What is SpringShop?

SpringShop is an online store which offers the purchase of electronic devices. By accessing the website, the user has options of registering or logging into an existing account. After that, the user selects a product from the existing list and proceeds to finalize the order.

Applied technologies

- Java (version 11)

- Maven

- Spring Boot (version 2.6.6)

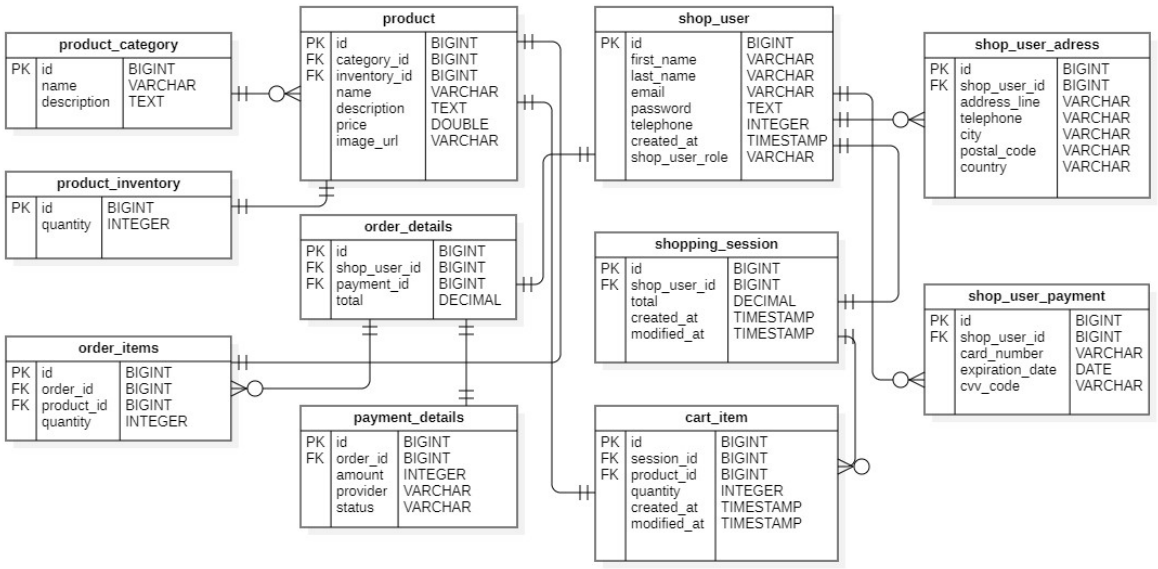
- Lombok – library for automation of creating classes

- Spring Security – a tool to ensure authentication, authorization, other security features

- PostgreSQL

- Spring Data JPA – a tool for using the database

Database UML diagram



Email notifications

JavaMail API was used to send messages to customers. It is a library that provides solutions for sending emails.

First add dependencies to the project:

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

Then in the applications.yaml file we need to add the configuration:

```
spring:
  mail:
    host: smtp.gmail.com
    port: 587
    username: spr.shop22@gmail.com
    password: qksgtegnzfqzlrnf
    properties:
      mail:
        smtp:
          auth: true
          starttls:
            enable: true
```

Implementation:

```
@Service
@AllArgsConstructor
public class EmailSenderService {
    private JavaMailSender mailSender;

    @Async
    public void sendEmail(String toEmail, String subject, String body) {
        SimpleMailMessage message = new SimpleMailMessage();
        message.setFrom("spr.shop22@gmail.com");
        message.setTo(toEmail);
        message.setText(body);
        message.setSubject(subject);
        mailSender.send(message);
    }
}
```

CRUD Endpoints

For each of the classes in the project, we created methods which are endpoints. They can be used to display objects of a class, add, or remove them.

1. **Product controller**

```java
@RestController
@RequestMapping(path = "api/v1/products")
@AllArgsConstructor
public class ProductController {

    private final ProductService productService;

    @GetMapping
    public List<Product> getAllProducts() { return this.productService.getAllProducts(); }

    @GetMapping(path = "{productId}")
    public Optional<Product> getProductById(@PathVariable("productId") Long productId) {
        return this.productService.getProductById(productId);
    }
}
```

- *getAllProducts* - returns a list of products from the database
- *getProductById* - returns a product that has the given identifier

2. **ShoppingSession controller**

```java
@RestController
@RequestMapping(path = "api/v1/shopping_session")
public class ShoppingSessionController {

    private final ShoppingSesionService shoppingSesionService;

    @Autowired
    public ShoppingSessionController(ShoppingSesionService shoppingSesionService) {
        this.shoppingSesionService = shoppingSesionService;
    }

    @GetMapping
    public List<ShoppingSession> getShoppingSessions() { return shoppingSesionService.getShoppingSessions(); }

    @DeleteMapping(path = "{shoppingSessionID}")
    public void deleteShoppingSession(@PathVariable("shoppingSessionID") int shoppingSessionID) {
        shoppingSesionService.deleteSession(shoppingSessionID);
    }
}
```

- *getShoppingSessions* - returns a list of shopping sessions
- *deleteShoppingSession* - removes a shopping session that has the identifier specified in the parameter

3. PaymentDetails controller

```java
@RestController
@RequestMapping(path = "api/v1/payment_details")
public class PaymentDetailsController {
    private final PaymentDetailsService paymentDetailsService;

    @Autowired
    public PaymentDetailsController(PaymentDetailsService paymentDetailsService) {
        this.paymentDetailsService = paymentDetailsService;
    }

    @GetMapping
    public List<PaymentDetails> getAllPayments() { return this.paymentDetailsService.getAllPayments(); }

    @GetMapping(path = "{paymentId}")
    public Optional<PaymentDetails> getPaymentById(@PathVariable("paymentId") Integer paymentId) {
        return this.paymentDetailsService.getPaymentById(paymentId);
    }

    @PostMapping
    public void addPayment(@RequestBody PaymentDetails payment) { this.paymentDetailsService.addPayment(payment); }

    @DeleteMapping(path = "{paymentId}")
    public void deletePaymentById(@PathVariable("paymentId") Integer paymentId) {
        this.paymentDetailsService.deletePaymentById(paymentId);
    }
}
```

- *getAllPayments* - returns list of payments
- *getPaymentById* - returns a payment that has the identifier specified in the parameter
- *addPayment* - adds a payment specified in the parameter to the database
- *deletePaymentById* - removes a payment that has the identifier specified in the parameter

4. OrderItems controller

```java
@RestController
@RequestMapping(path = "api/v1/order_items")
public class OrderItemsController {
    private final OrderItemsService orderItemsService;

    @Autowired
    public OrderItemsController(OrderItemsService orderItemsService) { this.orderItemsService = orderItemsService; }

    @GetMapping
    public List<OrderItems> getAllOrderItems() { return this.orderItemsService.getAllOrderItems(); }

    @GetMapping(path = "{orderItemsId}")
    public Optional<OrderItems> getOrderItemsById(@PathVariable("orderItemsId") Integer orderItemsId) {
        return this.orderItemsService.getOrderItemsById(orderItemsId);
    }

    @PostMapping
    public void addOrderItems(OrderItems orderItems) { this.orderItemsService.addOrderItems(orderItems); }

    @DeleteMapping(path = "{orderItemsId}")
    public void deleteOrderItemsById(@PathVariable("orderItemsId") Integer orderItemsId) {
        this.orderItemsService.deleteOrderItemsById(orderItemsId);
    }
}
```

- *getAllOrderItems* - returns a list of order items
- *getOrderItemsById* - returns order items that have the identifier specified in the parameter
- *addOrderItems* - adds order items specified in the parameter to the database

- *deleteOrderItemsById* - removes order items that have the identifier specified in the parameter

5. **CartItem controller**

```java
@RestController
@RequestMapping(path = "api/v1/cart_item")
public class CartItemController {

    private final CartItemService cartItemService;

    @Autowired
    public CartItemController(CartItemService cartItemService) { this.cartItemService = cartItemService; }

    @GetMapping
    public List<CartItem> getCartItems() { return cartItemService.getCartItems(); }

    @DeleteMapping(path = "{cartItemID}")
    public void deleteCartItem(@PathVariable("cartItemID") int cartItemID) {
        cartItemService.deleteCartItem(cartItemID);
    }
}
```

- *getCartItems* - returns a list of cart items
- *deleteCartItem* - removes a cart item that has the identifier specified in the parameter

6. **OrderDetails controller**

```java
@RestController
@RequestMapping(path = "api/v1/order")
public class OrderDetailsController {
    private final OrderDetailsService orderDetailsService;

    @Autowired
    public OrderDetailsController(OrderDetailsService orderDetailsService) {
        this.orderDetailsService = orderDetailsService;
    }

    @GetMapping
    public List<OrderDetails> getAllOrderDetails() { return this.orderDetailsService.getAllOrderDetails(); }

    @GetMapping(path = "{orderId}")
    public Optional<OrderDetails> getOrderDetailsById(@PathVariable("orderId") Integer orderId) {
        return this.orderDetailsService.getOrderDetailsById(orderId);
    }

   @PostMapping
   public void addOrderDetails(@RequestBody OrderDetails orderDetails) {
       this.orderDetailsService.addOrderDetails(orderDetails);
   }

   @DeleteMapping(path = "{orderId}")
   public void deleteOrderDetailsById(@PathVariable("orderId") Integer orderId) {
       this.orderDetailsService.deleteOrderDetailById(orderId);
   }
}
```

- *getAllOrderDetails* - returns a list of orders with details

- *getOrderDetailsById* - returns details of the order that has the identifier specified in the parameter
- *addOrderDetails* - adds order details specified in the parameter to the database
- *deleteOrderDetailsById* - removes order details that have the identifier specified in the parameter

7. ShopUser controller

```java
@RestController
@RequestMapping(path = "api/v1/shop-users")
@AllArgsConstructor
public class ShopUserController {
    private final ShopUserService shopUserService;
    private final ShopUserAddressService shopUserAddressService;
    private final ShopUserPaymentService shopUserPaymentService;

    @GetMapping
    public List<ShopUser> getUsers() { return shopUserService.getUsers(); }

    @GetMapping(path = "{shopUserId}")
    public ShopUser getUserById(@PathVariable("shopUserId") Long shopUserId) {
        return shopUserService.getUserById(shopUserId);
    }

    @GetMapping(path = "{shopUserId}/addresses")
    public Optional<List<ShopUserAddress>> getUserAddressesByShopUserId(@PathVariable("shopUserId") Long shopUserId) {
        return shopUserAddressService.getAddressesByUserId(shopUserId);
    }

    @GetMapping(path = "{shopUserId}/payments")
    public Optional<List<ShopUserPayment>> getShopUserPaymentsByShopUserId(@PathVariable("shopUserId") Long shopUserId) {
        return shopUserPaymentService.getShopUserPaymentsByShopUserId(shopUserId);
    }

    @DeleteMapping(path = "{shopUserId}")
    public void deleteUser(@PathVariable("shopUserId") Long shopUserId) { shopUserService.deleteUser(shopUserId); }
}
```

- *getUsers* - returns a list of shop users
- *getUserById* - returns a user that has the identifier specified in the parameter
- *getUserAdressesByShopUserId* - returns a list of adresses of the user that has the identifier specified in the parameter
- *getShopUserPamentsByShopUserId* - returns a list of payments of the user that has the identifier specified in the parameter
- *deleteUser* - removes the shop user that has the identifier specified in the parameter