KITCHEN STORY SOURCE CODE

Prepared By: SHUBHAM MEHTA

Front-end/Kitchen-Story

index.html

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">

  <meta name="viewport" content="width=device-width,
initial-scale=1, shrink-to-fit=no">

<link rel="stylesheet"
href="assets/fonts/ionicons.min.css">
<link rel="stylesheet" href="assets/fonts/font-
awesome.min.css">
<link rel="stylesheet"
href="assets/bootstrap/css/bootstrap.min.css">
<script
src="assets/bootstrap/js/bootstrap.min.js"></script>
<link rel="stylesheet" href="assets/fonts/material-
icons.min.css">

  <title>Kitchen Story</title>
  <base href="/">

</head>
<body>
  <app-root></app-root>
</body>
</html>
```

Login.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Login</title>
 <link type="text/css" rel="stylesheet"
href="resources/css/login.css">

</head>
<body>

  <center> <h1> Login </h1> </center>
    <form action="loginProcess" method="POST">
        <div class="container">
            <input type="hidden" name="command"
value="LOGIN" />
            <label>Username : </label>
            <br/>
            <input type="text" placeholder="Enter
Username" name="username" required>
            <br/>
            <label>Password : </label>
            <br/>
            <input type="password" placeholder="Enter
Password" name="password" required>
            <br/>
            <button type="submit">Login</button>
            <br/>
```

```
            <a
href="${pageContext.request.contextPath}/register">Don't
have an account? Register now!</a>

        </div>
    </form>


</body>
</html>
```

Add-product.component.html

```
br><br>
<div class="login-form">
    <form [formGroup]="checkoutFormGroup"
(ngSubmit)="onSubmit()" >
        <h2 class="text-center">Add Product</h2>
        <div formGroupName="product">
        <div class="form-group">
            <input formControlName="name" type="text"
class="form-control" placeholder="Name"
required="required">
        </div>
        <div class="form-group">
            <input  formControlName="description"
type="text" class="form-control"
placeholder="Description" required="required">
        </div>
        <div class="form-group">
```

```
            <input formControlName="unitPrice"
type="text" class="form-control" placeholder="Price"
required="required">
        </div>
        <div class="form-group">
            <input formControlName="imageUrl"
type="text" class="form-control" placeholder="Image URL"
required="required">
        </div>


        <div class="input-space">
            <select formControlName="category">
                <option readonly="true"
 selected>Categoery</option>
                <option *ngFor="let category1 of
productCategories" [ngValue]="category" >
                    {{ category1.categoryName }}
                </option>
            </select>
        </div>

        <div class="form-group">
            <button type="submit" class="btn btn-success
btn-block">Add!</button>
        </div>


    </div>
    </form>
```

```
</div>
```

Cart-details.component.html

```
<div class="main-content">
    <div class="section-content section-content-p30">
        <div class="container-fluid">

            <div *ngIf="cartItems.length > 0">

                <table class="table table-bordered">
                    <tr>
                        <th width="20%">Product
Image</th>
                        <th width="50%">Product
Detail</th>
                        <th width="30%">Action</th>
                    </tr>

                    <tr *ngFor="let tempCartItem of
cartItems">
                        <td>
                            <img src="{{
tempCartItem.imageUrl }}" class="img-responsive"
width="150px" />
                        </td>
                        <td>
                            <p>{{ tempCartItem.name
}}</p>
```

```
                            <p>{{ tempCartItem.unitPrice
| currency: 'USD' }}</p>
                        </td>
                        <td>


                            <button
(click)="remove(tempCartItem)" class="btn btn-danger
btn-sm mt-2">Remove</button>


                        </td>
                    </tr>

                    <tr>
                        <td colspan="2"></td>
                        <td style="font-weight: bold">
                            <p>Total Quantity: {{
totalQuantity }}</p>

                            <p>Total Price: {{
totalPrice | currency: 'USD' }}</p>
                        </td>
                    </tr>

                </table>

            </div>

            <!-- if cart is empty then display a message
-->
```

```
            <div *ngIf="cartItems.length == 0"
class="alert alert-warning col-md-12" role="alert">
                Your shopping cart is empty.
            </div>

        </div>
    </div>
</div>
```

Product-category-menu-component.html

```
<div class="menu-sidebar-content js-scrollbar1">
    <nav class="navbar-sidebar">
        <ul class="list-unstyled navbar-list">
            <li>
                <a routerLink="/products"
routerLinkActive="active-link">
                    All
                </a>
            </li>

                <li *ngFor="let tempProductCategory of
productCategories">

                    <a routerLink="/category/{{
tempProductCategory.id }}" routerLinkActive="active-
link">
                        {{
tempProductCategory.categoryName }}
                    </a>
```

```
                </li>

                <li>
                <a *ngIf="username == 'admin'"
 routerLink="/add-product">
                    <button style="font-size: 12px;"
type="button" class="btn btn-primary"> Add
Product</button>
                </a>

                 </li>


            </ul>
    </nav>
</div>
```

Product-details-component.html

```
<div class="detail-section">
    <div class="container-fluid">
<!--  -->
        <img src="{{ product.imageUrl }}" class="detail-
img">

        <h3>{{ product.name }}</h3>
        <div class="price">{{ product.unitPrice |
currency:'USD' }}</div>
        <button (click)="addToCart()" class="btn btn-
primary btn-sm">Add to cart</button>
```

```
        <hr>
        <h4>Description</h4>
        <p>{{ product.description }}</p>

        <a routerLink="/products" class="mt-5">Back to
Product List</a>
        {{username}}
    </div>
</div>
```

Product=list-component.html

```
<p *ngFor="let tempProduct of products">
    {{ tempProduct.name}}: {{ tempProduct.unitPrice |
currency:'USD'}}
</p>
```

Register.component.html

```
<br><br>
<div class="login-form">
    <form action="/examples/actions/confirmation.php"
method="post">
        <h2 class="text-center">Register</h2>
        <div class="form-group">
            <input type="text" class="form-control"
placeholder="Username" required="required">
        </div>
        <div class="form-group">
```

```
            <input type="text" class="form-control"
placeholder="Email" required="required">
        </div>
        <div class="form-group">
            <input type="password" class="form-control"
placeholder="Password" required="required">
        </div>
        <div class="form-group">
            <button type="submit" class="btn btn-primary
btn-block">Register</button>
        </div>

    </form>
    <p class="text-center"><a routerLink="/login">Back
to Login</a></p>
</div>
```

Serach.component.html

```
<div class="form-header">
<!--  -->
    <input #myInput type="text"
           placeholder="Search for products ..."
           class="au-input au-input-xl"
           (keyup.enter)="doSearch(myInput.value)"  />

    <button (click)="doSearch(myInput.value)" class="au-
btn-submit">
        Search
    </button>
```

```
</div>
```

Update-account.component.html

```
<br><br>
<div class="login-form">
    <form [formGroup]="checkoutFormGroup" >
        <h2 class="text-center">Update Account</h2>

        <div formGroupName="user">
        <div class="form-group">
            <input value={{username}}
formControlName="username" type="text" class="form-
control" placeholder="Username" required="required">
        </div>
        <div class="form-group">
            <input value={{this.users[0].email}}
formControlName="email" type="text" class="form-control"
placeholder="Email" required="required">
        </div>
        <div class="form-group">
            <input value={{this.users[0].password}}
formControlName="password1" type="text" class="form-
control" placeholder="Password" required="required">
        </div>
        <div class="form-group">
            <button type="submit" class="btn btn-success
btn-block">Update</button>
        </div>
    </div>
    </form>
```

```
</div>
```

Back-end /Kitchen-Story

MyDataRestConfig.java

```
package com.simplilearn.KitchenStory.config;


import com.simplilearn.KitchenStory.entity.Product;
import
com.simplilearn.KitchenStory.entity.ProductCategory;
import
org.springframework.beans.factory.annotation.Autowired;
import
org.springframework.context.annotation.Configuration;
import
org.springframework.data.rest.core.config.RepositoryRest
Configuration;
import
org.springframework.data.rest.webmvc.config.RepositoryRe
stConfigurer;
import org.springframework.http.HttpMethod;
import
org.springframework.web.servlet.config.annotation.CorsRe
gistry;

import javax.persistence.EntityManager;
```

```
import javax.persistence.metamodel.EntityType;
import java.util.ArrayList;
import java.util.List;
import java.util.Set;
@Configuration
public class MyDataRestConfig implements
RepositoryRestConfigurer {

    private EntityManager;

    @Autowired
    public MyDataRestConfig(EntityManager
theEntityManager) {
        entityManager = theEntityManager;
    }

    @Override
    public void
configureRepositoryRestConfiguration(RepositoryRestConfi
guration config, CorsRegistry cors) {

        HttpMethod[] theUnsupportedActions =
{HttpMethod.PUT, HttpMethod.POST, HttpMethod.DELETE};

        // disable HTTP methods for Product: PUT, POST
and DELETE
        config.getExposureConfiguration()
                .forDomainType(Product.class)
                .withItemExposure((metdata, httpMethods)
-> httpMethods.disable(theUnsupportedActions))
```

```
                .withCollectionExposure((metdata,
httpMethods) ->
httpMethods.disable(theUnsupportedActions));

        // disable HTTP methods for ProductCategory:
PUT, POST and DELETE
        config.getExposureConfiguration()
                .forDomainType(ProductCategory.class)
                .withItemExposure((metdata, httpMethods)
-> httpMethods.disable(theUnsupportedActions))
                .withCollectionExposure((metdata,
httpMethods) ->
httpMethods.disable(theUnsupportedActions));

        // call an internal helper method
        exposeIds(config);
    }

    private void exposeIds(RepositoryRestConfiguration
config) {

        // expose entity ids
        //

        // - get a list of all entity classes from the
entity manager
        Set<EntityType<?>> entities =
entityManager.getMetamodel().getEntities();

        // - create an array of the entity types
        List<Class> entityClasses = new ArrayList<>();
```

```
        // - get the entity types for the entities
        for (EntityType tempEntityType : entities) {

entityClasses.add(tempEntityType.getJavaType());
        }

        // - expose the entity ids for the array of
entity/domain types
        Class[] domainTypes = entityClasses.toArray(new
Class[0]);
        config.exposeIdsFor(domainTypes);
    }
}
```

ProductController.java

```
package com.simplilearn.KitchenStory.controller;



import com.simplilearn.KitchenStory.entity.Product;
import
com.simplilearn.KitchenStory.service.ProductService;
import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import
org.springframework.web.bind.annotation.GetMapping;
```

```java
import
org.springframework.web.bind.annotation.ModelAttribute;
import
org.springframework.web.bind.annotation.PostMapping;
import
org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.*;

@CrossOrigin(origins= {"*"}, maxAge = 4800,
allowCredentials  =  "false")
@RestController
@RequestMapping("/products")
public class ProductController {


    private ProductService productService;

    public ProductController(){

    }


    public ProductController(ProductService
theProductService) {
        productService = theProductService;
    }


    @PostMapping("/save")
```

```
    public String
saveEmployee(@ModelAttribute("product") Product
theProduct) {

        // save the employee
        productService.save(theProduct);

        // use a redirect to prevent duplicate
submissions
        return "redirect:/products";
    }

}
```

ProductCategoryRepository.java

```
package com.simplilearn.KitchenStory.dao;



import
com.simplilearn.KitchenStory.entity.ProductCategory;
import
org.springframework.data.jpa.repository.JpaRepository;
import
org.springframework.data.rest.core.annotation.Repository
RestResource;
import
org.springframework.web.bind.annotation.CrossOrigin;

@CrossOrigin("http://localhost:4200")
```

```
@RepositoryRestResource(collectionResourceRel =
"productCategory", path = "product-category")
public interface ProductCategoryRepository extends
JpaRepository<ProductCategory, Long> {
}
```

UserRepository.java

```
package com.simplilearn.KitchenStory.dao;


import com.simplilearn.KitchenStory.entity.Product;
import com.simplilearn.KitchenStory.entity.User;

import org.springframework.data.domain.Page;
import org.springframework.data.domain.Pageable;
import
org.springframework.data.jpa.repository.JpaRepository;
import
org.springframework.web.bind.annotation.CrossOrigin;
import
org.springframework.web.bind.annotation.RequestParam;


@CrossOrigin("http://localhost:4200")
public interface UserRepository extends
JpaRepository<User, Long> {

        Page<User>
findByUsername(@RequestParam("username") String
username, Pageable pageable);
```

```
}
```

Product.java

```
package com.simplilearn.KitchenStory.entity;


import lombok.Data;
import org.hibernate.annotations.CreationTimestamp;
import org.hibernate.annotations.UpdateTimestamp;

import javax.persistence.*;
import java.math.BigDecimal;
import java.util.Date;
@Entity
@Table(name="product")
@Data
public class Product {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "id")
    private Long id;

    @ManyToOne
    @JoinColumn(name = "category_id", nullable = false)
    private ProductCategory category;


    @Column(name = "name")
```

```
    private String name;

    @Column(name = "description")
    private String description;

    @Column(name = "unit_price")
    private BigDecimal unitPrice;

    @Column(name = "image_url")
    private String imageUrl;



    @Column(name = "date_created")
    @CreationTimestamp
    private Date dateCreated;

    @Column(name = "last_updated")
    @UpdateTimestamp
    private Date lastUpdated;
}
```

ProductCategory.java

```
package com.simplilearn.KitchenStory.entity;

import lombok.Getter;
import lombok.Setter;

import javax.persistence.*;
```

```java
import java.util.Set;

@Entity
@Table(name="product_category")
// @Data -- known bug
@Getter
@Setter
public class ProductCategory {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "id")
    private Long id;

    @Column(name = "category_name")
    private String categoryName;

    @OneToMany(cascade = CascadeType.ALL, mappedBy =
"category")
    private Set<Product> products;

}
```

User.java

```java
package com.simplilearn.KitchenStory.entity;

import lombok.Data;
import org.hibernate.annotations.CreationTimestamp;
import org.hibernate.annotations.UpdateTimestamp;
```

```java
import javax.persistence.*;
import java.math.BigDecimal;
import java.util.Date;
@Entity
@Table(name="users")
@Data
public class User {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "id")
    private Long id;

    @Column(name = "username")
    private String username;

    @Column(name = "password")
    private String password;

    @Column(name = "email")
    private String email;

    @Column(name = "type")
    private int type;

}
```

ProductService.java

```java
package com.simplilearn.KitchenStory.service;
```

```
import com.simplilearn.KitchenStory.entity.Product;

public interface ProductService {

    public void save(Product theProduct);


}
```

ProductServiceImpl.java

```
package com.simplilearn.KitchenStory.service;

import
com.simplilearn.KitchenStory.dao.ProductRepository;
import com.simplilearn.KitchenStory.entity.Product;
import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import
org.springframework.web.bind.annotation.CrossOrigin;

@CrossOrigin("http://localhost:4200")
public class ProductServiceImpl implements
ProductService {

    private ProductRepository productRepository;

    @Autowired
```

```
    public ProductServiceImpl(ProductRepository
theProductRepository) {

        productRepository = theProductRepository;
    }



    @Override
    public void save(Product theProduct) {
        productRepository.save(theProduct);
    }
}
```

KitchenStoryApplication.java

```
package com.simplilearn.KitchenStory;

import org.springframework.boot.SpringApplication;
import
org.springframework.boot.autoconfigure.SpringBootApplica
tion;

@SpringBootApplication
public class KitchenStoryApplication {

    public static void main(String[] args) {

SpringApplication.run(KitchenStoryApplication.class,
args);
```

```
    }

}
```

Application.properties

```
spring.datasource.driver-class-
name=com.mysql.cj.jdbc.Driver
spring.datasource.url=jdbc:mysql://localhost:3307/kitche
n-story
spring.datasource.username=root
spring.data.rest.base-path=/api
```

KitchenStoryApplicationest.java

```
package com.simplilearn.KitchenStory;

import org.junit.jupiter.api.Test;
import
org.springframework.boot.test.context.SpringBootTest;

@SpringBootTest
class KitchenStoryApplicationTests {

    @Test
    void contextLoads() {
    }
}
```

Pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-
parent</artifactId>
        <version>2.5.2</version>
        <relativePath/> <!-- lookup parent from
repository -->
    </parent>
    <groupId>com.simplilearn</groupId>
    <artifactId>Kitchen-Story</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <name>Kitchen-Story</name>
    <description>Demo project for Spring
Boot</description>
    <properties>
        <java.version>11</java.version>
    </properties>
    <dependencies>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-data-
jpa</artifactId>
        </dependency>
```

```
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-data-
rest</artifactId>
        </dependency>



        <dependency>
            <groupId>mysql</groupId>
            <artifactId>mysql-connector-
java</artifactId>
            <scope>runtime</scope>
        </dependency>
        <dependency>
            <groupId>org.projectlombok</groupId>
            <artifactId>lombok</artifactId>
            <optional>true</optional>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-
test</artifactId>
            <scope>test</scope>
        </dependency>
    </dependencies>

    <build>
        <plugins>
            <plugin>

<groupId>org.springframework.boot</groupId>
```

```
                <artifactId>spring-boot-maven-
plugin</artifactId>
                <configuration>
                    <excludes>
                        <exclude>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>
                        </exclude>
                    </excludes>
                </configuration>
            </plugin>
        </plugins>
    </build>

</project>
```

Kitchen-story.sql

```
-- phpMyAdmin SQL Dump
-- version 5.1.0
-- https://www.phpmyadmin.net/
--
-- Host: 127.0.0.1:3307
-- Generation Time: Jul 14, 2021 at 11:38 PM
-- Server version: 10.4.18-MariaDB
-- PHP Version: 8.0.3

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
```

```
START TRANSACTION;
SET time_zone = "+00:00";


/*!40101 SET
@OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET
@OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET
@OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- Database: `kitchen-story`
--

-- --------------------------------------------------------
---

--
-- Table structure for table `product`
--

CREATE TABLE `product` (
  `id` int(20) NOT NULL,
  `name` varchar(255) NOT NULL,
  `description` varchar(500) DEFAULT NULL,
  `unit_price` decimal(15,0) DEFAULT NULL,
  `image_url` varchar(255) DEFAULT NULL,
  `date_created` datetime(6) DEFAULT NULL,
  `last_updated` datetime(6) DEFAULT NULL,
  `category_id` bigint(20) NOT NULL
```

```
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--
-- Dumping data for table `product`
--

INSERT INTO `product` (`id`, `name`, `description`,
`unit_price`, `image_url`, `date_created`,
`last_updated`, `category_id`) VALUES
(1, 'Apple', 'An apple is an edible fruit produced by an
apple tree. Apple trees are cultivated worldwide and are
the most widely grown species in the genus Malus', '20',
'https://cdn.picpng.com/apple/apple-view-25231.png',
'2021-07-13 08:26:21.000000', '2021-07-13
11:32:47.000000', 1),
(2, 'Orange', 'The vitamin C in oranges helps your body
in lots of ways:\r\nProtects your cells from
damage.\r\nHelps your body make collagen, a protein that
heals wounds and gives you smoother skin.', '24',
'https://cdn.pixabay.com/photo/2016/02/23/17/42/orange-
1218158_960_720.png', '2021-07-23 11:33:01.000000',
'2021-07-12 11:32:52.000000', 1),
(3, 'Pears', 'Pears are fruits produced and consumed
around the world, growing on a tree and harvested in
late Summer into October. The pear tree and shrub are a
species of genus Pyrus, ', '11',
'https://toppng.com/uploads/preview/pear-fruits-
11528330750x9tey9x2an.png', '2021-07-22
11:13:25.000000', '2021-07-05 11:32:56.000000', 1),
(4, 'onion', 'onion nice', '14',
'https://toppng.com/uploads/preview/brown-onion-
```

```
115283502637y3hhhogtu.png', '2021-07-15
23:17:00.000000', '2021-07-13 23:17:03.000000', 3),
(5, 'potato', 'potato', '53',
'https://toppng.com/uploads/preview/otato-free-download-
png-potato-11563264419umm60g6okx.png', '2021-07-15
23:17:00.000000', '2021-07-15 23:17:00.000000', 3),
(7, 'Rice', 'Rice', '34',
'https://toppng.com/uploads/preview/white-basmati-
jasmine-transprent-cooked-rice-
11563246781w8drsgmryp.png', '2021-07-15
23:17:00.000000', '2021-07-13 23:17:03.000000', 4),
(8, 'wheat', 'wheat', '35',
'http://masalakada.com/uploads/products/162220605311.jpg
', '2021-07-15 23:17:00.000000', '2021-07-13
23:17:03.000000', 4),
(9, 'fish ', 'fish ', '56',
'https://purepng.com/public/uploads/large/91508177304fwt
qbi6ctvq3s7govin9kdhbopkgx6pm2tw9buwrhpiqjgygotyhs5dblx1
tu7hnlc4ybfyrbkoebudhrtkjjfco08gx1ebrpncy.png', '2021-
07-15 23:17:00.000000', '2021-07-13 23:17:03.000000',
5),
(10, 'milk ', 'milk ', '24', 'https://clipart-
best.com/img/milk/milk-clip-art-31.png', '2021-07-15
23:17:00.000000', '2021-07-15 23:17:00.000000', 6),
(11, 'Yogurt', 'Yogurt also spelled yoghurt, yogourt or
yoghourt, is a food produced by bacterial fermentation
of milk. The bacteria used to make yogurt are known as
yogurt cultures24', 'ز',
'https://e7.pngegg.com/grimaces/139/319/peg-clipart-
yogurt-yogurt.png', '2021-05-19 23:24:19.000000', '2021-
05-19 23:24:19.000000', 6);
```

```
-- --------------------------------------------------------

--

-- Table structure for table `product_category`
--

CREATE TABLE `product_category` (
  `id` bigint(20) NOT NULL,
  `category_name` varchar(255) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--
-- Dumping data for table `product_category`
--

INSERT INTO `product_category` (`id`, `category_name`)
VALUES
(1, 'Fruits'),
(3, 'Vegetables'),
(4, 'Grains'),
(5, 'Protein'),
(6, 'Dairy');

-- --------------------------------------------------------

--

-- Table structure for table `users`
--
```

```sql
CREATE TABLE `users` (
  `id` int(50) NOT NULL,
  `username` varchar(255) NOT NULL,
  `password` varchar(255) NOT NULL,
  `type` int(50) NOT NULL DEFAULT 0,
  `email` varchar(255) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--
-- Dumping data for table `users`
--

INSERT INTO `users` (`id`, `username`, `password`,
`type`, `email`) VALUES
(1, 'admin', 'admin', 1, 'admin@admin.com');

--
-- Indexes for dumped tables
--

--
-- Indexes for table `product`
--
ALTER TABLE `product`
  ADD PRIMARY KEY (`id`),
  ADD KEY `category-id` (`category_id`);

--
-- Indexes for table `product_category`
--
```

```
ALTER TABLE `product_category`
  ADD PRIMARY KEY (`id`);

--
-- Indexes for table `users`
--
ALTER TABLE `users`
  ADD PRIMARY KEY (`id`);

--
-- AUTO_INCREMENT for dumped tables
--

--
-- AUTO_INCREMENT for table `product`
--
ALTER TABLE `product`
  MODIFY `id` int(20) NOT NULL AUTO_INCREMENT,
AUTO_INCREMENT=15;

--
-- AUTO_INCREMENT for table `product_category`
--
ALTER TABLE `product_category`
  MODIFY `id` bigint(20) NOT NULL AUTO_INCREMENT,
AUTO_INCREMENT=7;

--
-- AUTO_INCREMENT for table `users`
--
ALTER TABLE `users`
```

```
  MODIFY `id` int(50) NOT NULL AUTO_INCREMENT,
AUTO_INCREMENT=2;

--
-- Constraints for dumped tables
--

--
-- Constraints for table `product`
--
ALTER TABLE `product`
  ADD CONSTRAINT `category-id` FOREIGN KEY
(`category_id`) REFERENCES `product_category` (`id`) ON
DELETE CASCADE ON UPDATE CASCADE;
COMMIT;

/*!40101 SET
CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET
CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET
COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
```

******THE END******

X==========X==========X