidden

0 Product
can post 1000 more products

product optimization | + Add New A Product | Batch Processing Tool

Product name	SKU classify	Category	Prices	Warehouse	Sales	Edit infor
Name 1	domestic	A	1000 $	L.A	Sales	edit
Name 2	domestic	A	2000 $	L.A	Sales	edit
Name 3	domestic	A	3000 $	L.A	Sales	edit
Name 4	domestic	A	4000 $	L.A	Sales	edit
Name 5	domestic	A	5000 $	L.A	Sales	edit

Figure 5.1.2c All products page



Add 1 new product
Please select the appropriate category for your product.

Product name: Input... 0/120

Product type name... Choose the correct category, click here to find out

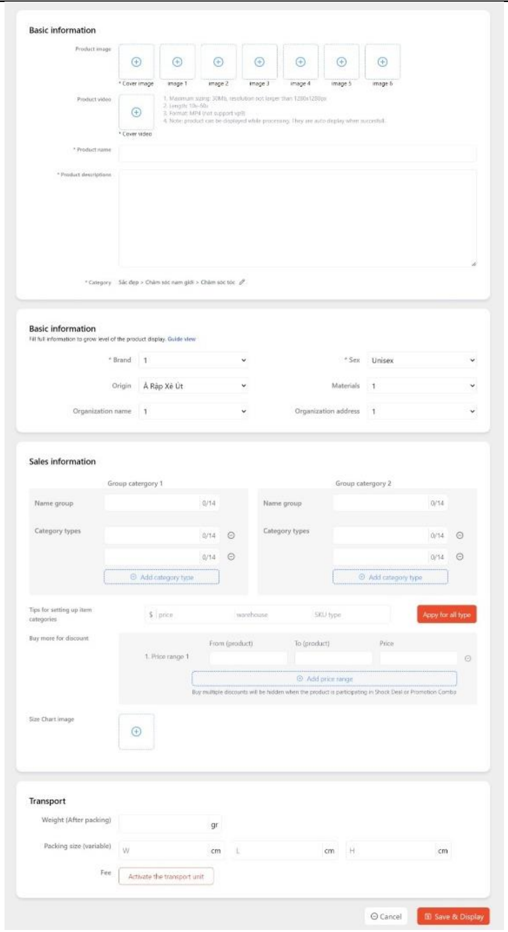
- data >
- included >
- machine >

ô 4

Selected: Thiết Bị Điện Gia Dụng > Thiết bị điện gia dụng lớn > Thiết bị làm mát

Next

Figure 5.1.2d Create category page



Basic information

Product image: * Cover image, image 1, image 2, image 3, image 4, image 5, image 6

Product video: * Cover video
1. Maximum sizing: 30Mb, resolution not larger than 1280x1280px
2. Length: 10s-60s
3. Format: MP4 (not support vp9)
4. Note: product can be displayed while processing. They are auto display when successful.

* Product name
* Product descriptions
* Category: Sắc đẹp > Chăm sóc nam giới > Chăm sóc tóc

Basic information
Fill full information to grow level of the product display. Guide view

* Brand	1	* Sex	Unisex
Origin	Ả Rập Xê Út	Materials	1
Organization name	1	Organization address	1

Sales information

Group catergory 1: Name group 0/14; Category types 0/14, 0/14; Add category type
Group catergory 2: Name group 0/14; Category types 0/14, 0/14; Add category type

Tips for setting up item categories: $ price | warehouse | SKU type | Appy for all type

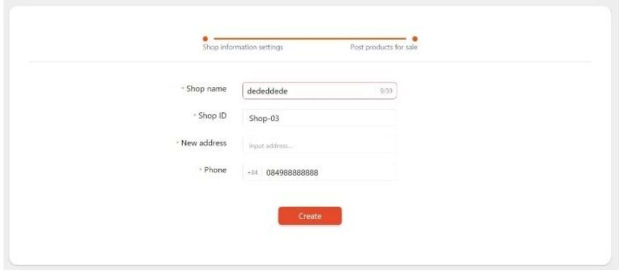
Buy more for discount: From (product) | To (product) | Price
1. Price range 1
Add price range
Buy multiple discounts will be hidden when the product is participating in Shock Deal or Promotion Combo

Size Chart image

Transport
Weight (After packing): gr
Packing size (variable): W cm | L cm | H cm
Fee: Activate the transport unit

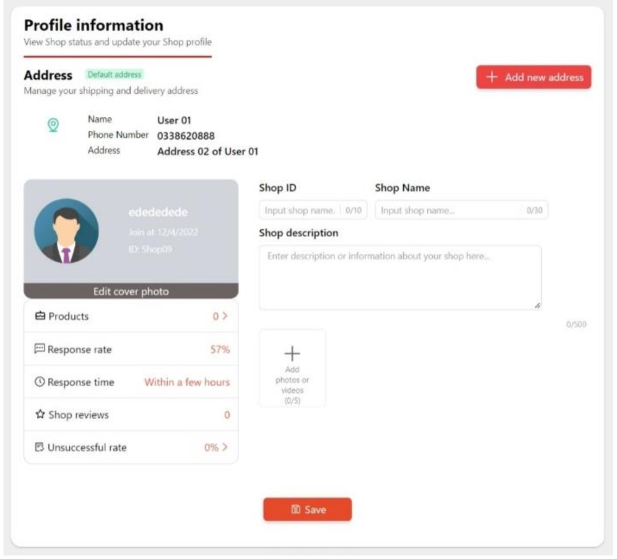
Cancel | Save & Display

Figure 5.1.2e Add products by category page


Shop information settings — Post products for sale

- * Shop name: dededdede 9/30
- * Shop ID: Shop-03
- * New address: Input address...
- * Phone: +84 084988888888

Figure 5.1.2f Create new shop page


Profile information

View Shop status and update your Shop profile

Address Default address

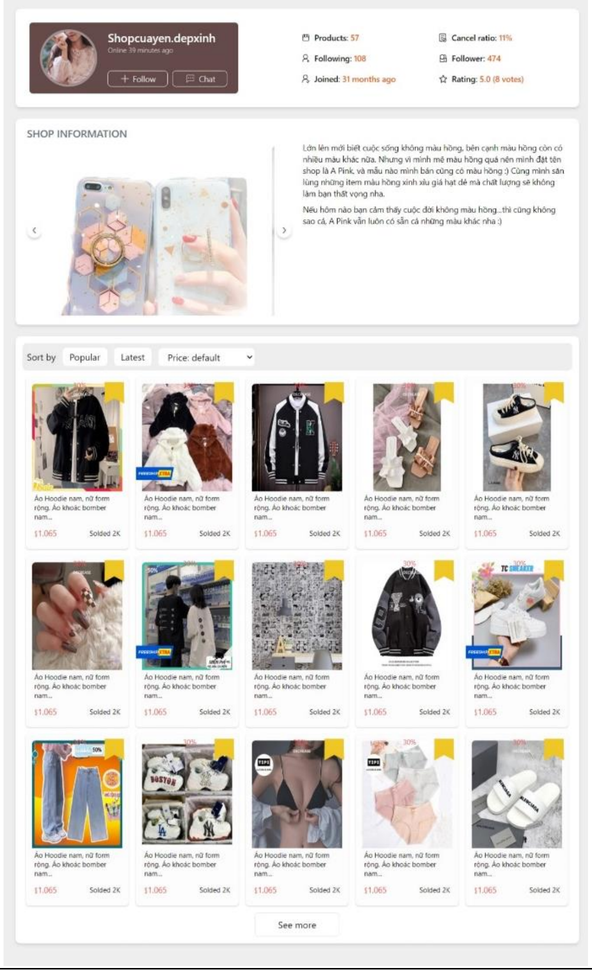
Manage your shipping and delivery address

Name	User 01
Phone Number	0338620888
Address	Address 02 of User 01

edededede
Join at 12/4/2022
ID: Shop09

Products	0
Response rate	57%
Response time	Within a few hours
Shop reviews	0
Unsuccessful rate	0%

Shop ID: Input shop name. 0/10
Shop Name: Input shop name... 0/30
Shop description: Enter description or information about your shop here... 0/500


Shopcuayen.depxinh
Online 39 minutes ago

Products: 57	Cancel ratio: 11%
Following: 108	Follower: 474
Joined: 31 months ago	Rating: 5.0 (8 votes)

SHOP INFORMATION

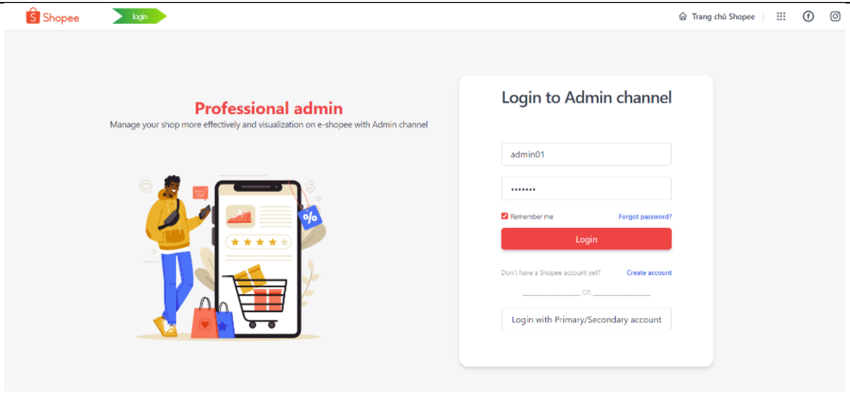
Lớn lên mới biết cuộc sống không màu hồng, bên cạnh màu hồng còn có nhiều màu khác nữa. Nhưng vì mình mê màu hồng quá nên mình đặt tên shop là A Pink, và mẫu nào mình bán cũng có màu hồng :) Cùng mình săn lùng những item màu hồng xinh xỉu giá hạt dẻ mà chất lượng sẽ không làm bạn thất vọng nha.

Nếu hôm nào bạn cảm thấy cuộc đời không màu hồng...thì cũng không sao cả, A Pink vẫn luôn có sẵn cả những màu khác nha :)

Sort by: Popular | Latest | Price: default

Áo Hoodie nam, nữ form rộng. Áo khoác bomber nam... $1.065 Solded 2K
Áo Hoodie nam, nữ form rộng. Áo khoác bomber nam... $1.065 Solded 2K
Áo Hoodie nam, nữ form rộng. Áo khoác bomber nam... $1.065 Solded 2K
Áo Hoodie nam, nữ form rộng. Áo khoác bomber nam... $1.065 Solded 2K
Áo Hoodie nam, nữ form rộng. Áo khoác bomber nam... $1.065 Solded 2K
Áo Hoodie nam, nữ form rộng. Áo khoác bomber nam... $1.065 Solded 2K
Áo Hoodie nam, nữ form rộng. Áo khoác bomber nam... $1.065 Solded 2K
Áo Hoodie nam, nữ form rộng. Áo khoác bomber nam... $1.065 Solded 2K
Áo Hoodie nam, nữ form rộng. Áo khoác bomber nam... $1.065 Solded 2K
Áo Hoodie nam, nữ form rộng. Áo khoác bomber nam... $1.065 Solded 2K
Áo Hoodie nam, nữ form rộng. Áo khoác bomber nam... $1.065 Solded 2K
Áo Hoodie nam, nữ form rộng. Áo khoác bomber nam... $1.065 Solded 2K
Áo Hoodie nam, nữ form rộng. Áo khoác bomber nam... $1.065 Solded 2K
Áo Hoodie nam, nữ form rộng. Áo khoác bomber nam... $1.065 Solded 2K
Áo Hoodie nam, nữ form rộng. Áo khoác bomber nam... $1.065 Solded 2K

III. Admin interface


Professional admin

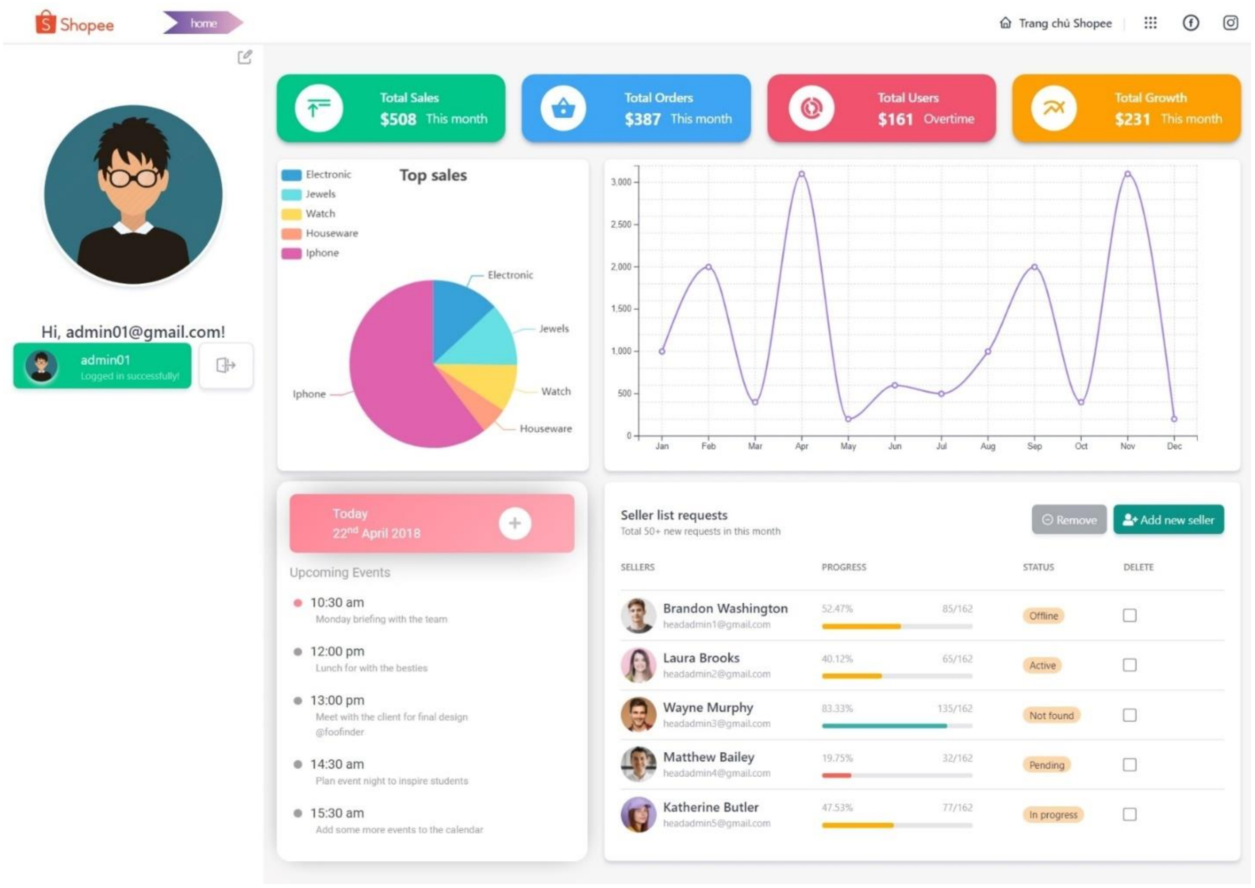
Manage your shop more effectively and visualization on e-shopee with Admin channel

Login to Admin channel

- admin01
- •••••••
- [✓] Remember me — Forgot password?

Don't have a Shopee account yet? Create account

Figure 5.1.3a Admin login page



Upcoming Events

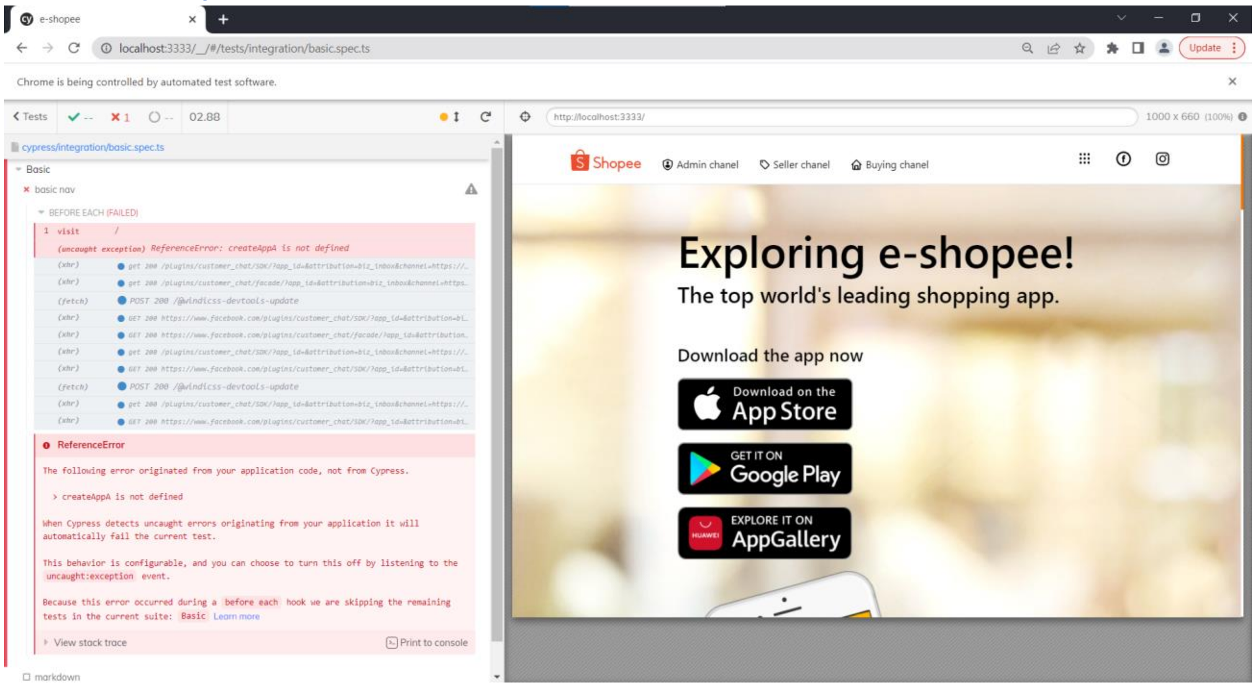
- 10:30 am
  Monday briefing with the team
- 12:00 pm
  Lunch for with the besties
- 13:00 pm
  Meet with the client for final design @foofinder
- 14:30 am
  Plan event night to inspire students
- 15:30 am
  Add some more events to the calendar

Seller list requests
Total 50+ new requests in this month

SELLERS	PROGRESS		STATUS	DELETE
Brandon Washington headadmin1@gmail.com	52.47%	85/162	Offline	☐
Laura Brooks headadmin2@gmail.com	40.12%	65/162	Active	☐
Wayne Murphy headadmin3@gmail.com	83.33%	135/162	Not found	☐
Matthew Bailey headadmin4@gmail.com	19.75%	32/162	Pending	☐
Katherine Butler headadmin5@gmail.com	47.53%	77/162	In progress	☐

Figure 5.1.3b Admin dashboard page

5.2 Analysis and unit test



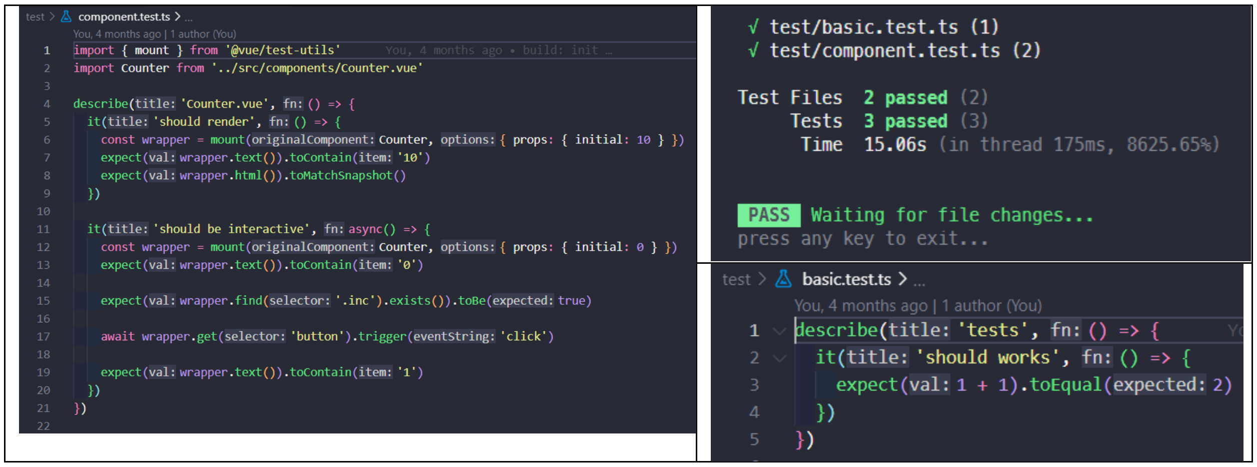
localhost:3333/__/#/tests/integration/basic.spec.ts

Chrome is being controlled by automated test software.

http://localhost:3333/

```
cypress/integration/basic.spec.ts
Basic
  × basic nav
    BEFORE EACH (FAILED)
    1 visit       /
      (uncaught exception) ReferenceError: createAppA is not defined
      (xhr)     get 200 /plugins/customer_chat/SDK/?app_id=&attribution=biz_inbox&channel=https://…
      (xhr)     get 200 /plugins/customer_chat/facade/?app_id=&attribution=biz_inbox&channel=https…
      (fetch)   POST 200 /@windicss-devtools-update
      (xhr)     GET 200 https://www.facebook.com/plugins/customer_chat/SDK/?app_id=&attribution=bi…
      (xhr)     GET 200 https://www.facebook.com/plugins/customer_chat/facade/?app_id=&attribution…
      (xhr)     get 200 /plugins/customer_chat/SDK/?app_id=&attribution=biz_inbox&channel=https://…
      (xhr)     GET 200 https://www.facebook.com/plugins/customer_chat/SDK/?app_id=&attribution=bi…
      (fetch)   POST 200 /@windicss-devtools-update
      (xhr)     get 200 /plugins/customer_chat/SDK/?app_id=&attribution=biz_inbox&channel=https://…
      (xhr)     GET 200 https://www.facebook.com/plugins/customer_chat/SDK/?app_id=&attribution=bi…

ReferenceError

The following error originated from your application code, not from Cypress.

  > createAppA is not defined

When Cypress detects uncaught errors originating from your application it will
automatically fail the current test.

This behavior is configurable, and you can choose to turn this off by listening to the
uncaught:exception event.

Because this error occurred during a before each hook we are skipping the remaining
tests in the current suite: Basic Learn more

View stack trace                                              Print to console
```

Figure 5.2a Testing router with cypress



```
test > component.test.ts > ...
You, 4 months ago | 1 author (You)
import { mount } from '@vue/test-utils'        You, 4 months ago • build: init ...
import Counter from '../src/components/Counter.vue'

describe(title: 'Counter.vue', fn: () => {
  it(title: 'should render', fn: () => {
    const wrapper = mount(originalComponent: Counter, options: { props: { initial: 10 } })
    expect(val: wrapper.text()).toContain(item: '10')
    expect(val: wrapper.html()).toMatchSnapshot()
  })

  it(title: 'should be interactive', fn: async() => {
    const wrapper = mount(originalComponent: Counter, options: { props: { initial: 0 } })
    expect(val: wrapper.text()).toContain(item: '0')

    expect(val: wrapper.find(selector: '.inc').exists()).toBe(expected: true)

    await wrapper.get(selector: 'button').trigger(eventString: 'click')

    expect(val: wrapper.text()).toContain(item: '1')
  })
})
```

```
 √ test/basic.test.ts (1)
 √ test/component.test.ts (2)

 Test Files  2 passed (2)
      Tests  3 passed (3)
       Time  15.06s (in thread 175ms, 8625.65%)


 PASS  Waiting for file changes...
 press any key to exit...
```

```
test > basic.test.ts > ...
You, 4 months ago | 1 author (You)
describe(title: 'tests', fn: () => {
  it(title: 'should works', fn: () => {
    expect(val: 1 + 1).toEqual(expected: 2)
  })
})
```

Figure 5.2b Testing component unit render with vitest

5.3 Test for use cases

- Before entering a certain url, if you are not logged in, the url will be redirected to the login page.
- <u>In buyer session:</u>
  - If registered with the same name, the server reports an error that the user already exists.
  - Date of birth must be in DD/MM/YYYY format.
  - The registered phone number in the account must be 11 digits.
  - credit card number must be 16 digits, cvv number must be 3 digit, post ZIP code must be 5 digits.
  - When registering and updating an account, the fields cannot be left blank.
- <u>In seller session:</u>
  - When registering to add a product, if the address already exists in the account profile, there is no need to fill it in anymore.
- <u>In admin session:</u>
  - Before login to admin must logout user account.

VI. CONCLUSIONS

- Through this project, we can learn about create SPA (Single Page App) as dynamic website to communicate with server through API connections.
- Authenticate with token access.
- Learning about MVC model and CRUD actions in API.
- Practicing and enhancing about SQL actions and database knowledge.
- Understanding about principle of database working.
- Improving coding skills and UI/UX.
- Initial forming about microservices project by components system.