

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH  
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN



**BÁO CÁO ĐỒ ÁN MÔN HỌC**

*Công Nghệ Web Và Ứng Dụng - SE347.K11.PMCL*

Giảng viên: **Trần Anh Dũng**

**ĐỀ TÀI: WEBSITE BÁN HOA QUA MẠNG**

<b><u>NHÓM 01:</u></b>	<b>La Văn Tiến</b>	<b>– 15520878</b>
	<b>Nguyễn Tuấn Phương Nam</b>	<b>– 15520519</b>
	<b>Lê Viết Huỳnh</b>	<b>– 15520327</b>

## MỤC LỤC

<b>I.</b>	<b>Giới thiệu tổng quan</b>	<b>3</b>
<b>II.</b>	<b>Khảo sát yêu cầu, phân tích Use Case và phân chia công việc</b>	<b>3</b>
	1. Khảo sát yêu cầu và phân tích Use Case	3
	2. Phân chia công việc	4
<b>III.</b>	<b>Phân tích, thiết kế và các công nghệ sử dụng</b>	<b>5</b>
	1. Thiết kế tổng quan dự án	5
	2. Thiết kế cơ sở dữ liệu	5
	3. Thiết kế back-end	6
	4. Thiết kế front-end	6
	5. Các công nghệ sử dụng	7
<b>IV.</b>	<b>Hiện thực phần mềm</b>	<b>7</b>
	1. Phần cơ sở dữ liệu	7
	2. Phần back-end	9
	3. Phần front-end	14
<b>V.</b>	<b>Hướng dẫn cài đặt</b>	<b>24</b>
<b>VI.</b>	<b>Hướng dẫn sử dụng và trưng bày sản phẩm bằng hình ảnh</b>	<b>25</b>
	1. Đối với khách hàng	25
	2. Đối với người quản lý	29
<b>VII.</b>	<b>Tổng kết đánh giá và hướng phát triển</b>	<b>33</b>
	1. Tổng kết và đánh giá:	33
	2. Hướng phát triển	34
<b>VIII.</b>	<b>Tài liệu tham khảo</b>	<b>34</b>

### **I. Giới thiệu tổng quan**

- Dự án cuối kỳ của nhóm em là một website bán hoa qua mạng theo phong cách tối giản và hiện đại, sử dụng gam màu đen trắng xanh tạo điểm nhấn, ngoài ra còn bởi cách thiết kế và logic “càng đơn giản càng tốt” - tạo cho khách hàng sự dễ chịu và tiết kiệm thời gian của cả người quản lý shop cũng như khách hàng.
- Cung cấp cho khách hàng cũng như người quản lý shop những công cụ hiệu quả và ít rườm rà nhất có thể để dễ dàng hoàn thành công việc của mình, từ việc chọn hàng, đặt hàng cho đến việc tạo tài khoản hoặc lấy lại mật khẩu.
- Nhằm vận dụng những kiến thức học được trên lớp và trao đổi qua internet để thử thách bản thân, tạo ra sản phẩm có sự đột phá và không theo khuôn mẫu cũ, tuy nhiên vẫn đảm bảo được các tính năng cần thiết của một trang web bán hàng cơ bản.

## **II. Khảo sát yêu cầu, phân tích Use Case và phân chia công việc**

### **1. Khảo sát yêu cầu và phân tích Use Case:**

Phần mềm Website bán hoa qua mạng có những chức năng như sau:

- Phần dành cho khách hàng truy cập:
  - Trang mua hàng và giỏ hàng.
  - Trang thông tin người dùng và liệt kê hóa đơn đã đặt.
  - Tạo tài khoản, đăng nhập, quên mật khẩu.
  - Trang thông tin liên lạc với cửa hàng
- Phần dành cho người quản lý truy cập:
  - Trang quản lý sản phẩm.
  - Trang tổng kết hóa đơn đã đặt.
- Các tính năng tiện lợi:
  - Hỗ trợ hoàn toàn đa ngôn ngữ: tiếng Việt, tiếng Anh. Chuyển đổi dễ dàng giữa 2 ngôn ngữ một cách tức thời.
  - Thanh điều hướng thiết kế phong cách giản lược, có giới hạn khả năng truy cập tùy loại tài khoản đăng nhập.

- Trang bán hàng giao diện tối giản, có chức năng tìm sản phẩm và sắp xếp sản phẩm theo giá, mỗi một ô hàng thể hiện đầy đủ thông tin của món hàng và đầy đủ chức năng đặt hàng mà không cần phải làm một trang thông tin chi tiết riêng biệt, tiết kiệm thời gian cho khách hàng. Khi bấm vào món hàng nào sẽ hiện hình ảnh chi tiết và thông tin chi tiết của món hàng đó.
- Giỏ hàng hỗ trợ thêm số lượng hoặc xóa nhiều sản phẩm đã chọn trực tiếp.
- Hỗ trợ khách viếng thăm có thể đặt hàng không cần tạo tài khoản.
- Hỗ trợ tính năng One-click Order, bấm 1 cái là đặt hàng luôn như Now.
- Sau khi tạo tài khoản hoặc đổi mật khẩu xong sẽ tự động lập tức đăng nhập lại cho khách hàng.
- Trang quản lý sản phẩm cho phép import/export excel, chức năng thêm, sửa và xóa một hoặc nhiều sản phẩm một lúc, đầy đủ bộ lọc, tìm kiếm và sắp xếp.
- Có spinner khi đợi load dữ liệu hoặc chuyển trang.

## **2. Phân chia công việc:**

- Phân tích thiết kế, quản lý dự án và khởi tạo code nền: La Văn Tiến
- Thiết kế và xây dựng database, tạo files dịch thuật: Lê Viết Huỳnh
- Thiết kế và xây dựng back-end: Nguyễn Tuấn Phương Nam
- Thiết kế và xây dựng front-end: La Văn Tiến
- Kế hoạch làm việc:
  - Các thành viên nhóm sẽ dùng chung 2 tài khoản github để commit code, sẽ có 2 repositories, 1 cái chính 1 cái để test tính năng mới và dự phòng.
  - Các thành viên sẽ họp với nhau mỗi 2 tuần một lần để khảo sát, đồng nhất tiến độ và định hướng đi tiếp theo.

## **III. Phân tích, thiết kế và các công nghệ sử dụng**

## 1. Thiết kế tổng quan dự án:

- Tất cả code sẽ nằm trong 1 project duy nhất, khi package lại sẽ ra một file jar duy nhất giúp việc dễ dàng deploy lên một server/cloud.
- Các bảng trong cơ sở dữ liệu sẽ hoàn toàn không có khóa ngoại tham chiếu đến bảng khác, việc tham chiếu hay xử lý sẽ phụ thuộc hoàn toàn vào phía code. Điều này sẽ giúp đơn giản hóa quá trình phát triển phần mềm, dễ bảo trì nâng cấp và sửa lỗi. Theo xu hướng của các công ty làm web lớn hiện tại như PALTech.
- Giảm tối thiểu số bảng cần sử dụng.
- Endpoints thiết kế dựa theo REST API nhưng không cần phải tuân thủ chặt chẽ, mỗi một bảng tương ứng với 1 endpoint bên phía back-end. Sau đó thêm các requests phù hợp vào thêm. Back-end cần bố cục các API theo bảng. VD: bảng Product thì tương ứng phía back-end có 3 files: Product entity, Product repository và Product controller để cùng một folder product. Các DTOs nếu cần thiết sẽ để cùng trong controller tương ứng không tạo DTO thừa thãi. Không phân ra một folder toàn models riêng, một folder toàn controllers riêng như kiểu cũ. Việc này là để modul hóa tối đa code base, tạo điều kiện cho việc dễ dàng bảo trì và nâng cấp.
- Hệ thống sẽ không sử dụng cookies hay sessions kiểu truyền thống mà sẽ sử dụng một mô hình thiết kế hiện đại khá phổ biến hiện nay là Stateless Authentication, chứng thực và theo dõi users cũng như cấp phép API thông qua tokens.
- Front-end các trang cho người dùng yêu cầu phải hỗ trợ tốt responsive.

## 2. Thiết kế cơ sở dữ liệu:

**Product** (id, name, description, imgUrl, price, quantity, typeName, categoryName).

**Category** (id, name).

**Type** (id, name, categoryName).

**User** (id, name, password, email, phone, address, district, city, answer, type, enable).

*// type có thể là “ADMIN”, “USER” hoặc “GUESS”.*

*// enable là kiểu boolean: tài khoản đang hoạt động hoặc bị khóa*

**Bill** (id, placementDate, productId, productQuantity, price, settlementDate, status, userId, phone, detailAddress).

*// Nếu người đặt là khách không có tài khoản thì userId = 0.*

*// status có thể là “SUCCESS”, “FAILED” hoặc “PENDING”.*

### 3. Thiết kế back-end:

Sẽ có 5 API tương ứng với 5 bảng trong cơ sở dữ liệu. Mỗi API sẽ có:

- Class entity: Spring Entity tự động map entity với bảng.
- Interface repository: Spring Data JPA cung cấp sẵn các thao tác CRUD cơ bản. Nếu có phát sinh nhu cầu query đặc biệt thì chỉ cần khai báo prototype ở đây là JPA sẽ tự hiểu. JPA query dựa trên tên method. VD: public List<User> findAllByEmailAndCity(...);
- File controller: toàn bộ endpoints cơ bản theo chuẩn REST API, các endpoints đặc biệt khác và logic của API.

Ngoài 5 APIs còn có 2 class services là:

- MailService: cài đặt việc gửi mail.
- UserService: theo dõi việc đăng nhập của các users vào hệ thống.

### 4. Thiết kế front-end:

Sẽ có 5 pages:

- Cửa hàng (shop - hỗ trợ responsive - mọi users ngoại trừ “ADMIN”): trưng bày hàng và cho khả năng chọn vào giỏ hàng.
- Thông tin (info - hỗ trợ responsive - chỉ “USER” mới được truy cập): hiển thị thông tin chi tiết của khách hàng cũng như các hóa đơn đã đặt cũng như tổng tiền khách hàng đã tiêu tại cửa hàng.

- Liên hệ (contact - hỗ trợ responsive - mọi users): Thông tin liên lạc với cửa hàng, ở đây chỉ để tên các thành viên nhóm một cách tượng trưng.
- Kho hàng (admin or warehouse - chỉ “ADMIN” mới được truy cập): cho phép toàn bộ thao tác quản lý và kiểm soát đối với các sản phẩm.
- Tổng kết (summary - chỉ “ADMIN” mới được truy cập): thể hiện đầy đủ các hóa đơn đã đặt với cửa hàng cộng với tổng doanh số.

Các tính năng giỏ hàng, đăng nhập, đăng ký, đăng xuất, đổi mật khẩu đều nằm gọn trên thanh điều hướng cho phép việc dễ dàng và tiện lợi thao tác. Cộng với tông màu trắng đen, nền sáng chủ đạo, hướng đến một phong cách tối giản hiện đại.

## **5. Các công nghệ sử dụng:**

- Database: MySQL 8
- Back-end: Java 11 Spring Boot và Spring Data JPA
- Front-end: Angular TypeScript và Bootstrap
- Build/Package/Deploy: Maven

## **IV. Hiện thực phần mềm:**

### **1. Phần cơ sở dữ liệu:**

- Bảng sẽ tự động được tạo nếu chưa tồn tại khi chạy phần back-end Spring Boot lên.

- Tất cả các Category và Type cũng như các tài khoản admin sẽ được chuẩn bị trước trong file run.sql.

```

38 VALUES (15, 'IRISES', 'FLOWERS');
39 insert into type (id, name, category_name)
40 values (16, 'LISTIANTHUS', 'FLOWERS');
41 insert into type (id, name, category_name)
42 values (17, 'PEONIES', 'FLOWERS');
43 insert into type (id, name, category_name)
44 values (18, 'STOCK', 'FLOWERS');
45 insert into type (id, name, category_name)
46 values (19, 'SNAPDRAGONS', 'FLOWERS');
47 insert into type (id, name, category_name)
48 values (20, 'TULIPS', 'FLOWERS');
49 insert into type (id, name, category_name)
50 values (21, 'TROPICAL FLOWERS', 'FLOWERS');
51 insert into type (id, name, category_name)
52 values (22, 'MIXED BOUQUETS', 'FLOWERS');
53
54 insert into type (id, name, category_name)
55 values (23, 'BLOOMING PLANTS', 'PLANTS');
56 insert into type (id, name, category_name)
57 values (24, 'GREEN PLANTS', 'PLANTS');
58 insert into type (id, name, category_name)
59 values (25, 'ORCHIDS & TROPICALS', 'PLANTS');
60
61 insert into user values (1, 'Đường Đời', 'Hô Chí Minh', 'Bình Thạnh', 'flowershop.noreply@gmail.com', true, 'lavantien',
62 'MTIzNHf3ZXI=', '0960960096', 'ADMIN');
63 insert into user values (2, 'Đường Đời', 'Hô Chí Minh', 'Bình Thạnh', 'flowershop1.noreply@gmail.com', true, 'nguyentuanphuongnam',
64 'MTIzNHf3ZXI=', '0960960096', 'ADMIN');
65 insert into user values (3, 'Đường Đời', 'Hô Chí Minh', 'Bình Thạnh', 'flowershop2.noreply@gmail.com', true, 'leviethuynh',
66 'MTIzNHf3ZXI=', '0960960096', 'ADMIN');

```

- Các Product mẫu sẽ được chuẩn bị bằng file product.json, chạy Postman gọi POST API /product để thêm dữ liệu vào database.



The image shows a development environment with two main components:

### IDE (Left Panel)

- Project Structure:** A file explorer on the left shows a project named 'flowershop' with subdirectories like 'db', 'frontend', 'project-pictures', 'release', 'src', and 'target'.
- product.json:** A JSON file is open in the editor, containing two product entries:
 

```

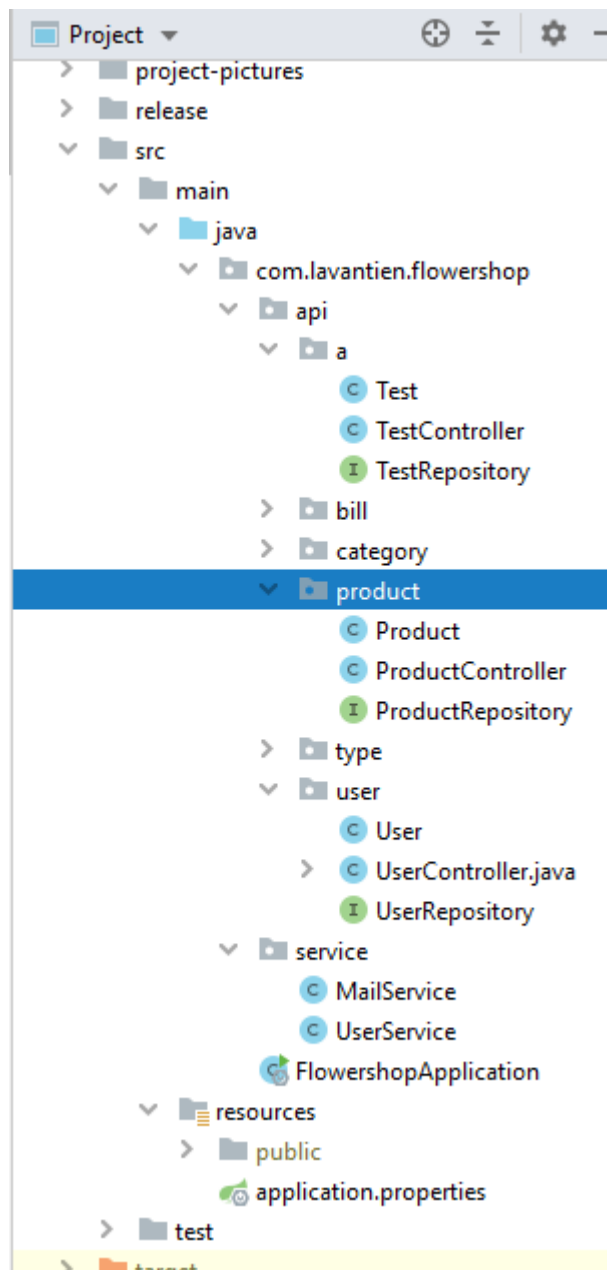
      {
        "id": 1,
        "categoryName": "FLOWERS",
        "description": "With its feminine feel and European charm, this alluring glass cube is the perfect Mother's Day surprise! It's carefully hand-decorated and filled with a fresh bouquet of daisies and roses.",
        "imgUrl": "aHR0cHM6Ly9pbWcudGVsZWZsb3JhLmNvbS9pbWFnZXNvb18wL2xZfZmxvd2VyczpUMThNNDAwQyxwZ182L3dFODAwLGHfMTAwMCxjc19ub19jbXlrLGNfcGFkl2ZfanBnLHFfYXV0bzplY28sZV9zaGFycGVuOjIwMC9mbG93ZXJzL1QxOUUzMDBDL1RlbGVmbG9yYSdzV2lsZGZsb3d1ckluRm91cXV1dFBN",
        "name": "Wildflower In Flight Bouquet",
        "price": 59.99,
        "typeName": "ROSES",
        "quantity": 10
      },
      {
        "id": 2,
        "categoryName": "FLOWERS",
        "description": "The shimmering, iridescent aqua blue glass of this stunning cylinder vase is the perfect partner for a fresh, fabulous bouquet of lavender roses, alstroemeria and tulips!",
        "imgUrl": "aHR0cHM6Ly9pbWcudGVsZWZsb3JhLmNvbS9pbWFnZXNvb18wL2xZfZmxvd2VyczpUMThNNDAwQyxwZ182L3dFODAwLGHfMTAwMCxjc19ub19jbXlrLGNfcGFkl2ZfanBnLHFfYXV0bzplY28sZV9zaGFycGVuOjIwMC9mbG93ZXJzL1QxOUUzMDBDL1RlbGVmbG9yYSdzV2lsZGZsb3d1ckluRm91cXV1dFBN",
        "name": "Art Glass Garden Bouquet",
        "price": 69.99,
        "typeName": "ROSES",
        "quantity": 10
      }
      
```

### REST Client (Right Panel)

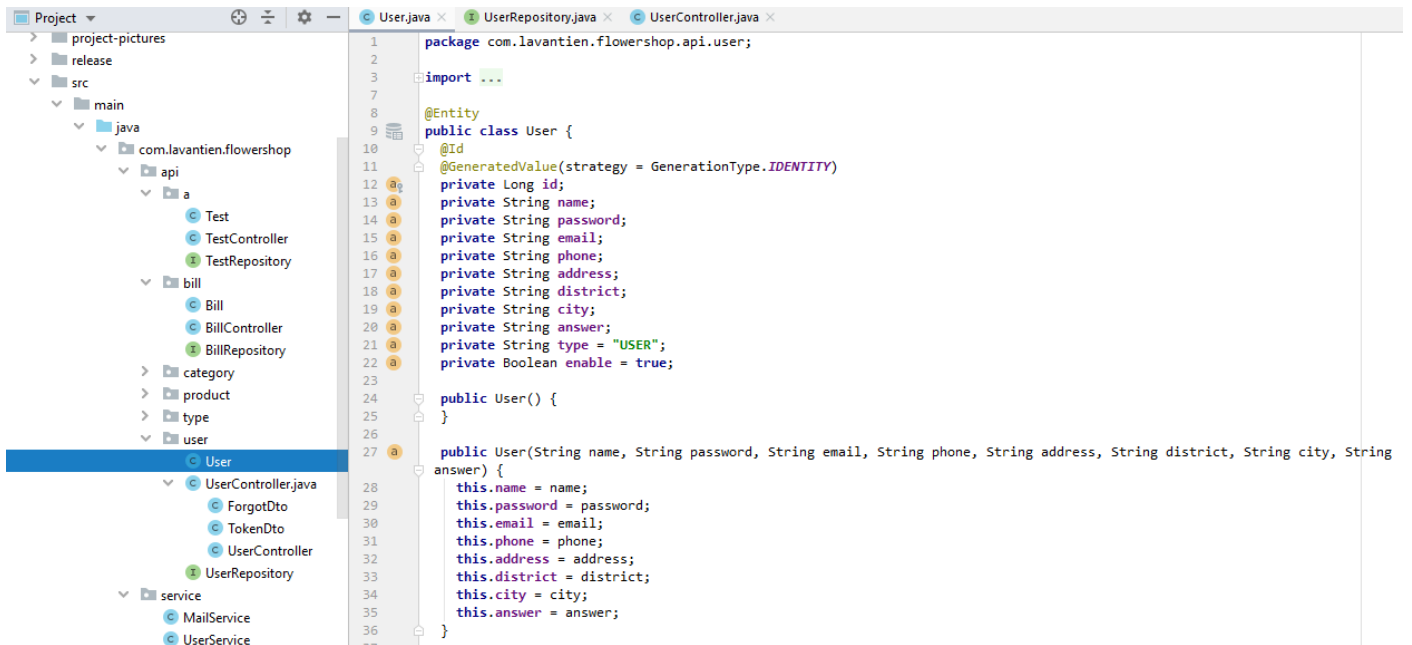
- Method:** POST
- URL:** http://localhost:8080/api/product
- Body:** The body tab is selected, showing the same JSON data as in the IDE.

## 2. Phần back-end:

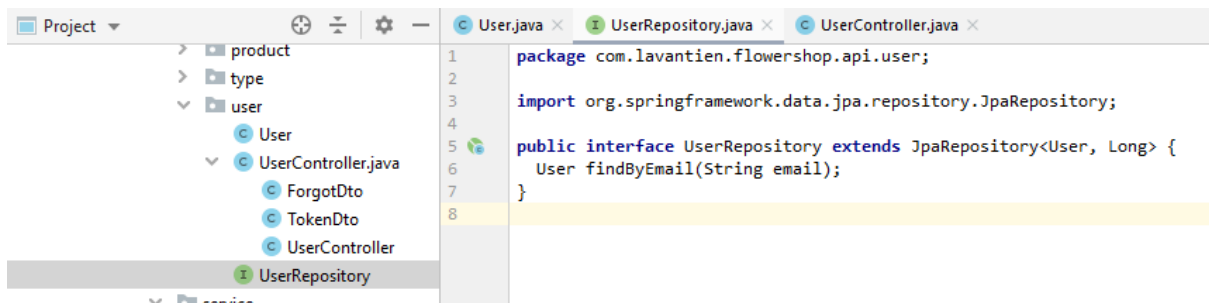
- Cấu trúc tổng thể:



- Ví dụ một API:
  - User entity:



➤ User repository:



➤ User controller:

```

1 package com.lavantien.flowershop.api.user;
2
3 import ...
4
12
13 @RestController
14 @RequestMapping("/api/user")
15 public class UserController {
16     private UserRepository userRepository;
17     private MailService mailService;
18     private UserService userService;
19
20     @Autowired
21     public UserController(UserRepository userRepository, MailService mailService, UserService userService) {
22         this.userRepository = userRepository;
23         this.mailService = mailService;
24         this.userService = userService;
25     }
26
27     @GetMapping
28     public ResponseEntity<List<User>> getAll() { return ResponseEntity.ok(userRepository.findAll()); }
29
30     @PostMapping
31     public ResponseEntity<List<User>> createMany(@RequestBody List<User> users) {
32         return ResponseEntity.ok(userRepository.saveAll(users));
33     }
34
35     @DeleteMapping
36     public ResponseEntity deleteMany(@RequestBody(required = false) List<Long> ids) {
37         if (ids == null) {
38             userRepository.deleteAll();
39             return ResponseEntity.ok().build();
40         }
41         userRepository.deleteAll(userRepository.findAllById(ids));
42         return ResponseEntity.ok().build();
43     }
44
45     @GetMapping("/{id}")
46     public ResponseEntity<User> getById(@PathVariable Long id) {
47         Optional<User> user = userRepository.findById(id);
48         if (user.isEmpty()) {
49             return ResponseEntity.badRequest().build();
50         }
51         return ResponseEntity.ok(user.get());
52     }
53
54     @PostMapping("/create")
55     public ResponseEntity<User> create(@RequestBody User user) {
56         user.setPassword(Base64.getEncoder().encodeToString(user.getPassword().getBytes()));
57         return ResponseEntity.ok(userRepository.save(user));
58     }
59
60     @PutMapping("/{id}")
61     public ResponseEntity<User> update(@PathVariable Long id, @RequestBody User user) {
62         if (userRepository.findById(id).isEmpty()) {
63             return ResponseEntity.badRequest().build();
64         }
65         return ResponseEntity.ok(userRepository.save(user));
66     }
67
68     @DeleteMapping("/{id}")
69     public ResponseEntity delete(@PathVariable Long id) {
70         if (userRepository.findById(id).isEmpty()) {
71             return ResponseEntity.badRequest().build();
72         }
73         userRepository.deleteById(id);
74         return ResponseEntity.ok().build();
75     }
76
77     @PostMapping(value = "/login", consumes = "text/plain")
78     public ResponseEntity<TokenDto> doLogin(@RequestBody String info) {
79         String decodedInfo = new String(Base64.getDecoder().decode(info));
80     }
81

```

```

82         int index = decodedInfo.indexOf("j0z");
83         String email = decodedInfo.substring(0, index);
84         String password = decodedInfo.substring(index + 3);
85         User foundUser = userRepository.findByEmail(email);
86         String foundEncodedPassword = foundUser != null ? userRepository.findByEmail(email).getPassword() : "ajB6";
87         String foundPassword = new String(Base64.getDecoder().decode(foundEncodedPassword));
88         TokenDto tokenDto = new TokenDto(Base64.getEncoder().encodeToString("0+GUESS".getBytes()), phone: "0", detailAddress: "A, Bình
    Thanh, Hồ Chí Minh");
89         if (foundPassword.compareTo(password) == 0) {
90             if (userService.loggedInIds.isEmpty() || userService.loggedInIds.indexOf(foundUser.getId()) == -1) {
91                 userService.loggedInIds.add(foundUser.getId());
92             }
93             tokenDto.setToken(Base64.getEncoder().encodeToString((foundUser.getId() + "+" + foundUser.getType()).getBytes()));
94             tokenDto.setPhone(foundUser.getPhone());
95             tokenDto.setDetailAddress(foundUser.getAddress() + ", " + foundUser.getDistrict() + ", " + foundUser.getCity());
96         }
97         return ResponseEntity.ok(tokenDto);
98     }
99
100     @PostMapping("/logout")
101     public ResponseEntity<TokenDto> doLogout(@RequestBody TokenDto tokenDto) {
102         String decodedInfo = new String(Base64.getDecoder().decode(tokenDto.getToken()));
103         int index = decodedInfo.indexOf("+");
104         Long id = Long.parseLong(decodedInfo.substring(0, index));
105         // String type = decodedInfo.substring(index + 1);
106         if (!userService.loggedInIds.isEmpty()) {
107             userService.loggedInIds.remove(id);
108         }
109         return ResponseEntity.ok(new TokenDto(Base64.getEncoder().encodeToString("0+GUESS".getBytes()), phone: "0", detailAddress: "A,
    Bình Thanh, Hồ Chí MinhA, Bình Thanh, Hồ Chí Minh"));
110     }
111
112     @PostMapping("/resetPassword")
113     public ResponseEntity<TokenDto> doResetPassword(@RequestBody ForgotDto forgotDto) {
114         User foundUser = userRepository.findByEmail(forgotDto.getEmail());
115         if (foundUser == null || foundUser.getAnswer().compareTo(forgotDto.getAnswer()) != 0) {
116             return ResponseEntity.ok(new TokenDto(Base64.getEncoder().encodeToString("0+GUESS".getBytes()), phone: "0", detailAddress: "A,
    Bình Thanh, Hồ Chí Minh"));
117         }
118         foundUser.setPassword(forgotDto.getPassword());
119         userRepository.save(foundUser);
120         return ResponseEntity.ok(new TokenDto(Base64.getEncoder().encodeToString((foundUser.getId() + "+" + foundUser.getType())
    .getBytes()), foundUser.getPhone(), detailAddress: foundUser.getAddress() + ", " + foundUser.getDistrict() + ", " + foundUser
    .getCity());
121     }
122 }
123
124 class TokenDto {
125     private String token;
126     private String phone;
127     private String detailAddress;
128
129     public TokenDto() {
130     }
131
132     public TokenDto(String token, String phone, String detailAddress) {
133         this.token = token;
134         this.phone = phone;
135         this.detailAddress = detailAddress;
136     }
137
138     public String getToken() { return token; }
139
140     public void setToken(String token) { this.token = token; }
141
142     public String getPhone() { return phone; }
143
144     public void setPhone(String phone) { this.phone = phone; }
145
146     public String getDetailAddress() { return detailAddress; }
147
148     public void setDetailAddress(String detailAddress) { this.detailAddress = detailAddress; }
149
150
151
152
153
154
155
156
157
158

```

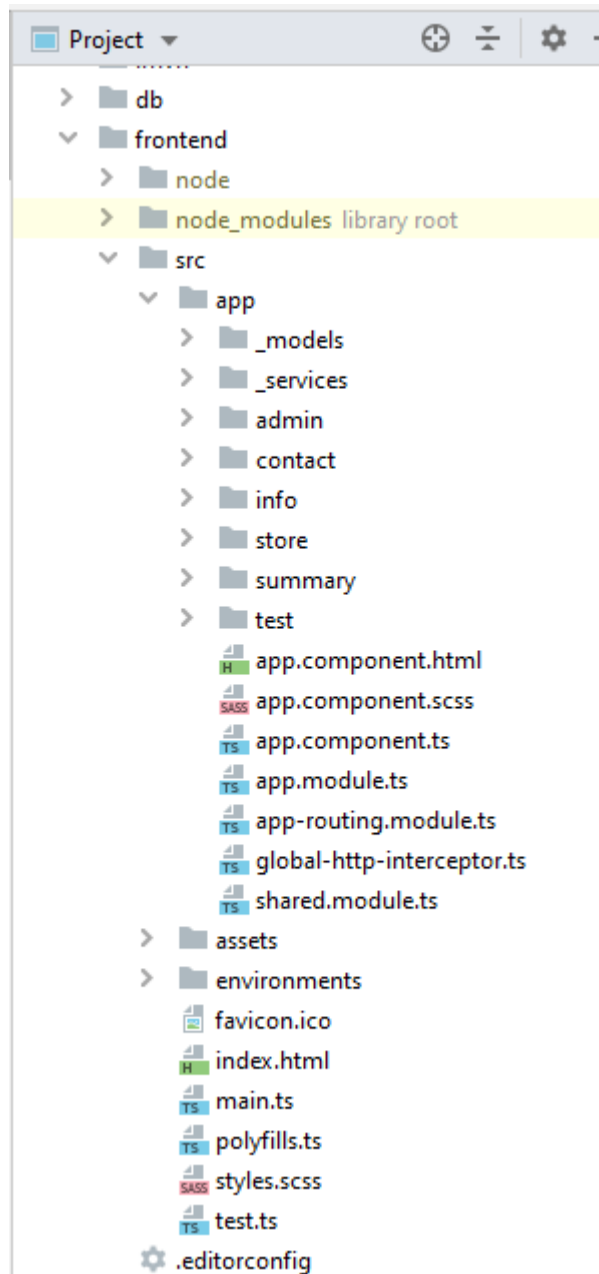
```

161 }
162
163 class ForgotDto {
164     private String email;
165     private String answer;
166     private String password;
167
168     public ForgotDto() {
169     }
170
171     public ForgotDto(String email, String answer, String password) {
172         this.email = email;
173         this.answer = answer;
174         this.password = password;
175     }
176
177     public String getEmail() { return email; }
178
179     public void setEmail(String email) { this.email = email; }
180
181     public String getAnswer() { return answer; }
182
183     public void setAnswer(String answer) { this.answer = answer; }
184
185     public String getPassword() { return password; }
186
187     public void setPassword(String password) { this.password = password; }
188 }
189
190
191
192
193
194
195
196
197
198
199
200
201

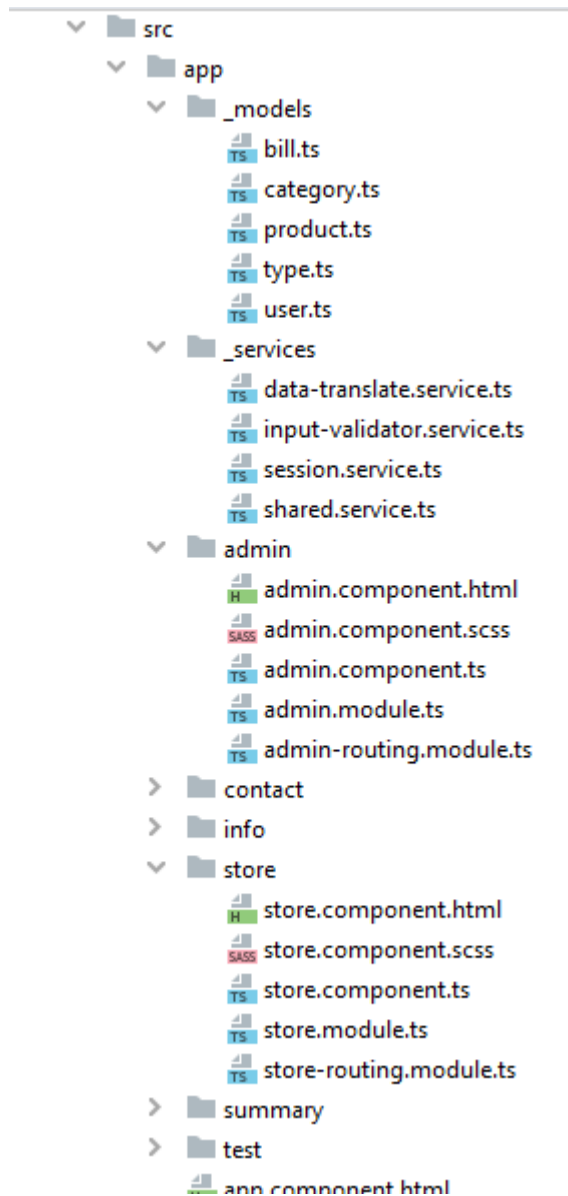
```

### 3. Phần front-end:

- Cấu trúc tổng thể:



- Cấu trúc chi tiết: mỗi component là một thư mục con. Gồm có 5 file chính là:
  - component.ts: Logic và render.
  - .HTML & .SCSS: HTML và CSS tương ứng của page.
  - .module.ts: import thư viện nếu cần.
  - ROUTING: khai báo routing của page tương ứng.



- Ví dụ file main (app.component.ts):

```
import ...

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.scss']
})
export class AppComponent implements OnInit, OnDestroy {
  isAdmin = false;
  isTest = false;
  isLoggedIn = false;
  faAngleDoubleUp = faAngleDoubleUp;
  faAngleDoubleDown = faAngleDoubleDown;
  faArrowLeft = faArrowLeft;
  faArrowRight = faArrowRight;
  faStore = faStore;
  faUser = faUser;
  faWarehouse = faWarehouse;
  faHandshake = faHandshake;
```



```

faCubes = faCubes;
faSearch = faSearch;
faSignInAlt = faSignInAlt;
faSignOutAlt = faSignOutAlt;
faChartLine = faChartLine;
faShoppingCart = faShoppingCart;
faMinus = faMinus;
faPlus = faPlus;
modalRef: BsModalRef;
modalRef2: BsModalRef;
loginForm = {
  email: '',
  password: ''
};
signUpForm = {
  name: '',
  email: '',
  reEmail: '',
  password: '',
  rePassword: '',
  answer: '',
  reAnswer: '',
  phone: '',
  address: '',
  district: 'Bình Thạnh',
  city: 'Hồ Chí Minh'
};
forgotPasswordForm = {
  email: '',
  answer: '',
  password: '',
  rePassword: ''
};
cartForm = {
  phone: '',
  address: '',
  district: 'Bình Thạnh',
  city: 'Hồ Chí Minh'
};
cities: City[] = [];
districts: District[] = [];
bgPrimary = '';
tcPrimary = '';
displayBg = 'LIGHT';
displayBgs = ['LIGHT', 'BLUE', 'GRAY', 'GREEN', 'RED', 'YELLOW',
'TEAL', 'BLACK', 'WHITE', 'TRANS'];
bgs = ['bg-light', 'bg-primary', 'bg-secondary', 'bg-success',
'bg-danger', 'bg-warning', 'bg-info', 'bg-dark', 'bg-white', 'bg-
transparent'];
tcs = ['text-dark', 'text-white', 'text-white', 'text-white',
'text-white', 'text-dark', 'text-white', 'text-white', 'text-
dark', 'text-dark'];
countOfIndividualProduct: number[] = [];
totalPriceOfIndividualProduct: number[] = [];
addedProducts: Product[] = [];
countAddedProduct = 0;
totalPriceOfAddedProduct = 0;
wrongLogin = false;
wrongCreate = false;

```

```

wrongForgot = false;
private subscriptions = new Subscription();
translate_CREATE_USER_SUCCESSFUL = '';
translate_RESET_PASSWORD_FAILED = '';
translate_RESET_PASSWORD_SUCCESSFUL = '';
translate_ORDER_SUCCESSFUL = '';

constructor(private http: HttpClient,
             private router: Router,
             private modalService: BsModalService,
             private inputValidator: InputValidatorService,
             private sharedService: SharedService,
             private sessionService: SessionService,
             public translate: TranslateService) {
    translate.addLangs(['en', 'vi']);
    translate.setDefaultLang('en');
    const browserLang = translate.getBrowserLang();
    translate.use(browserLang.match(/en|vi/) ? browserLang : 'en');

    this.subscriptions.add(this.sharedService.getGlobalBackgroundPrimary().subscribe(bg => {
        this.bgPrimary = bg[0];
        this.tcPrimary = bg[1];
    }));

    this.subscriptions.add(this.sessionService.getNewlyAddedProduct().subscribe(product => {
        if (!!product) {
            const prodIndex = this.addedProducts.findIndex(x => x.id === product.id);
            if (prodIndex === -1) {
                this.addedProducts.push(product);
                this.countOfIndividualProduct.push(1);
                this.totalPriceOfIndividualProduct.push(product.price);
                ++this.countAddedProduct;
            } else {
                ++this.countOfIndividualProduct[prodIndex];
                this.totalPriceOfIndividualProduct[prodIndex] += product.price;
            }
            this.totalPriceOfAddedProduct += product.price;
        } else {
            this.countAddedProduct = 0;
        }
    }));

    this.subscriptions.add(this.translate.stream('ALERT.CREATE_USER_SUCCESSFUL').subscribe(rs => {
        this.translate_CREATE_USER_SUCCESSFUL = rs;
    }));

    this.subscriptions.add(this.translate.stream('ALERT.RESET_PASSWORD_FAILED').subscribe(rs => {
        this.translate_RESET_PASSWORD_FAILED = rs;
    }));

    this.subscriptions.add(this.translate.stream('ALERT.RESET_PASSWORD_SUCCESSFUL').subscribe(rs => {
        this.translate_RESET_PASSWORD_SUCCESSFUL = rs;
    }));

```

```

    }));

    this.subscriptions.add(this.translate.stream('ALERT.ORDER_SUCCESSFUL').subscribe(rs => {
      this.translate_ORDER_SUCCESSFUL = rs;
    }));
    if (!LocalStorage.getItem('token')) {
      LocalStorage.setItem('token', btoa('0+GUESS'));
      LocalStorage.setItem('phone', '0');
      LocalStorage.setItem('detailAddress', 'A, Bình Thạnh, Hồ Chí Minh');
    }
  }

  ngOnInit() {
    this.getCities();
    this.getDistricts();
    this.isLoggedIn = LocalStorage.getItem('token') !== null &&
    atob(LocalStorage.getItem('token')) !== '0+GUESS';
    this.isAdmin = LocalStorage.getItem('token') !== null &&
    atob(LocalStorage.getItem('token')).substring(atob(LocalStorage.getItem('token')).indexOf('+') + 1) === 'ADMIN';
    if (this.isAdmin) {
      this.router.navigate(['/admin']);
    }
  }

  ngOnDestroy() {
    this.subscriptions.unsubscribe();
  }

  getCities() {
    this.http.get<City[]>('../assets/data/cities.json').subscribe(data => {
      if (!!data) {
        this.cities = data;
        this.signUpForm.city = this.cities[0].name;
      }
    }, error => {
      console.log(`Error: ${error}`);
      this.cities = [];
      this.signUpForm.city = '';
    }, () => {
    });
  }

  getDistricts() {
    this.http.get<District[]>('../assets/data/districts.json').subscribe(data => {...});
  }

  onChangeThemeColor() {...}

  openLoginModal(template: TemplateRef<any>) {
    this.modalRef = this.modalService.show(template);
  }

```

```

onLogin() {
  if (!this.inputValidator.isEmail(this.loginForm.email) ||
!this.inputValidator.isPassword(this.loginForm.password)) {
    this.wrongLogin = true;
    this.onRefreshLoginForm();
    return;
  }
  this.wrongLogin = false;
  this.http.post<TokenDto>('/api/user/login',
btoa(this.loginForm.email + 'j0z' + this.loginForm.password),
{headers: new HttpHeaders({'Content-Type':
'text/plain'})}).subscribe((rs) => {
  localStorage.removeItem('token');
  localStorage.setItem('token', rs.token);
  localStorage.removeItem('phone');
  localStorage.setItem('phone', rs.phone);
  localStorage.removeItem('detailAddress');
  localStorage.setItem('detailAddress', rs.detailAddress);
  this.isLoggedIn = true;
  this.isAdmin = localStorage.getItem('token') !== null &&
atob(localStorage.getItem('token')).substring(atob(localStorage.g
etItem('token')).indexOf('+') + 1) === 'ADMIN';
  }, error => {
    console.log(`Error: ${error}`);
  }, () => {
    this.modalRef.hide();
  });
}

onLogout() {
  let tokenDto: TokenDto = {
    token: localStorage.getItem('token'),
    phone: localStorage.getItem('phone'),
    detailAddress: localStorage.getItem('detailAddress')
  };
  this.http.post<TokenDto>('/api/user/logout',
tokenDto).subscribe((rs) => {
    localStorage.removeItem('token');
    localStorage.setItem('token', rs.token);
    localStorage.removeItem('phone');
    localStorage.setItem('phone', rs.phone);
    localStorage.removeItem('detailAddress');
    localStorage.setItem('detailAddress', rs.detailAddress);
    this.isLoggedIn = false;
    this.isAdmin = false;
    this.router.navigate(['/shop']);
  }, error => {
    console.log(`Error: ${error}`);
  }, () => {
  });
}

openSignUpModal(template: TemplateRef<any>) {
  this.modalRef2 = this.modalService.show(template);
}

onCreateUser() {
  if (!this.inputValidator.isEmail(this.signUpForm.email) ||
!this.inputValidator.isPassword(this.signUpForm.password) ||

```

```

this.signUpForm.password !== this.signUpForm.rePassword ||
this.signUpForm.email !== this.signUpForm.reEmail ||
this.signUpForm.answer !== this.signUpForm.reAnswer) {
    this.wrongCreate = true;
    this.onRefreshSignUpForm();
    return;
}
this.wrongCreate = false;
this.http.post<User>('/api/user/create',
this.signUpForm).subscribe(() => {
    alert(this.translate_CREATE_USER_SUCCESSFUL);
    this.loginForm.email = this.signUpForm.email;
    this.loginForm.password = this.signUpForm.password;
    this.onLogin();
}, error => {
    console.log(`Error: ${error}`);
}, () => {
    this.modalRef2.hide();
});
}

openForgotPasswordModal(template: TemplateRef<any>) {
    this.modalRef2 = this.modalService.show(template);
}

onVerify() {
    if (!this.inputValidator.isEmail(this.forgotPasswordForm.email)
    ||
    !this.inputValidator.isPassword(this.forgotPasswordForm.password)
    || this.forgotPasswordForm.password !==
this.forgotPasswordForm.rePassword) {
        this.wrongForgot = true;
        this.onRefreshForgotPasswordForm();
        return;
    }
    this.wrongForgot = false;
    this.forgotPasswordForm.answer =
btoa(this.forgotPasswordForm.answer);
    this.forgotPasswordForm.password =
btoa(this.forgotPasswordForm.password);
    this.forgotPasswordForm.rePassword =
btoa(this.forgotPasswordForm.rePassword);
    this.http.post<TokenDto>('/api/user/resetPassword',
this.forgotPasswordForm).subscribe((rs) => {
        if (atob(rs.token) === '0+GUESS') {
            alert(this.translate_RESET_PASSWORD_FAILED);
        } else {
            alert(this.translate_RESET_PASSWORD_SUCCESSFUL);
            this.loginForm.email = this.forgotPasswordForm.email;
            this.loginForm.password =
atob(this.forgotPasswordForm.password);
            this.onLogin();
        }
    }, error => {
        console.log(`Error: ${error}`);
    }, () => {
        this.modalRef2.hide();
    });
}

```

```

openCartModal(template: TemplateRef<any>) {
    this.cartForm.phone = LocalStorage.getItem('phone');
    let detailAddress = LocalStorage.getItem('detailAddress');
    let endAddress = detailAddress.indexOf(', ');
    let endDistrict = detailAddress.lastIndexOf(', ');
    this.cartForm.address = detailAddress.substring(0, endAddress);
    this.cartForm.district = detailAddress.substring(endAddress +
2, endDistrict);
    this.cartForm.city = detailAddress.substring(endDistrict + 2);
    this.modalRef = this.modalService.show(template, {class:
'modal-lg'}));
}

cancelAndDecreaseItem(index: number) {
    if (index === -1) {
        this.totalPriceOfAddedProduct = 0;
        this.countAddedProduct = 0;
        this.addedProducts.length = 0;
        this.countOfIndividualProduct.length = 0;
        this.totalPriceOfIndividualProduct.length = 0;
        return;
    }
    this.totalPriceOfAddedProduct -=
this.addedProducts[index].price;
    if (this.countOfIndividualProduct[index] === 1) {
        this.addedProducts.splice(index, 1);
        this.countOfIndividualProduct.splice(index, 1);
        this.totalPriceOfIndividualProduct.splice(index, 1);
        --this.countAddedProduct;
    } else {
        --this.countOfIndividualProduct[index];
        this.totalPriceOfIndividualProduct[index] -=
this.addedProducts[index].price;
    }
}

increaseItem(index: number) {
    ++this.countOfIndividualProduct[index];
    this.totalPriceOfIndividualProduct[index] +=
this.addedProducts[index].price;
    this.totalPriceOfAddedProduct +=
this.addedProducts[index].price;
}

onSettle() {
    let bills: Bill[] = [];
    let today = new Date();
    let todayStr = today.getUTCFullYear() + '-' +
today.getUTCMonth() + '-' + today.getUTCDate() + ', ' +
today.getUTCHours() + ':' + today.getUTCMinutes() + ':' +
today.getUTCSeconds();
    let userId =
parseInt(atob(LocalStorage.getItem('token')).substr(0, 1));
    for (let i = 0; i < this.addedProducts.length; ++i) {
        let bill: Bill = {
            placementDate: todayStr,
            productId: this.addedProducts[i].id,
            productQuantity: this.countOfIndividualProduct[i],

```

```

        price: Math.ceil(this.totalPriceOfIndividualProduct[i]),
        userId: userId,
        settlementDate: todayStr,
        status: 'SUCCESS',
        phone: this.cartForm.phone,
        detailAddress: this.cartForm.address + ', ' +
this.cartForm.district + ', ' + this.cartForm.city
    };
    bills.push(bill);
}
this.http.post<Bill[]>('/api/bill', bills).subscribe(() => {
    alert(this.translate_ORDER_SUCCESSFUL);
}, error => {
    console.log(`Error: ${error}`);
}, () => {
    this.modalRef.hide();
});
}

onRefreshLoginForm() {
    this.loginForm = {
        email: '',
        password: ''
    };
}

onRefreshSignUpForm() {
    this.signUpForm = {
        name: '',
        email: '',
        reEmail: '',
        password: '',
        rePassword: '',
        answer: '',
        reAnswer: '',
        phone: '',
        address: '',
        district: 'Bình Thạnh',
        city: 'Hồ Chí Minh'
    };
}

onRefreshForgotPasswordForm() {
    this.forgotPasswordForm = {
        email: '',
        answer: '',
        password: '',
        rePassword: ''
    };
}

scrollTop() {
    window.scrollTo(0, 0);
}

scrollBottom() {
    window.scrollTo(0, document.body.scrollHeight);
}

```

```

scrollLeft() {
    window.scrollTo(0, window.pageYOffset);
}

scrollRight() {
    window.scrollTo(document.body.scrollWidth, window.pageYOffset);
}
}

interface City {
    name: string;
}

interface District {
    name: string;
    cityName: string;
}

interface TokenDto {
    token: string;
    phone: string;
    detailAddress: string;
}

```

## V. Hướng dẫn cài đặt:

0. Máy tính môi trường phát triển (*development*) cần cài trước: JDK 11, NodeJS và IntelliJ.
1. Mở thư mục gốc trong ‘IntelliJ IDEA’.
2. Chỉnh sửa tệp ‘*application.properties*’ (tài khoản và phiên bản của máy chủ MySQL) theo cơ sở dữ liệu.
3. Chạy 2 dòng đầu tiên trong tệp ‘*db / run.sql*’ để tạo tên cơ sở dữ liệu (schema)

‘*flowershop*’ trong Máy chủ MySQL.

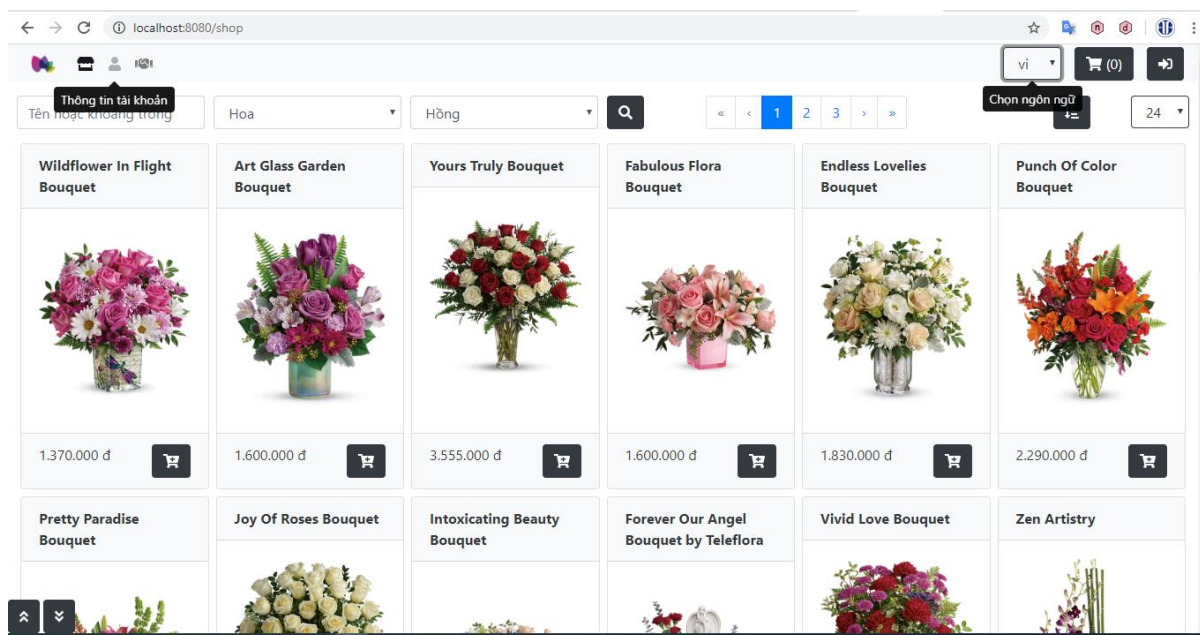
4. Khởi động Spring Boot ‘*FlowershopApplication*’. Các bảng sẽ được tạo nếu chạy lần đầu tiên.
5. Chạy phần còn lại trong tệp ‘*db / run.sql*’ để điền dữ liệu mặc định cho ứng dụng của chúng tôi.
6. Trong ‘Postman’, gọi ‘POST’ tại ‘*http://localhost:8080/api/test*’ với ‘body JSON’ được sao chép từ ‘*db / mock-test.json*’.
7. Trong ‘Postman’, hãy gọi ‘POST’ tại ‘*http://localhost:8080/api/product*’ với ‘body JSON’ được sao chép từ ‘*db / product.json*’.

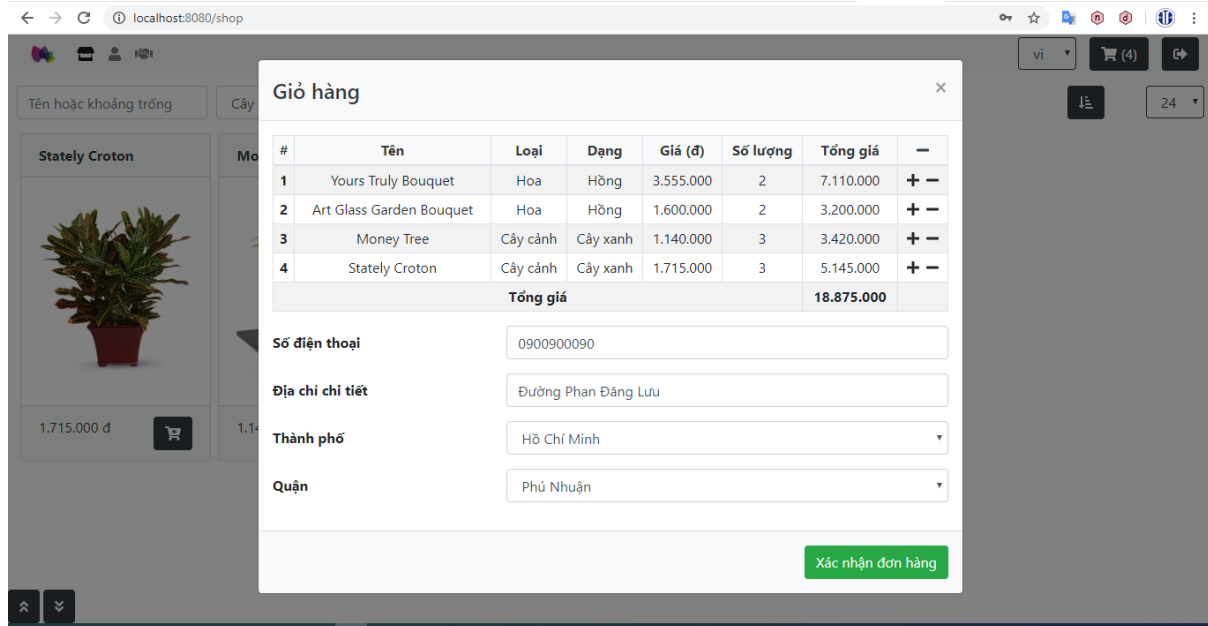
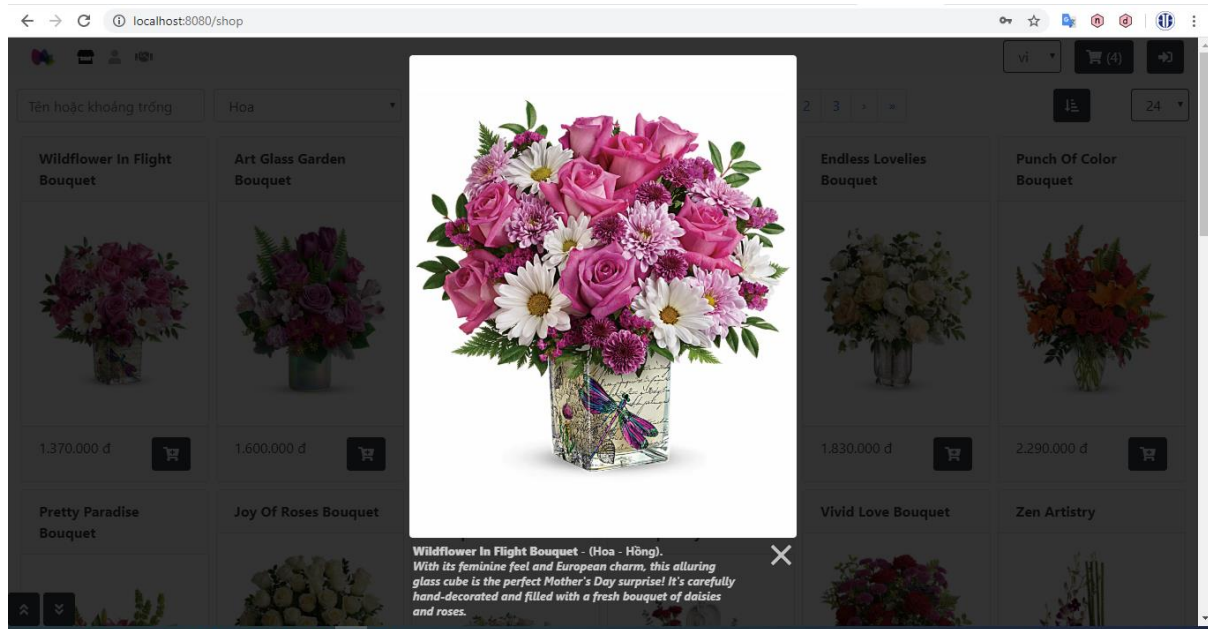


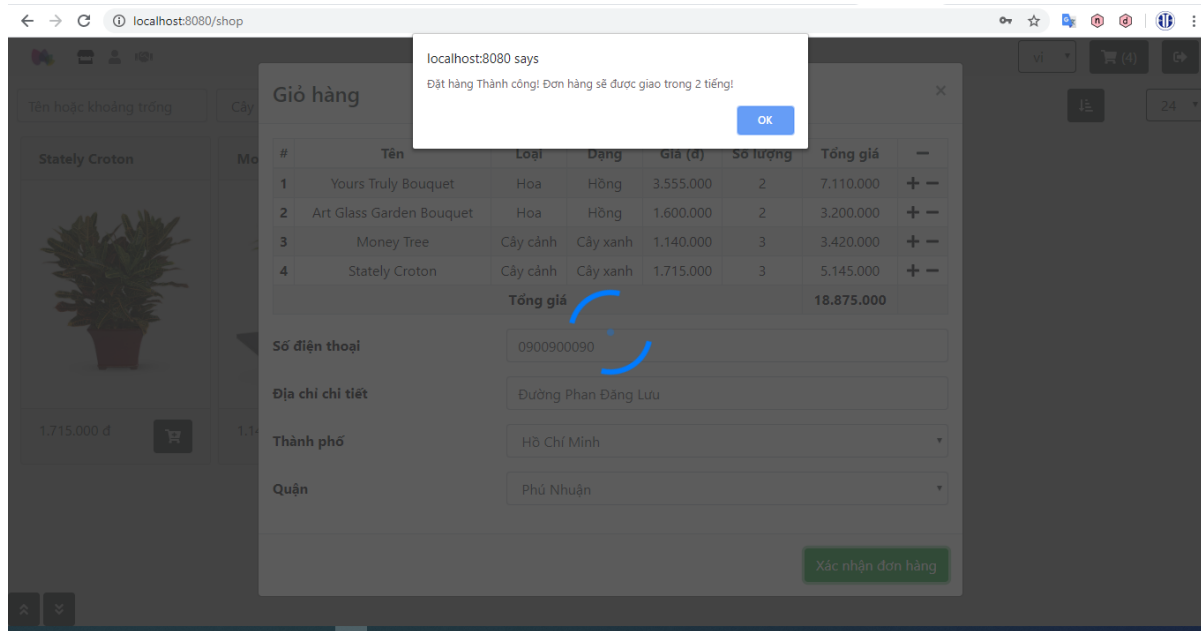
8. Chạy `'npm install'` trong `'cmd'` (hoặc `'terminal'`) trong thư mục `'frontend'`.
9. Mở `'frontend / package.json'` và nhấp vào `'green arrow'` bên cạnh `'start-dev'` để chạy `'frontend'` trong chế độ phát triển.
10. Mở trình duyệt tại `'http://localhost:4200'` để truy cập ứng dụng web.
11. Môi trường *production* chỉ cần cài đặt `'JDK 11'`, `'Maven'` và `'MySQL Server 8'`. Sau đó thiết lập `'database'` và `'application.properties'` đúng như trong `'Môi trường phát triển (development)'`.
12. Chạy `'mvn package'` trong `'cmd'` trong thư mục gốc. Điều này sẽ tạo ra một gói `'Jar'` trong thư mục `'target'` có tên là `'flowershop-1.0.jar'`.
13. Chạy gói `'Jar'` này trong `'cmd'` bằng lệnh này `'java -jar target/flowershop-1.0.jar'`.
14. Truy cập ứng dụng web thông qua điểm cuối, ví dụ như `'AWS Cloud Endpoint'`.

## VI. Hướng dẫn sử dụng và trưng bày sản phẩm bằng hình ảnh:

### 1. Đối với khách hàng:





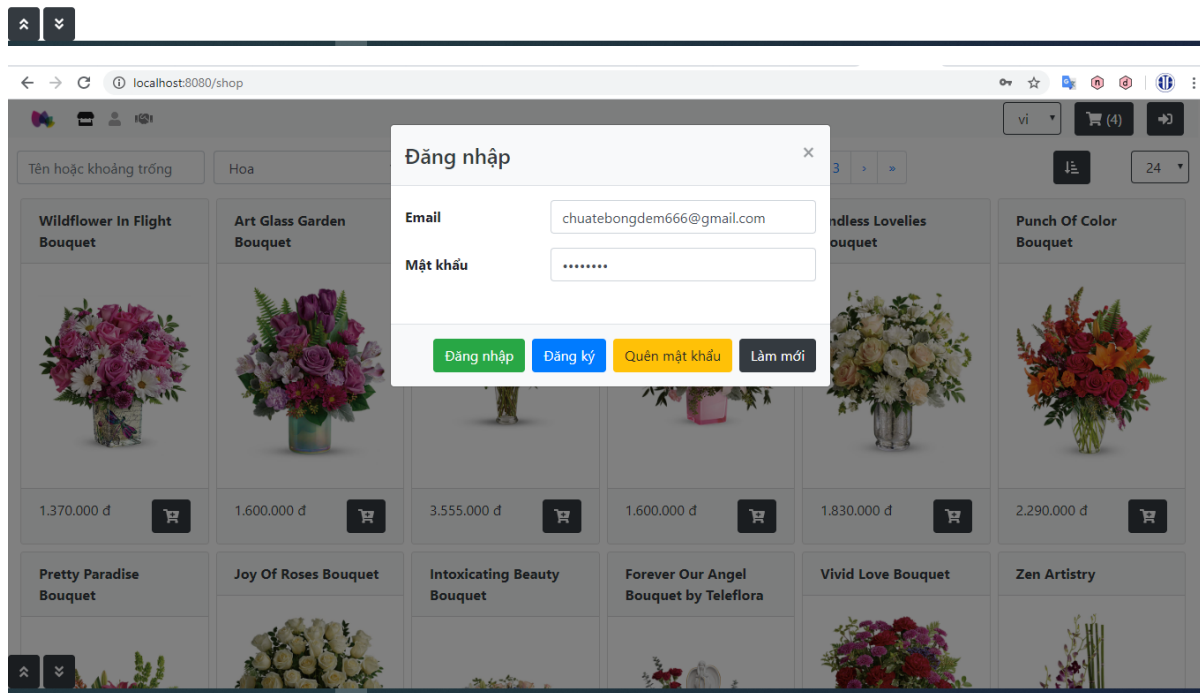


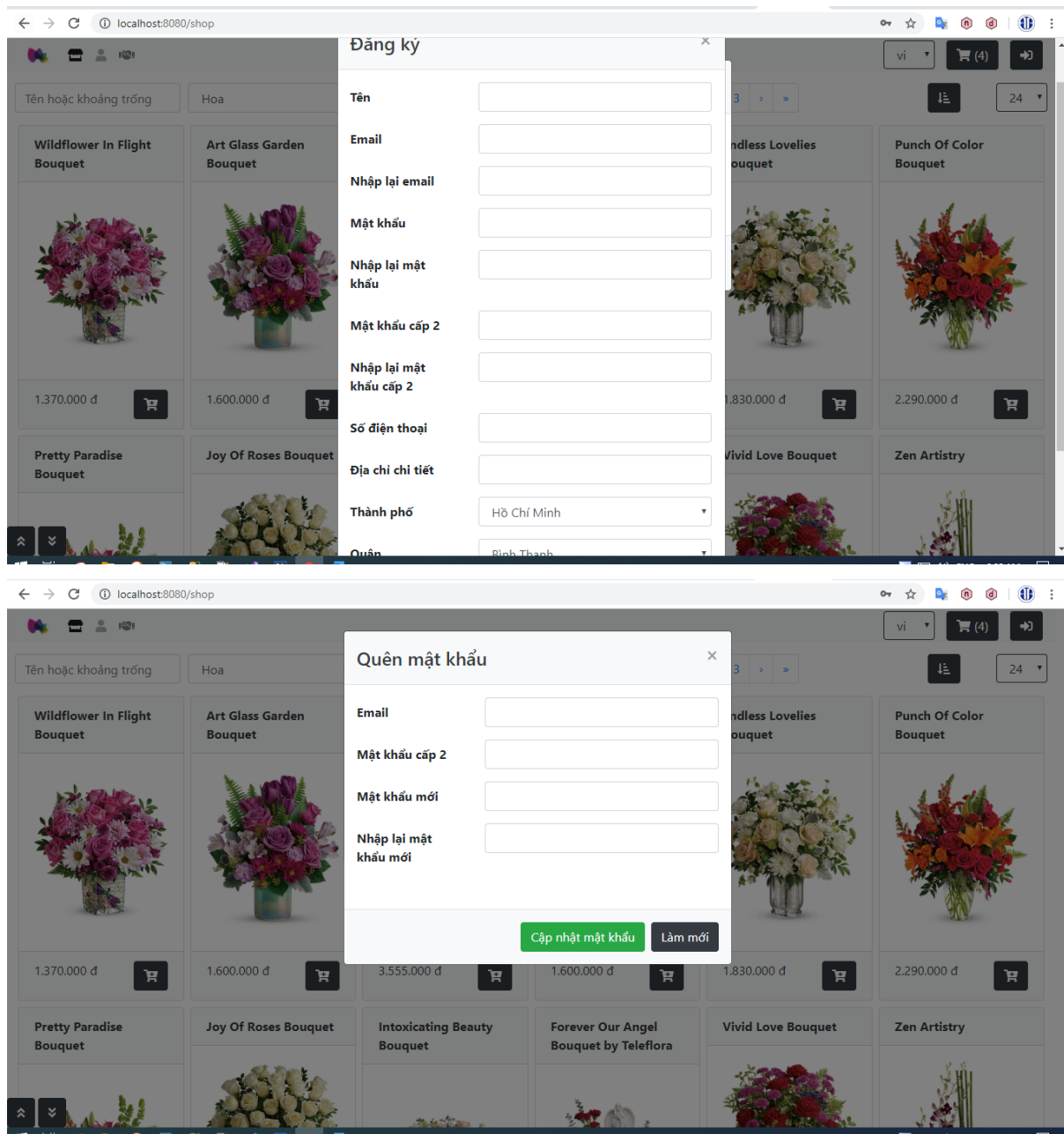
Tên	Thông tin tài khoản	chuatebongdem
Email		chuatebongdem666@gmail.com
Số điện thoại		0900900090
Địa chỉ chi tiết		Đường Phan Đăng Lưu, Phú Nhuận

#	Tên	Loại	Dạng	Giá (đ)	Số lượng	Tổng giá	Ngày đặt
1	Art Glass Garden Bouquet	Hoa	Hồng	1.600.000	4	6.400.000	2019-11-25, 14:48:15
2	Yours Truly Bouquet	Hoa	Hồng	3.555.000	3	10.665.000	2019-11-25, 14:48:15
3	Money Tree	Cây cảnh	Cây xanh	1.140.000	2	2.280.000	2019-11-25, 14:48:15
4	Stately Croton	Cây cảnh	Cây xanh	1.715.000	2	3.430.000	2019-11-25, 14:48:15
5	Wildflower In Flight Bouquet	Hoa	Hồng	1.370.000	4	5.480.000	2019-11-25, 14:53:27
6	Art Glass Garden Bouquet	Hoa	Hồng	1.600.000	6	9.600.000	2019-11-25, 14:53:27
7	Yours Truly Bouquet	Hoa	Hồng	3.555.000	4	14.220.000	2019-11-25, 14:53:27
8	Fabulous Flora Bouquet	Hoa	Hồng	1.600.000	4	6.400.000	2019-11-25, 14:53:27
9	Punch Of Color Bouquet	Hoa	Hồng	2.290.000	3	6.870.000	2019-11-25, 14:53:27
10	Art Glass Garden Bouquet	Hoa	Hồng	1.600.000	3	4.800.000	2019-11-25, 15:15:9
11	Yours Truly Bouquet	Hoa	Hồng	3.555.000	4	14.220.000	2019-11-25, 15:15:9
12	Wildflower In Flight Bouquet	Hoa	Hồng	1.370.000	3	4.110.000	2019-11-25, 15:15:9
13	Fabulous Flora Bouquet	Hoa	Hồng	1.600.000	2	3.200.000	2019-11-25, 15:15:9
Tổng giá						91.675.000	

Dự án của lavantien @12/2019 - mã nguồn ở: <https://github.com/lavantien/flowershop>


































## 2. Đối với người quản lý:

localhost:8080/admin

Kho hàng

Tên hoặc khoảng trống Hoa Hồng

#	Tên	Mô tả	Giá (đ)	Ảnh	Tồn kho	Đã bán	Loại	Dạng	Hành động
1	Wildflower In Flight Bouquet	With its feminine feel and European charm, this al...	1.370.000		10	0	Hoa	Hồng	 
2	Art Glass Garden Bouquet	The shimmering, iridescent aqua blue glass of this...	1.600.000		10	0	Hoa	Hồng	 
3	Yours Truly Bouquet	A truly breathtaking tribute to your love, this ro...	3.555.000		10	0	Hoa	Hồng	 
4	Fabulous Flora Bouquet	Just fabulous! From its perky pink cube and perfec...	1.600.000		10	0	Hoa	Hồng	 
5	Endless Lovelies Bouquet	Timeless and touching, this serenely beautiful bou...	1.830.000		10	0	Hoa	Hồng	 
6	Punch Of Color Bouquet	A punch of sunset-inspired color, this bold, beaut...	2.290.000		10	0	Hoa	Hồng	 
7	Pretty Paradise Bouquet	Send that special someone on a trip to paradise, n...	5.855.000		10	0	Hoa	Hồng	 
8	Joy Of Roses Bouquet	Pure joy! Two dozen wondrous white roses take cent...	2.865.000		10	0	Hoa	Hồng	 
9	Intoxicating Beauty Bouquet	Luxuriate in the natural, intoxicating beauty of t...	6.315.000		10	0	Hoa	Hồng	 

localhost:8080/admin

Tên hoặc khoảng trống Hoa

Tạo sản phẩm mới

Tên

Mô tả

Giá (đ)

Ảnh

Tồn kho



















Đã bán

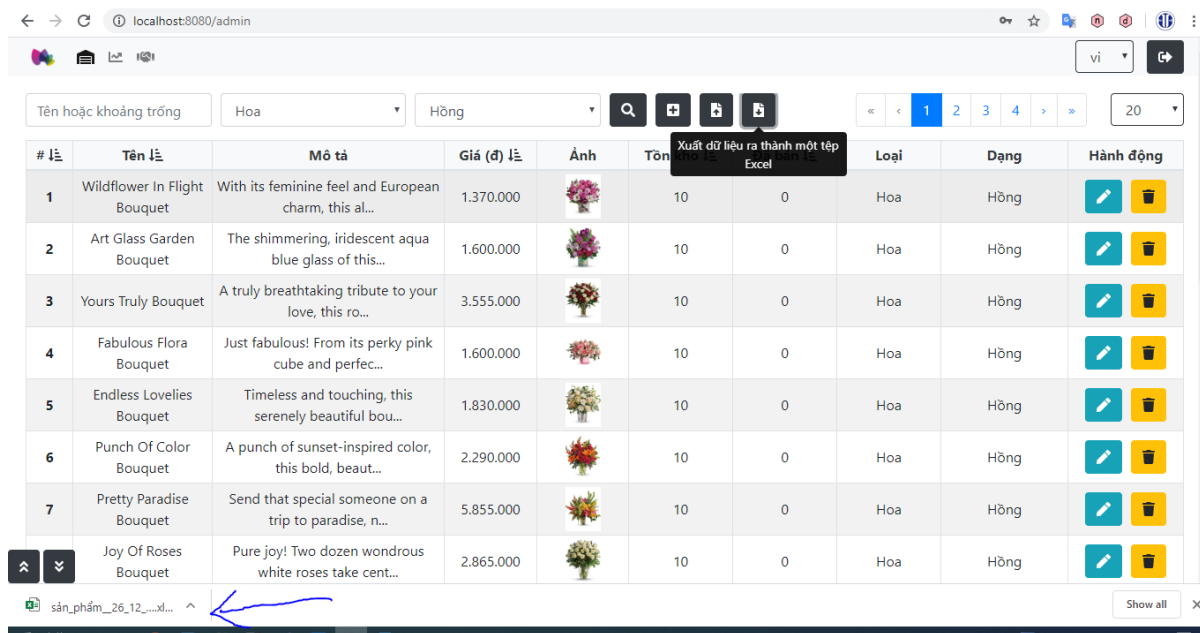
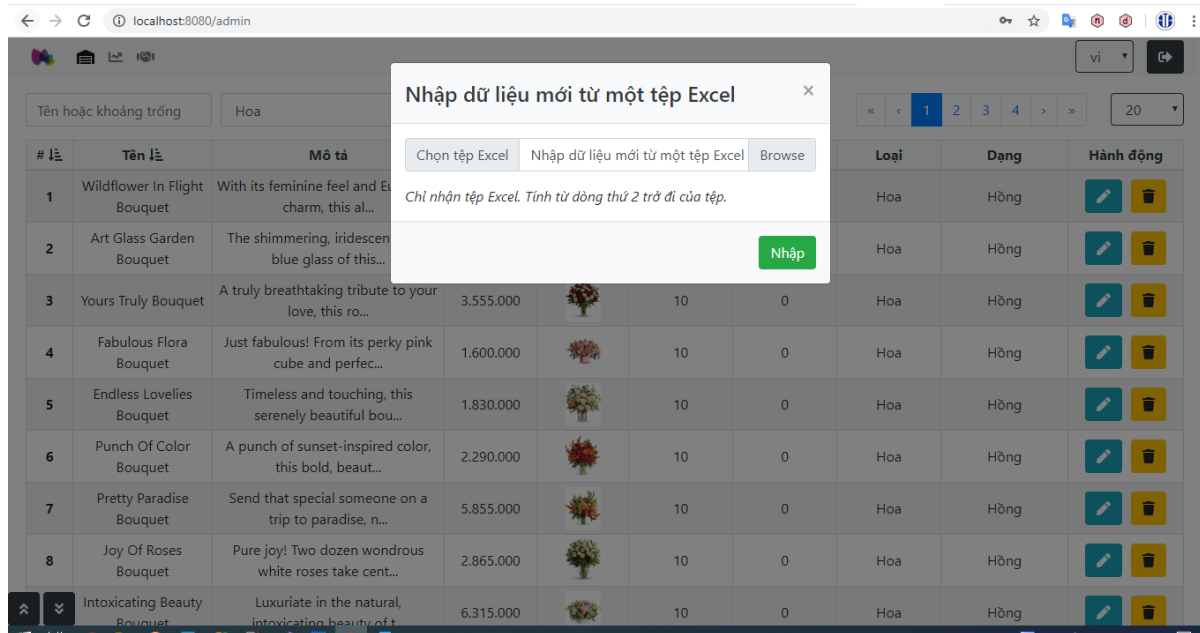
Loại

Dạng

Lưu

Làm mới

Loại	Dạng	Hành động
Hoa	Hồng	 
Hoa	Hồng	 
Hoa	Hồng	 
Hoa	Hồng	 
Hoa	Hồng	 
Hoa	Hồng	 
Hoa	Hồng	 
Hoa	Hồng	 
Hoa	Hồng	 









- Giao diện tối giản hiện đại. Trang trưng bày hàng hóa hỗ trợ tốt responsive nhờ ứng dụng triệt để Bootstrap cho khách hàng truy cập trang web từ điện thoại.
- Đã vận dụng thành công những kiến thức đã học trên lớp và tìm tòi qua internet vào một sản phẩm hoàn chỉnh.
- Tiến bộ hơn trong kỹ năng làm việc nhóm, chuẩn bị, lên kế hoạch cũng như thuyết trình.

## **2. Hướng phát triển của phần mềm:**

- Thêm vào tính năng thanh toán qua ví điện tử hoặc liên kết tài khoản ngân hàng.
- Hỗ trợ theo dõi sản phẩm mới bằng email.
- Thêm vào trang quản lý người dùng cho người quản lý có thể dễ dàng block hoặc theo dõi các tài khoản đã đăng ký.

## **VIII. Tài liệu tham khảo**

- MySQL Documents: <https://dev.mysql.com/doc/refman/8.0/en/>
- Spring Boot Documents: <https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/>
- Spring Data JPA API Documents: <https://docs.spring.io/spring-data/jpa/docs/current/api/org/springframework/data/jpa/repository/JpaRepository.html>
- Angular Documents: <https://angular.io/docs>
- Bootstrap Documents: <https://getbootstrap.com/docs/4.4/getting-started/introduction/>
- NgxBootstrap Documents: <https://valor-software.com/ngx-bootstrap/#/documentation>
- Những nơi khác như Google, Youtube, StackOverflow và Medium.