과제 발표

홈쇼핑 결제까지의 과정 구현

1. 프로젝트 목적

💡 홈쇼핑에서 사용자가 물품들을 검색하고 장바구니에 넣은 다음
주문을 넣어서 결제하기까지의 일련의 과정들을 구현하는 것

2. 개발 내용

구현할 시나리오

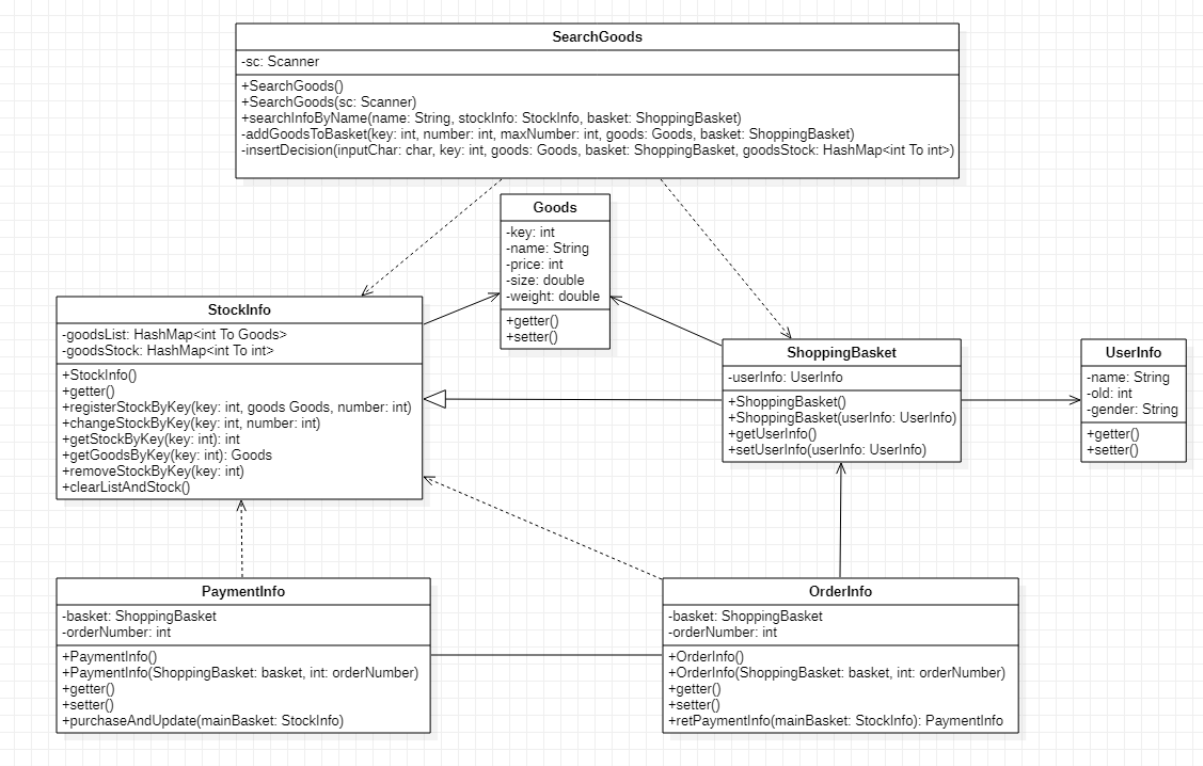
💡 1. 사용자(유저)가 사이트(서버)에 등록되어있는 물품들을 조회

2. 사용자가 서버에 등록되어있는 물품들 중 원하는 물품들을 장바구니에 등록

3. 사용자가 자신의 장바구니에서 자신의 물품들을 조회 및 삭제

4. 사용자가 장바구니 페이지에서 구매버튼을 눌러 물품 구입 및 서버의 물품 목록 갱신

2.1 클래스 다이어그램



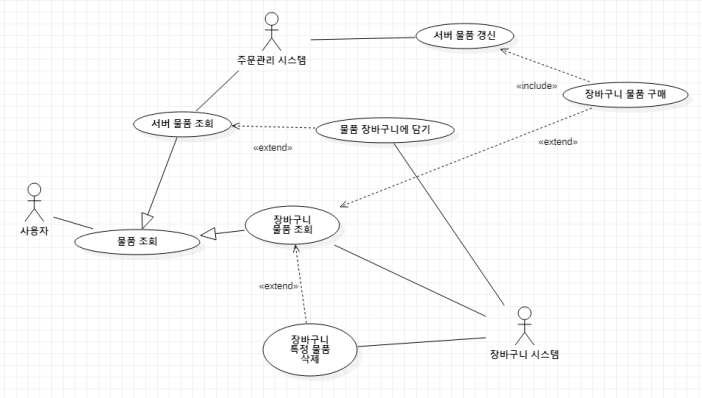
- 클래스 다이어그램

2.2 프로젝트 계획서 설명

가. 단계별 마일스톤

1) 해당 프로젝트의 시나리오와 유즈케이스 다이어그램을 작성

진행한 프로젝트는 사용자가 이미 회원등록과 로그인이 완료된 상태라고 가정하고 시나리오를 작성하였다. 이때 시나리오는 크게 4개의 시나리오로 나누어서 작성되었으며 해당 유즈케이스 다이어그램은 다음과 같이 나왔다.



- 유즈 케이스 다이어그램

2) 해당 프로젝트의 클래스 다이어그램을 작성

처음 클래스 다이어그램을 작성했을 때는 상품의 종류들만 구분되어있는 목록 정보 클래스와 상품의 수량만 구분되어있는 재고정보 클래스를 나누어서 다이어그램을 작성함. 상품을 조회하거나 장바구니에서 제어, 구매할 때 위 두개의 클래스를 모두 제어하는것으로 시스템을 설계하였지만 부적합하다고 판단하여 결국 앞에서 발표한 클래스로 구조를 바꾸어서 다이어그램을 작성함.

3) 다이어그램들을 토대로 코드 작성, 클래스 다이어그램 수정

시나리오 작성시 매우 제한적으로 범위를 줄여서 해당 코드를 작성하는데 다이어그램에서 예상하지 못한 클래스가 추가되거나 삭제되지는 않음. 다만 클래스간의 관계에서 약간의 변화가 있어서 클래스 다이어그램을 수정

나. 작업과 공정율

각 클래스별로 작업후 공정율을 적용

기능별 구분	클래스	공정율
재고	Goods	100%
	StockInfo	100%
장바구니	ShoppingBasket	100%
	UserInfo	50%
조회	SearchGoods	100%
주문	OrderInfo	100%
	PaymentInfo	70%

구현 발표

가. 구현 내용

- Goods

```
package com.main;
public class Goods {
    private String name;
    private int price;
    private double size;
    private double weight;
  private int key;

    public Goods() {}
    public Goods(int key, String name, int price, double size, double weight) {
        this.key = key;
        this.name = name;
        this.price = price;
        this.size = size;
        this.weight = weight;
    }

    public int getKey() {
        return key;
    }

    public void setKey(int key) {
        this.key = key;
    }

    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public int getPrice() {
        return price;
    }
```

```
    public void setPrice(int price) {
        this.price = price;
    }
    public double getSize() {
        return size;
    }
    public void setSize(double size) {
        this.size = size;
    }
    public double getWeight() {
        return weight;
    }
    public void setWeight(double weight) {
        this.weight = weight;
    }

    @Override
    public String toString() {
        return String.format("제품 키: %d 이름: %s, 가격: %d원, 크기: %f, 무게: %f", key, name, price, size, weight);
    }
}
```

- StockInfo

```
package com.main;
import java.util.HashMap;

public class StockInfo {
    private HashMap<Integer, Goods> goodsList;
    private HashMap<Integer, Integer> goodsStock;

    public StockInfo() {
        goodsList = new HashMap<>();
        goodsStock = new HashMap<>();
    }

    public HashMap<Integer, Goods> getGoodsList() {
        return goodsList;
    }

    public HashMap<Integer, Integer> getGoodsStock() {
        return goodsStock;
    }

    public void registerStockByKey(int key, Goods goods, int number) {
        goodsList.put(key, goods);
        goodsStock.put(key, number);

    }

    public void changeStockByKey(int key, int number) {
        if(goodsList.containsKey(key)) {
            goodsStock.put(key,number);
        }
    }

    public int getStockByKey(int key) {
        return goodsStock.get(key);
    }

    public Goods getGoodsByKey(int key) {
        return goodsList.get(key);
    }

    public void removeStockByKey(int key) {
        goodsList.remove(key);
        goodsStock.remove(key);
    }

    public void clearListAndStock() {
      goodsList.clear();
      goodsStock.clear();
    }

    @Override
    public String toString() {
        StringBuilder str = new StringBuilder();
        goodsList.forEach((key, goods)->{
            String s =  String.format("//상품 키: %d, 상품 이름: %s, 상품 갯수: %d, 상품 가격: %d// \n", key, goods.getName(), goodsStock.get
            str.append(s);
        });
```

```
        return str.toString();
    }
}
```

- ShoppingBasket

```
package com.main;
import java.util.HashMap;

public class ShoppingBasket extends StockInfo{
    private UserInfo userInfo;
    public ShoppingBasket() {
        super();
    }
    public ShoppingBasket(UserInfo userInfo) {
        super();
        this.userInfo = userInfo;
    }

    public UserInfo getUserInfo() {
        return userInfo;
    }

    public void setUserInfo(UserInfo userInfo) {
        this.userInfo = userInfo;
    }
    @Override
    public void changeStockByKey(int key, int number) {
        super.changeStockByKey(key, number);
        if(getGoodsByKey(key)==null) {
          System.out.println("바꾸자 하는 상품의 키가 존재하지 않습니다");
        } else {
          System.out.println(getGoodsByKey(key)+"\n위 상품의 수량이" + getStockByKey(key) + " 으로 변경");
        }
    }

    @Override
    public void removeStockByKey(int key) {
        System.out.println("아래와 같은 상품을 장바구니에서 제거\n" + getGoodsByKey(key));
        super.removeStockByKey(key);
    }

    @Override
    public String toString() {
        StringBuilder str = new StringBuilder();

        str.append("장바구니 사용자 정보\n"+userInfo.toString()+"\n\n");

        str.append("장바구니 상품 목록 출력\n");
        HashMap<Integer, Integer> goodsStock = getGoodsStock();

        getGoodsList().forEach((key, goods)->{
            String s =  String.format("//장바구니 상품 키: %d, 상품 이름: %s, 상품 갯수: %d, 상품 가격: %d// \n", key, goods.getName(), goodsSt
            str.append(s);
        });

        return str.toString();
    }

}
```

- UserInfo

```
package com.main;
public class UserInfo {
    private String name;
    private int old;
    private String gender;

    public UserInfo() {}
    public UserInfo(String name, int old, String gender) {
        this.name = name;
        this.old = old;
        this.gender = gender;
    }

    public String getName() {
```

```
        return name;
    }

    public int getOld() {
        return old;
    }

    public String getGender() {
        return gender;
    }

    public void setName(String name) {
        this.name = name;
    }

    public void setOld(int old) {
        this.old = old;
    }

    public void setGender(String gender) {
        this.gender = gender;
    }

    @Override
    public String toString() {
        return String.format("사용자의 이름: %s, 사용자의 나이: %d, 사용자의 성별: %s", name, old, gender);
    }
}
```

- SearchGoods

```
package com.main;
import java.util.HashMap;
import java.util.Scanner;

public class SearchGoods {
    Scanner sc;

    public SearchGoods() {}
    public SearchGoods(Scanner sc) {
        this.sc = sc;
    }


    private void addGoodsToBasket(int key, int number, int maxNumber, Goods goods, ShoppingBasket basket) {
        if(number <= 0) {
            System.out.println("수량은 1개 이상부터 입력 가능합니다. 장바구니 입력이 취소됩니다.");
        } else if(number > maxNumber) {
            System.out.println("입력하신 수량이 재고의 최대수량보다 많아 최대수량을 장바구니에 담겠습니다.");
            basket.registerStockByKey(key, goods, maxNumber);
        } else {
            System.out.println("입력하신 수량을 장바구니에 담겠습니다.");
            basket.registerStockByKey(key, goods, number);
        }
    }

    private void insertDecision(char inputChar, int key, Goods goods, ShoppingBasket basket, HashMap<Integer, Integer> goodsStock) {
      if(inputChar == 'y') {
            System.out.print("구입하고자 하는 상품의 수량을 입력하세요. 수량 : ");
            int number = Integer.parseInt(sc.nextLine());
            int maxNumber = goodsStock.get(key);

            addGoodsToBasket(key, number, maxNumber, goods, basket);
            System.out.println();
        }
    }

    public void searchInfoByName(String name, StockInfo stockInfo, ShoppingBasket basket) {

        System.out.println("상품명이 "+name+"인 상품을 검색한 결과");

        HashMap<Integer, Goods> goodsList = stockInfo.getGoodsList();
        HashMap<Integer, Integer> goodsStock = stockInfo.getGoodsStock();
        if(goodsList.isEmpty()) {
            System.out.println("검색결과 없음");
        }
        boolean searched = false;

        for(int key: goodsList.keySet()) {
          Goods goods = goodsList.get(key);
          if(name.equals(goods.getName())) {
              searched = true;
```

```
                System.out.println(String.format("//상품 키: %d, 상품 이름: %s, 상품 갯수: %d, 상품 가격: %d// \n", key, goods.getName(), goo
                System.out.print("해당 상품을 구입하겠습니까? (y/n) : ");
                String input = sc.nextLine();
                char inputChar = input.charAt(0);
                insertDecision(inputChar, key, goods, basket, goodsStock);
            }
        }
        if(!searched) {
          System.out.println("검색결과 없음");
        }

    }


}
```

- OrderInfo

```
package com.main;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;

public class OrderInfo {
    private ShoppingBasket basket;
    private int orderNumber;

    public OrderInfo() {}
    public OrderInfo(ShoppingBasket basket, int orderNumber) {
        this.basket = basket;
        this.orderNumber = orderNumber;
    }

    public ShoppingBasket getBasket() {
        return basket;
    }

    public int getOrderNumber() {
        return orderNumber;
    }

    public void setBasket(ShoppingBasket basket) {
        this.basket = basket;
    }

    public void setOrderNumber(int orderNumber) {
        this.orderNumber = orderNumber;
    }

    public PaymentInfo retPaymentInfo(StockInfo mainBasket) {

        HashMap<Integer, Goods> goodsList = basket.getGoodsList();
        HashMap<Integer, Integer> goodsStock = basket.getGoodsStock();

        HashMap<Integer, Goods> mainList = mainBasket.getGoodsList();
        HashMap<Integer, Integer> mainStock = mainBasket.getGoodsStock();

        int totalPrice = 0;
        List<Integer> removeds = new ArrayList<>();
        for(int key : goodsList.keySet()) {
          if(!mainList.containsKey(key) || !mainStock.containsKey(key)) {
            System.out.println("재고 목록에 구매하고자 하는 상품 "+goodsList.get(key).getName()+"가 존재하지 않습니다");
          } else if(mainStock.get(key) < goodsStock.get(key)) {
            System.out.println("구매하고자 하는 상품 " + goodsList.get(key).getName() +"의 수량이 재고의 수량보다 적어서 구매목록에서 제외합니다");
//            basket.removeStockByKey(key);;
            removeds.add(key);

          } else {
            System.out.println("상품 " + goodsList.get(key).getName() +" 을 수량 "+goodsStock.get(key)+"만큼 구매합니다." );

            totalPrice += goodsList.get(key).getPrice()*goodsStock.get(key);

          }
        }

        for(int removed : removeds) {
          basket.removeStockByKey(removed);
        }
        System.out.println("총 가격 : "+totalPrice);
        PaymentInfo paymentInfo = new PaymentInfo(basket, orderNumber);
        return paymentInfo;
    }
```

```
}
```

- PaymentInfo

```
package com.main;
import java.util.HashMap;

public class PaymentInfo {
    private ShoppingBasket basket;
    private int orderNumber;

    public PaymentInfo() {}
    public PaymentInfo(ShoppingBasket basket, int orderNumber) {
        this.basket = basket;
        this.orderNumber = orderNumber;
    }

    public ShoppingBasket getBasket() {
    return basket;
  }
  public void setBasket(ShoppingBasket basket) {
    this.basket = basket;
  }
  public int getOrderNumber() {
    return orderNumber;
  }
  public void setOrderNumber(int orderNumber) {
    this.orderNumber = orderNumber;
  }
  public void purchaseAndUpdate(StockInfo mainBasket) {
      HashMap<Integer, Goods> goodsList = basket.getGoodsList();
        HashMap<Integer, Integer> goodsStock = basket.getGoodsStock();
        HashMap<Integer, Integer> mainStock = mainBasket.getGoodsStock();
        int totalPrice = 0;
        for(int key : goodsList.keySet()) {
          int updateStock = mainStock.get(key) - goodsStock.get(key);
          mainBasket.changeStockByKey(key, updateStock);
          totalPrice += goodsList.get(key).getPrice()*goodsStock.get(key);
        }
        System.out.println("총 가격 : "+totalPrice);
        System.out.println("최종 결제가 완료되었습니다.");
    }
}
```

나. 발견된 난제

1. 사용자가 상품을 조회할 때 상품의 이름으로 조회를 하는데 재고관리 클래스에서는 상품을 정수인 key값으로 관리하고 있어서 상품이름으로 바로 상품을 찾을 수 없었고 이를 해결해야했다.
2. 사용자가 올바르게 장바구니를 담은 상태에서도 별도의 과정 없이 바로 결제를 진행하고 재고를 조작하면 장바구니에 담는 사이에 물건이 나가거나 판매자쪽에서 구매를 막는 등의 여러 문제에 대처할 수 없었다.

다. 해결 과정 및 개선내용

1. a) 상품이름으로 정렬이 된 상품테이블이 별도로 있었으면 좋았겠지만 데이터베이스가 없는 관계상 상품 Map을 for문으로 돌면서 찾고자 하는 상품명이 있는지 확인하는 방법으로 상품을 찾았다
2. b) 사용자가 물품을 구매하는 과정을 주문 과정과 결제 과정으로 클래스를 별도로 나누어서 진행하고 주문을 요청받을 시 결제를 결정하는 식으로 문제를 해결할 수 있었다. 물론 시연단계에서는 주문을 받을 시 바로 결제를 결정했기 때문에 이 단계가 보이지는 않는다.

시연

homeshopping.zip

질의 및 응답