C++프로그래밍및실습

편의점 매대 관리 시스템

중간보고서

제출일자: 2024.12.01.
제출자명: 김태균
제출자학번: 213961

1. 프로그램 실행방법

A. storage.csv파일과 main.exe파일을 같은 폴더에 두고 exe폴더를 cmd로 실행

2. 기능별 구현상황

A. 재고 관리 기능

i. 데이터베이스 읽기

ii. 재고 추가

iii. 상품 추가

iv. 상품 제거

v. 관리 명령어 입력창

vi. 저장 및 종료 기능

vii. 데이터베이스 출력

B. 매대 상품 수량 변경 기능

i. 상품 판매

ii. 매대 진열

C. 진열 필요 상품 제시 기능

i. 매대 상태 및 피크 타임 여부 판단, 진열 필요 상품 제시

3. 설계 및 구현

A. 재고 관리 기능 – 데이터베이스 읽기

i. 코드 스크린샷

storage.h

```
struct item
{
    string code;
    string name;
    string stand;
    int cur_stand_qnt;
    int max_stand_qnt;
    int inventory;
    bool was_found;
} typedef Item;
```

```
// 데이터 저장공간 클래스
class Storage{
private:
    fstream fs;
    string storage_name;
    vector<Item> database; // 벡터를 활용한 동적 배열
public:
    Storage(string filename);
    Storage(const Storage& other); // database 멤버 변수를 깊은 복사하기 위한 복사생성자
    ~Storage(); // 소멸시 변경된 내용을 저장하는 소멸자

    Item GetItem(string code) const; // code에 해당하는 상품을 vector로 반환
    Item GetItem(int line) const; // line번째 상품을 벡터로 반환
    int GetSize() const; // database의 크기 반환, 목록 부분을 제외한 사이즈이다

    void AddLine(Item new_line); // database 행 추가
    void RemoveItem(string code); // code에 해당하는 제품 삭제
    void ItemSold(string code, int qnt); // code에 해당하는 제품 판매
    void ItemToStand(string code, int qnt); // code에 해당하는 제품 qnt만큼 매대에 진열
    void ItemStore(string code, int qnt); // code에 해당하는 제품 qnt만큼 재고수 추가
    void PrintDatabase(); // database 출력
};
```

storage.cpp

```cpp
Storage::Storage(string filename)
{
    string str_buf;
    this->storage_name = filename;
    this->fs.open(filename, ios::in);

    getline(this->fs, str_buf,'\n'); // 목록 부분을 제외하기 위해 한 줄을 받기만 하는 작업
    while(!fs.eof())
    {
        vector<string> tmp; // 파일을 읽기 위한 임시 벡터, 한 줄을 저장
        getline(this->fs, str_buf,'\n');

        // string 형태로 읽어진 csv파일 한 줄을 읽어 tmp에 저장한다
        int position;
        int cur_position = 0;
        while((position = str_buf.find(",", cur_position)) != string::npos)
        {
            int len = position - cur_position;
            tmp.push_back(str_buf.substr(cur_position, len));
            cur_position = position + 1;
        }
        tmp.push_back(str_buf.substr(cur_position));

        // tmp의 string 값을 자료형에 맞게 Item 구조체에 저장
        Item data_extracted;
```

```cpp
            data_extracted.code = tmp[0];
            data_extracted.name = tmp[1];
            data_extracted.stand = tmp[2];
            data_extracted.cur_stand_qnt = stoi(tmp[3]);
            data_extracted.max_stand_qnt = stoi(tmp[4]);
            data_extracted.inventory = stoi(tmp[5]);
            data_extracted.was_found = true;

            // 추출한 Item을 database에 저장한다
            AddLine(data_extracted);
        }
    }
```

```cpp
Storage::Storage(const Storage &other)
{
    this->fs.open(this->storage_name);
    this->storage_name = other.storage_name;
    // 복사할 Storage 객체의 database에 기존의 database 복사
    for(Item i : other.database)
    {
        AddLine(i);
    }
}
```

```cpp
Item Storage::GetItem(string code) const
{
    // code로 해당 item 검색
    for(Item i : this->database)
    {
        if(i.code == code)
        {
            return i; // database에 해당 item이 있을 경우 반환
        }
    }
    cout << "Wrong item code" << endl; // 검색되지 않았을 경우 에러 메시지
    // 검색되지 않을 경우에 반환하는 was_found 값이 false인 NULL 구조체
    Item null_item;
    null_item.was_found = false;
    return null_item;
}

Item Storage::GetItem(int line) const
{
    return this->database[line]; // line번째 Item 반환
}

int Storage::GetSize() const
{
    return this->database.size(); // database의 목록부분을 제외한 사이즈
}
```

main.cpp

```cpp
7   int main(int argc, char** argv)
8   {
9       Storage storage("storage.csv");
```

ii. 입력

- string filename: csv파일의 파일명

iii. 결과

- main함수에서 storage 객체 생성

- csv파일을 읽고 Item 구조체 동적 배열에 저장

iv. 설명

- storage 객체 정의

- Item 구조체 정의

- storage 객체의 정보를 반환하는 get메서드 정의

- GetItem을 code로 검색했을 때 검색에 실패했을 경우 Item 구조체의 was_found값이 false인 Item 객체를 반환한다. 이를 통해 검색에 실패했음을 알 수 있다

- GetItem의 경우 상품의 code 또는 목록에서의 인덱스 값으로 검색할 수 있도록 오버로드 함

- storage 생성자를 정의해 파일명에 해당하는 csv파일을 읽고 멤버변수인 database에 값들을 저장

- 복사 생성자를 정의해 멤버변수 database를 복사할 때 깊은 복사를 할 수 있도록 설계함

v. 수업 연관 내용

- 7주차 function overload

- 10주차 vector

- 11주차 copy constructor

B. 재고 관리 기능 – 재고 추가

i. 코드 스크린샷

storage.cpp

```
void Storage::ItemStore(string code, int qnt)
{
    // code로 해당 item 검색
    for(int i = 0; i < this->database.size(); i++)
    {
        if(this->database[i].code == code)
        {
            this->database[i].inventory += qnt; // 재고 수 증가
        }
    }
}
```

ii. 입력

- string code: 재고 추가할 상품의 code

- int qnt: 추가할 재고량

iii. 반환값

- 없음

iv. 결과

- 상품의 재고량을 입력된 개수만큼 추가

v. 설명

- 입력된 code의 상품을 검색한 후 입력된 qnt만큼 재고를 추가해 저장

vi. 수업 연관 내용

- 10주차 vector

C. 재고 관리 기능 – 상품 추가

i. 코드 스크린샷

storage.cpp

```
void Storage::AddLine(Item new_line)
{
    this->database.push_back(new_line); // database에 item 추가
}
```

main.cpp

```
case 2: // New item
{
    // 상품 정보 입력
    string item_code;
    string item_name;
    string stand;
    string cur_stand_qnt;
    string max_stand_qnt;
    string inventory;

    cout << "Item Code: ";
    cin >> item_code;
    cout << "Item Name: ";
    cin >> item_name;
    cout << "Stand that Item belongs: ";
    cin >> stand;
    cout << "Current Qunatity in Stand: ";
    cin >> cur_stand_qnt;
    cout << "Max Qunatity of Stand: ";
    cin >> max_stand_qnt;
    cout << "Qunatity in Storage: ";
    cin >> inventory;
```

```
    // 새로운 상품 storage에 추가
    Item newItem;
    newItem.code = item_code;
    newItem.name = item_name;
    newItem.stand = stand;
    newItem.cur_stand_qnt = stoi(cur_stand_qnt);
    newItem.max_stand_qnt = stoi(max_stand_qnt);
    newItem.inventory = stoi(inventory);

    storage.AddLine(newItem);
    storage.PrintDatabase();
    break;
}
```

ii. 입력

- Item new_line: 새로 추가할 상품의 정보가 담긴 Item 구조체

iii. 반환값

- 없음

iv. 결과

- 상품을 database에 추가한다

v. 설명

- 입력값인 new_line은 database의 목록 규격에 맞춰서 미리 생성되어 있어야 한다

- 메뉴 선택 switch문 case 2이다

vi. 수업 연관 내용

- 10주차 vector

D. 재고 관리 기능 – 상품 제거

i. 코드 스크린샷

storage.cpp

```
void Storage::RemoveItem(string code)
{
    // code로 해당 item 검색
    for(int i = 0; i < this->database.size(); i++)
    {
        // 검색되었을 경우 해당 item 삭제
        if(this->database[i].code == code)
        {
            this->database.erase(this->database.begin() + i);
            return;
        }
    }
    // 검색되지 않았을 경우 에러 메시지
    cout << "There is no item that has code " << code << endl;
}
```

main.cpp

```
case 5: // Remove item
{
    string item_code;
    cout << "Item Code: ";
    cin >> item_code;

    // 상품 삭제 처리
    storage.RemoveItem(item_code);
    storage.PrintDatabase();
    break;
}
```

ii. 입력

- string code: 상품의 code값

iii. 반환값

- 없음

iv. 결과

- code값에 해당하는 상품을 삭제한다

v. 설명

- code값으로 검색하여 해당 코드의 상품을 제거

- 메뉴 선택 switch문 case 5이다

vi. 수업 연관 내용

- 10주차 vector

E. 재고 관리 기능 – 관리 명령어 입력창

i. 코드 스크린샷

main.cpp

```
    bool flag = true;
    int menu;
    while(flag)
    {
// 메뉴 선택
cout << "1. Sell item  2. New item  3. Place item on stand  4. Add item in storage " << endl;
cout << "5. Remove item 6. Save and Exit" << endl;

cin >> menu;
switch(menu){
```

ii. 입력

- bool flag: 반복문 탈출 flag

- int menu: 명령어 입력 값

iii. 결과

- 입력 받은 메뉴에 따라 상품 판매, 상품 추가, 상품 진열, 재고 추가, 상품 삭제, 저장 및 종료 명령을 실행한다

iv. 설명

- switch문을 통해 선택한 메뉴를 실행한다

- 저장 및 종료 명령의 경우 flag값을 바꿔 반복문을 탈출, 프로그램이 종료된다. 이때 소멸자에 의해 storage 객체가 소멸하며 변경된 내용을 저장한다

v. 수업 연관 내용

- 4주차 condition

F. 재고 관리 기능 – 저장 및 종료

i. 코드 스크린샷

storage.cpp

```
// 소멸하면서 데이터베이스의 내용을 같은 이름의 새로운 csv파일로 만들어 저장한다
Storage::~Storage()
{
    ofstream outfile(this->storage_name);
    for(int i = 0; i < this->database.size(); i++)
    {
        outfile << this->database[i].code << ",";
        outfile << this->database[i].name << ",";
        outfile << this->database[i].stand << ",";
        outfile << this->database[i].cur_stand_qnt << ",";
        outfile << this->database[i].max_stand_qnt << ",";
        outfile << this->database[i].inventory << endl;
    }
}
```

main.cpp

```
case 6:
{
    // flag를 false로 바꿔 무한 반복문 탈출
    flag = false;
    break;
}
```

ii. 입력

- storage_name: 파일명

- database: 상품 정보 저장된 동적 배열

iii. 결과

- 변경된 내용으로 동일한 파일명의 파일을 만들어 저장한다

iv. 설명

- 소멸자를 정의하여 main함수가 종료되어 프로그램이 종료되는 시점에 객체가 소멸하며 변경된 내용을 자동으로 저장하게 된다

- 메뉴 선택 switch문 case 6이다

v. 수업 연관 내용

- 10주차 Deconstructor

G. 재고 관리 기능 – 데이터베이스 출력

i. 코드 스크린샷

storage.cpp

```
void Storage::PrintDatabase()
{
    for(item i : this->database)
    {
        cout.width(13);
        cout << right << i.code << " ";
        cout.width(25);
        cout << right << i.name << " ";
        cout. width(13);
        cout << right << i.stand << " " << i.cur_stand_qnt << " "
             << i.max_stand_qnt << " " << i.inventory << endl;
    }
}
```

ii. 입력

- database: 상품정보 저장된 동적 배열

iii. 반환값

- 없음

iv. 결과

- 데이터베이스 출력

v. 설명

- 형식에 맞춰 출력할 수 있도록 cout.width()로 간격 설정 및 std::right로 간격에 맞춰 출력

H. 매대 상품 수량 변경 기능 – 상품 판매

i. 코드 스크린샷

storage.cpp

```
void Storage::ItemSold(string code, int qnt)
{
    // code로 해당 item 검색
    for(int i = 0; i < this->database.size(); i++)
    {
        // item이 검색 되었을 경우
        if(this->database[i].code == code)
        {
            int qnt_result = database[i].cur_stand_qnt - qnt; // 판매 후 진열된 수량
            // 진열된 수량이 판매 수량보다 적은 경우 에러 메시지
            if(qnt_result < 0)
            {
                cout << "Not enough items on the stand. Check quantity of the sold item" << endl;
                return;
            }
            // 아닐 경우 판매 수량만큼 진열된 수량 감소
            else
            {
                this->database[i].cur_stand_qnt  = qnt_result;
            }
        }
    }
}
```

main.cpp

```
case 1: // Sell item
{
    // 판매할 상품 code와 수량 입력
    string sell_item_code;
    int sell_qnt;
    cout << "Item Code: ";
    cin >> sell_item_code;
    cout << "Quantity: ";
    cin >> sell_qnt;

    // 해당 상품 code로 검색해서 없을 경우 에러 메시지 출력 후 break
    Item item = storage.GetItem(sell_item_code); // 검색되지 않았을 경우 오류 메시지 출력
    // 상품이 검색되지 않았을 경우 반환받은 객체 검사
    if(item.was_found == false)
    {
        break;
    }

    // 상품 판매 처리
    storage.ItemSold(sell_item_code, sell_qnt);
    storage.PrintDatabase();
    break;
}
```

ii. 입력

- string code: 판매할 상품의 code

- int qnt: 판매할 수량

iii. 반환값

- 없음

iv. 결과

- 판매한 수량만큼 매대에서 수량감소

- 매대에 수량이 부족할 경우 잘못된 입력이라고 판단

v. 설명

- 상품의 code로 database를 검색하여 해당 상품의 매대 진열 수량을 변경한다

- 메뉴 선택 switch문 case 1이다

I. 매대 상품 수량 변경 기능 – 매대 진열

i. 코드 스크린샷

storage.cpp

```
void Storage::ItemToStand(string code, int qnt)
{
    // code로 해당 item 검색
    for(int i = 0; i < this->database.size(); i++)
    {
        if(this->database[i].code == code)
        {
            // 재고가 부족할 경우 에러 메시지
            if(qnt > database[i].inventory)
            {
                cout << "Not enough items in the inventory. Check quantity of the item" << endl;
                return;
            }

            int qnt_result_cur = database[i].cur_stand_qnt + qnt; // 매대에 진열될 총 개수
            // 진열할 수 있는 최대치를 넘어서는 경우 에러 메시지
            if(qnt_result_cur > database[i].max_stand_qnt)
            {
                cout << "Not enough space for the item. Check quantity of the item" << endl;
                return;
            }
            this->database[i].cur_stand_qnt = qnt_result_cur; // 매대 진열 개수 변경
            this->database[i].inventory -= qnt; // 재고 수 변경
        }
    }
}
```

main.cpp

```
case 3: // Place item on stand
{
    // 상품 코드, 수량 입력
    string item_code;
    string qnt;

    cout << "Item Code: ";
    cin >> item_code;
    cout << "Quantity: ";
    cin >> qnt;

    // 상품 진열 처리
    storage.ItemToStand(item_code, stoi(qnt));
    storage.PrintDatabase();
    break;
}
```

ii. 입력

- string code: 판매할 상품의 code

- int qnt: 매대에 진열할 수량

iii. 반환값

- 없음

iv. 결과

- 재고 상품의 수량이 감소하고 매대에 진열된 수량이 증가한다

v. 설명

- 재고 수량이 부족하거나 매대에 공간이 부족한 상황에서 너무 많은 수량을 진열하려 한 경우 에러 메시지 출력

- 메뉴 선택 switch문 case3이다

J. 매대 상태 및 피크 타임 여부 판단, 진열 필요 상품 제시

i. 코드 스크린샷

main.cpp

```
cout << "Item to display on the stand" << endl;
if(isPeak == true)
    cout << "Peaktime = true" << endl;
else
    cout << "Peaktime = false" << endl;

for(int i = 0; i < storage.GetSize(); i++)
{
    // line번째 아이템들의 값을 저장
    // 추후 각 아이템의 정보를 vector<string>으로 저장하는 것이 아닌 구조체를 만들어 저장하는 것으로 class 정의 업데이트 예정
    Item item = storage.GetItem(i);
    int cur_stand_qnt =  item.cur_stand_qnt;
    int max_stand_qnt = item.max_stand_qnt;
    int storage_qnt = item.inventory;

    if(isPeak == true) // 피크타임일 경우 매대에 넣을 수 있는 공간이 있기만 하면 해당 상품 및 추가 진열 수량 제시
    {
        if(cur_stand_qnt < max_stand_qnt && storage_qnt >= (max_stand_qnt - cur_stand_qnt))
        {
            cout.width(13);
            cout << std::right << item.code;
            cout.width(25);
            cout << std::right << item.name << " ---> " << max_stand_qnt - cur_stand_qnt << endl;
        }
    } else // 피크타임이 아닐 경우
    {
        // 매대 최대 수량의 40%보다 적은 수량이 매대에 있을 경우 우선도가 높은 진열 품목으로 제시
        if(cur_stand_qnt < (int)(max_stand_qnt * 0.4) && storage_qnt >= ((int)(max_stand_qnt * 0.4) - cur_stand_qnt))
        {
            cout.width(13);
            cout << std::right << item.code;
            cout.width(25);
            cout << std::right << item.name << " ---> " << max_stand_qnt - cur_stand_qnt << "  Urgent" << endl;
        }
```

```
        // 매대 최대 수량의 40%이상 60% 미만의 수량이 매대에 있을 경우 우선도가 낮은 진열 품목으로 제시
        else if((cur_stand_qnt >= (int)(max_stand_qnt * 0.4) && cur_stand_qnt < (int)(max_stand_qnt * 0.6))
         && storage_qnt >= ((int)(max_stand_qnt * 0.6) - cur_stand_qnt))
        {
            cout.width(13);
            cout << std::right << item.code;
            cout.width(25);
            cout << std::right << item.name << " ---> " << max_stand_qnt - cur_stand_qnt << "  Not Urgent" << endl;
        }
    }
}
```

ii. 입력

- bool isPeak: 피크타임 여부

- Item item: 상품 정보

- int cur_stand_qnt, max_stand_qnt, storage_qnt: 상품 정보로부터 추출한 현재 매대 진열 수량, 최대 진열 가능 수량, 재고 수

iii. 결과

- 매대 진열된 수량과 최대 진열 가능 주량,

iv. 설명

- 피크타임인 경우: 모든 상품을 최대한 빠르게 진열해야 하기 때문에 우선순위를 제시하지 않고 최대 진열 가능 수량보다 매대 진열 수량이 적을 경우 진열해야 하는 상품과 수량을 제시한다

- 피크타임이 아닌 경우: 매대 최대 수량의 40%보다 적게 진열된 경우 우선순위가 높은 상품, 60%보다 작고 40%이상 진열된 경우 우선순위가 낮은 상품, 두 카테고리로 나누어 진열해야 하는 상품과 수량을 제시한다

- 재고 수 보다 많은 수량을 진열해야 하는 경우는 제시하지 않는다

4. 테스트

A. database 출력

```
      code                  name         stand
   1000000          ShirimpSnack       Snack 1 5 10
   1000002     PotatochipOriginal       Snack 3 5 30
   1000003        PotatochipOnion       Snack 5 5 7
   1000004         CornSnackSpicy       Snack 1 5 0
   1000005         CornSnackSweet       Snack 3 5 0
   3000000       FishcakebarSpicy  Refrigerated 5 12 0
   3000001         FishcakebarBBQ  Refrigerated 4 12 0
   3000002           ChickenChest  Refrigerated 6 12 0
   4000000     SweetSpicyCupnoodle    Cupnoodle 2 3 12
   4000001  SpicyChickenCupnoodle    Cupnoodle 3 3 8
   5000000                 Sprite       Walkin 9 15 6
   5000001                   Pepci      Walkin 12 15 0
   5000002          PepciZeroLime       Walkin 7 15 0
```

B. 상품 추가

```
1. Sell item  2. New item  3. Place item on stand  4. Add item in storage
5. Remove item 6. Save and Exit
2
Item Code: 5000003
Item Name: PepciZeroMango
Stand that Item belongs: Walkin
Current Qunatity in Stand: 14
Max Qunatity of Stand: 15
Qunatity in Storage: 20
      code                  name         stand
   1000000          ShirimpSnack       Snack 1 5 10
   1000002     PotatochipOriginal       Snack 3 5 30
   1000003        PotatochipOnion       Snack 5 5 7
   1000004         CornSnackSpicy       Snack 1 5 0
   1000005         CornSnackSweet       Snack 3 5 0
   3000000       FishcakebarSpicy  Refrigerated 5 12 0
   3000001         FishcakebarBBQ  Refrigerated 4 12 0
   3000002           ChickenChest  Refrigerated 6 12 0
   4000000     SweetSpicyCupnoodle    Cupnoodle 2 3 12
   4000001  SpicyChickenCupnoodle    Cupnoodle 3 3 8
   5000000                 Sprite       Walkin 9 15 6
   5000001                   Pepci      Walkin 12 15 0
   5000002          PepciZeroLime       Walkin 7 15 0
   5000003          PepciZeroMango      Walkin 14 15 20
Item to display on the stand
Peaktime = true
   1000000          ShirimpSnack ---> 4
   1000002     PotatochipOriginal ---> 2
   4000000     SweetSpicyCupnoodle ---> 1
   5000000                 Sprite ---> 6
   5000003          PepciZeroMango ---> 1
```

C. 상품 판매

D. 진열 필요 상품 제시(피크타임 아닌 경우)

```
         code                    name         stand
      1000000             ShirimpSnack         Snack 1 5 10
      1000002       PotatochipOriginal         Snack 3 5 30
      1000003          PotatochipOnion         Snack 5 5 7
      1000004           CornSnackSpicy         Snack 1 5 0
      1000005           CornSnackSweet         Snack 1 5 0
      3000000         FishcakebarSpicy  Refrigerated 5 12 0
      3000001          FishcakebarBBQ  Refrigerated 4 12 0
      3000002            ChickenChest  Refrigerated 6 12 0
      4000000      SweetSpicyCupnoodle     Cupnoodle 2 3 12
      4000001    SpicyChickenCupnoodle     Cupnoodle 3 3 8
      5000000                   Sprite        Walkin 9 15 6
      5000001                    Pepci        Walkin 12 15 0
      5000002           PepciZeroLime        Walkin 7 15 0
      5000003          PepciZeroMango        Walkin 14 15 20
 Item to display on the stand
 Peaktime = false
      1000000            ShirimpSnack ---> 4  Urgent
 1. Sell item  2. New item  3. Place item on stand  4. Add item in storage
 5. Remove item 6. Save and Exit
```

```
1
Item Code: 4000000
Quantity: 2
         code                    name         stand
      1000000             ShirimpSnack         Snack 1 5 10
      1000002       PotatochipOriginal         Snack 3 5 30
      1000003          PotatochipOnion         Snack 5 5 7
      1000004           CornSnackSpicy         Snack 1 5 0
      1000005           CornSnackSweet         Snack 1 5 0
      3000000         FishcakebarSpicy  Refrigerated 5 12 0
      3000001          FishcakebarBBQ  Refrigerated 4 12 0
      3000002            ChickenChest  Refrigerated 6 12 0
      4000000      SweetSpicyCupnoodle     Cupnoodle 0 3 12
      4000001    SpicyChickenCupnoodle     Cupnoodle 3 3 8
      5000000                   Sprite        Walkin 9 15 6
      5000001                    Pepci        Walkin 12 15 0
      5000002           PepciZeroLime        Walkin 7 15 0
      5000003          PepciZeroMango        Walkin 14 15 20
Item to display on the stand
Peaktime = false
      1000000            ShirimpSnack ---> 4  Urgent
      4000000      SweetSpicyCupnoodle ---> 3  Urgent
```

E. 진열 필요 상품 제시(피크 타임인 경우)

```
         code                    name         stand
      1000000             ShirimpSnack         Snack 1 5 10
      1000002       PotatochipOriginal         Snack 3 5 30
      1000003          PotatochipOnion         Snack 5 5 7
      1000004           CornSnackSpicy         Snack 1 5 0
      1000005           CornSnackSweet         Snack 3 5 0
      3000000         FishcakebarSpicy  Refrigerated 5 12 0
      3000001          FishcakebarBBQ  Refrigerated 4 12 0
      3000002            ChickenChest  Refrigerated 6 12 0
      4000000      SweetSpicyCupnoodle     Cupnoodle 2 3 12
      4000001    SpicyChickenCupnoodle     Cupnoodle 3 3 8
      5000000                   Sprite        Walkin 9 15 6
      5000001                    Pepci        Walkin 12 15 0
      5000002           PepciZeroLime        Walkin 7 15 0
Item to display on the stand
Peaktime = true
      1000000            ShirimpSnack ---> 4
      1000002      PotatochipOriginal ---> 2
      4000000      SweetSpicyCupnoodle ---> 1
      5000000                  Sprite ---> 6
1. Sell item  2. New item  3. Place item on stand  4. Add item in storage
5. Remove item 6. Save and Exit
```

F. 상품 진열

```
        code                    name         stand
     1000000             ShirimpSnack         Snack 1 5 10
     1000002       PotatochipOriginal         Snack 3 5 30
     1000003          PotatochipOnion         Snack 5 5 7
     1000004            CornSnackSpicy        Snack 1 5 0
     1000005            CornSnackSweet        Snack 3 5 0
     3000000         FishcakebarSpicy  Refrigerated 5 12 0
     3000001           FishcakebarBBQ  Refrigerated 4 12 0
     3000002             ChickenChest  Refrigerated 6 12 0
     4000000      SweetSpicyCupnoodle     Cupnoodle 2 3 12
     4000001    SpicyChickenCupnoodle     Cupnoodle 3 3 8
     5000000                   Sprite        Walkin 9 15 6
     5000001                    Pepci        Walkin 12 15 0
     5000002            PepciZeroLime        Walkin 7 15 0
Item to display on the stand
Peaktime = true
     1000000             ShirimpSnack ---> 4
     1000002       PotatochipOriginal ---> 2
     4000000      SweetSpicyCupnoodle ---> 1
     5000000                   Sprite ---> 6
1. Sell item  2. New item  3. Place item on stand  4. Add item in storage
5. Remove item 6. Save and Exit
3
Item Code: 5000000
Quantity: 6
```

```
        code                    name         stand
     1000000             ShirimpSnack         Snack 1 5 10
     1000002       PotatochipOriginal         Snack 3 5 30
     1000003          PotatochipOnion         Snack 5 5 7
     1000004            CornSnackSpicy        Snack 1 5 0
     1000005            CornSnackSweet        Snack 3 5 0
     3000000         FishcakebarSpicy  Refrigerated 5 12 0
     3000001           FishcakebarBBQ  Refrigerated 4 12 0
     3000002             ChickenChest  Refrigerated 6 12 0
     4000000      SweetSpicyCupnoodle     Cupnoodle 2 3 12
     4000001    SpicyChickenCupnoodle     Cupnoodle 3 3 8
     5000000                   Sprite        Walkin 15 15 0
     5000001                    Pepci        Walkin 12 15 0
     5000002            PepciZeroLime        Walkin 7 15 0
Item to display on the stand
Peaktime = true
     1000000             ShirimpSnack ---> 4
     1000002       PotatochipOriginal ---> 2
     4000000      SweetSpicyCupnoodle ---> 1
```

G. 재고 추가

```
        code                    name         stand
     1000000             ShirimpSnack         Snack 1 5 10
     1000002       PotatochipOriginal         Snack 3 5 30
     1000003          PotatochipOnion         Snack 5 5 7
     1000004            CornSnackSpicy        Snack 1 5 0
     1000005            CornSnackSweet        Snack 3 5 0
     3000000         FishcakebarSpicy  Refrigerated 5 12 0
     3000001           FishcakebarBBQ  Refrigerated 4 12 0
     3000002             ChickenChest  Refrigerated 6 12 0
     4000000      SweetSpicyCupnoodle     Cupnoodle 2 3 12
     4000001    SpicyChickenCupnoodle     Cupnoodle 3 3 8
     5000000                   Sprite        Walkin 15 15 0
     5000001                    Pepci        Walkin 12 15 0
     5000002            PepciZeroLime        Walkin 7 15 0
Item to display on the stand
Peaktime = true
     1000000             ShirimpSnack ---> 4
     1000002       PotatochipOriginal ---> 2
     4000000      SweetSpicyCupnoodle ---> 1
1. Sell item  2. New item  3. Place item on stand  4. Add item in storage
5. Remove item 6. Save and Exit
4
Item Code: 5000000
Quantity: 20
```

```
      code                    name         stand
   1000000           ShirimpSnack       Snack 1 5 10
   1000002      PotatochipOriginal       Snack 3 5 30
   1000003         PotatochipOnion       Snack 5 5 7
   1000004          CornSnackSpicy       Snack 1 5 0
   1000005          CornSnackSweet       Snack 3 5 0
   3000000       FishcakebarSpicy  Refrigerated 5 12 0
   3000001         FishcakebarBBQ  Refrigerated 4 12 0
   3000002           ChickenChest  Refrigerated 6 12 0
   4000000     SweetSpicyCupnoodle     Cupnoodle 2 3 12
   4000001   SpicyChickenCupnoodle     Cupnoodle 3 3 8
   5000000                 Sprite        Walkin 15 15 20
   5000001                   Pepci       Walkin 12 15 0
   5000002          PepciZeroLime       Walkin 7 15 0
```

H. 제품 삭제

```
      code                    name         stand
   1000000           ShirimpSnack       Snack 1 5 10
   1000002      PotatochipOriginal       Snack 3 5 30
   1000003         PotatochipOnion       Snack 5 5 7
   1000004          CornSnackSpicy       Snack 1 5 0
   1000005          CornSnackSweet       Snack 3 5 0
   3000000       FishcakebarSpicy  Refrigerated 5 12 0
   3000001         FishcakebarBBQ  Refrigerated 4 12 0
   3000002           ChickenChest  Refrigerated 6 12 0
   4000000     SweetSpicyCupnoodle     Cupnoodle 2 3 12
   4000001   SpicyChickenCupnoodle     Cupnoodle 3 3 8
   5000000                 Sprite        Walkin 15 15 20
   5000001                   Pepci       Walkin 12 15 0
   5000002          PepciZeroLime       Walkin 7 15 0
Item to display on the stand
Peaktime = true
   1000000           ShirimpSnack ---> 4
   1000002      PotatochipOriginal ---> 2
   4000000     SweetSpicyCupnoodle ---> 1
1. Sell item  2. New item  3. Place item on stand  4. Add item in storage
5. Remove item 6. Save and Exit
5
Item Code: 5000002
```

```
      code                    name         stand
   1000000           ShirimpSnack       Snack 1 5 10
   1000002      PotatochipOriginal       Snack 3 5 30
   1000003         PotatochipOnion       Snack 5 5 7
   1000004          CornSnackSpicy       Snack 1 5 0
   1000005          CornSnackSweet       Snack 3 5 0
   3000000       FishcakebarSpicy  Refrigerated 5 12 0
   3000001         FishcakebarBBQ  Refrigerated 4 12 0
   3000002           ChickenChest  Refrigerated 6 12 0
   4000000     SweetSpicyCupnoodle     Cupnoodle 2 3 12
   4000001   SpicyChickenCupnoodle     Cupnoodle 3 3 8
   5000000                 Sprite        Walkin 15 15 20
   5000001                   Pepci       Walkin 12 15 0
Item to display on the stand
Peaktime = true
   1000000           ShirimpSnack ---> 4
   1000002      PotatochipOriginal ---> 2
   4000000     SweetSpicyCupnoodle ---> 1
```

5. 결과 및 결론

A. 제안서의 기능1, 기능2-1을 수정 보완 함

B. database를 vector<string>에서 구조체로 바꾸는 작업을 진행했다