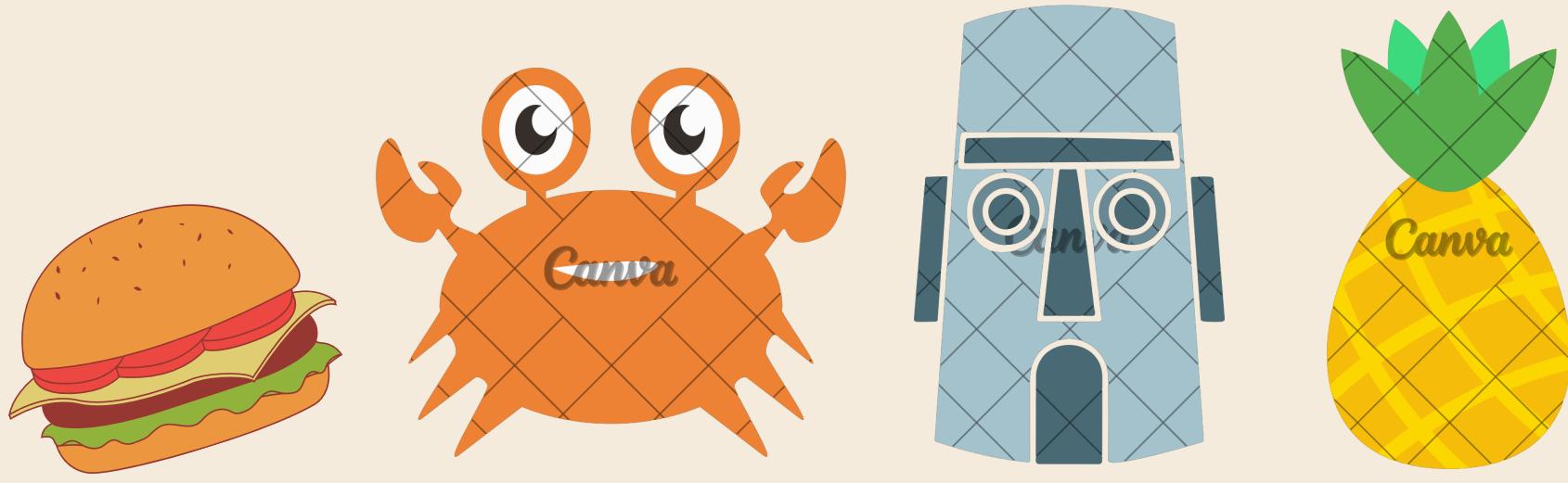


2024



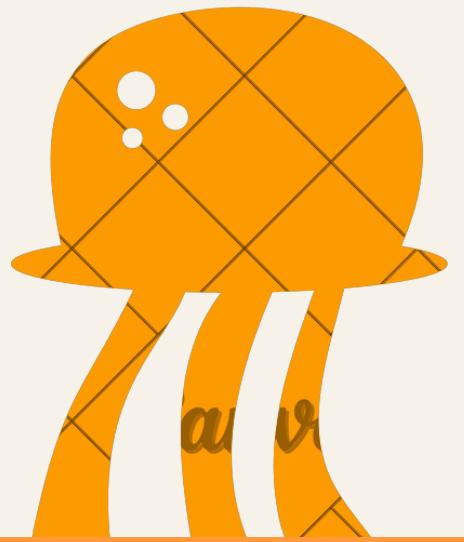
# 집게킹 포스기



C# 팀프로젝트

@Real Crab Burger

★ 강현욱 박준재 신지훈 정은희  
★ 정종철 조정진 고찬우



## 1. 주문 관리

고찬우, 조정진

- 주문 이벤트 기능 구현
- 다른 폼으로 이동 가능한 메인폼 구현

## 2. 결제

신지훈

- 결제기능 구현

## 3. 재고 관리

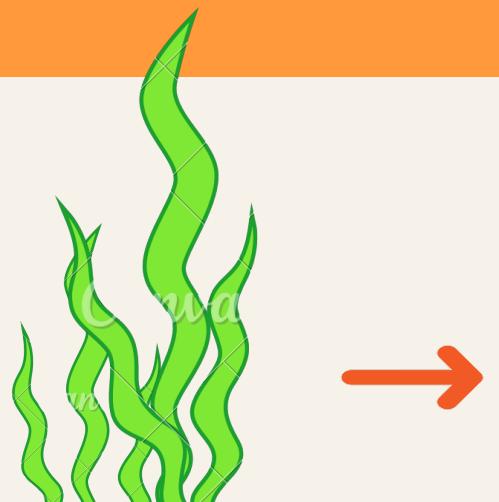
정은희, 정종철

- 메뉴별 재고 조회
- 재고 수정 기능 구현

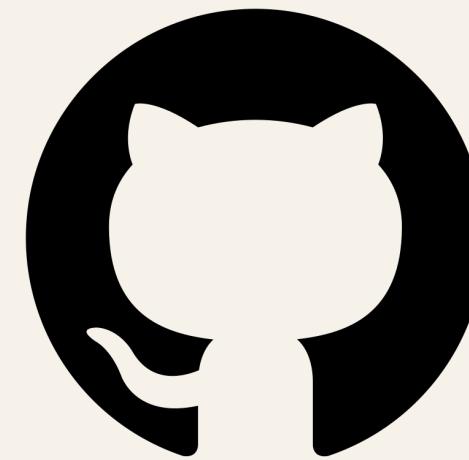
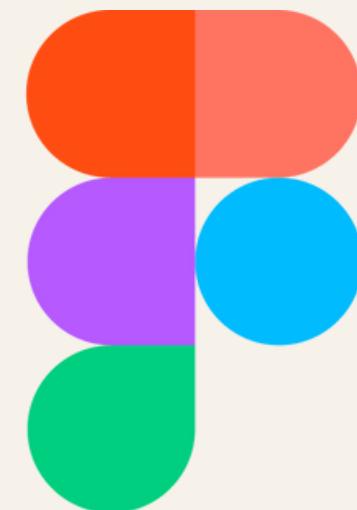
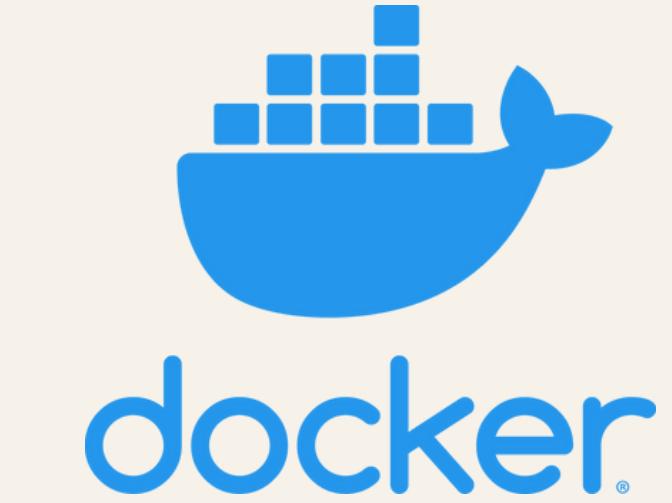
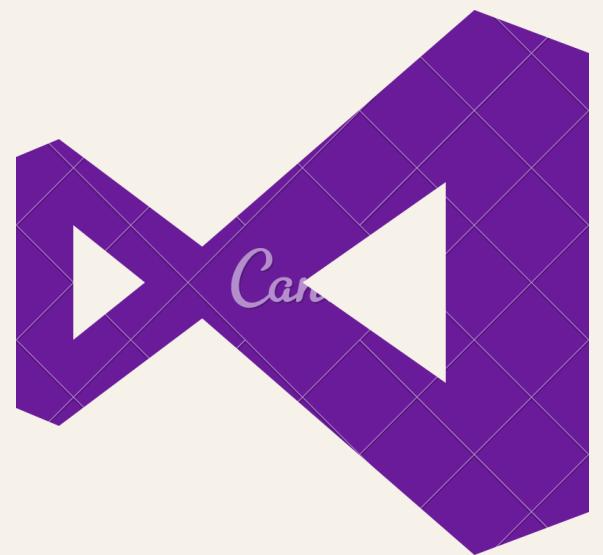
## 4. 매출 관리

강현욱, 박준재

- 전체 매출내역
- 날짜별 결제내역
- 현금, 카드 별 결제금액



# 사용도구



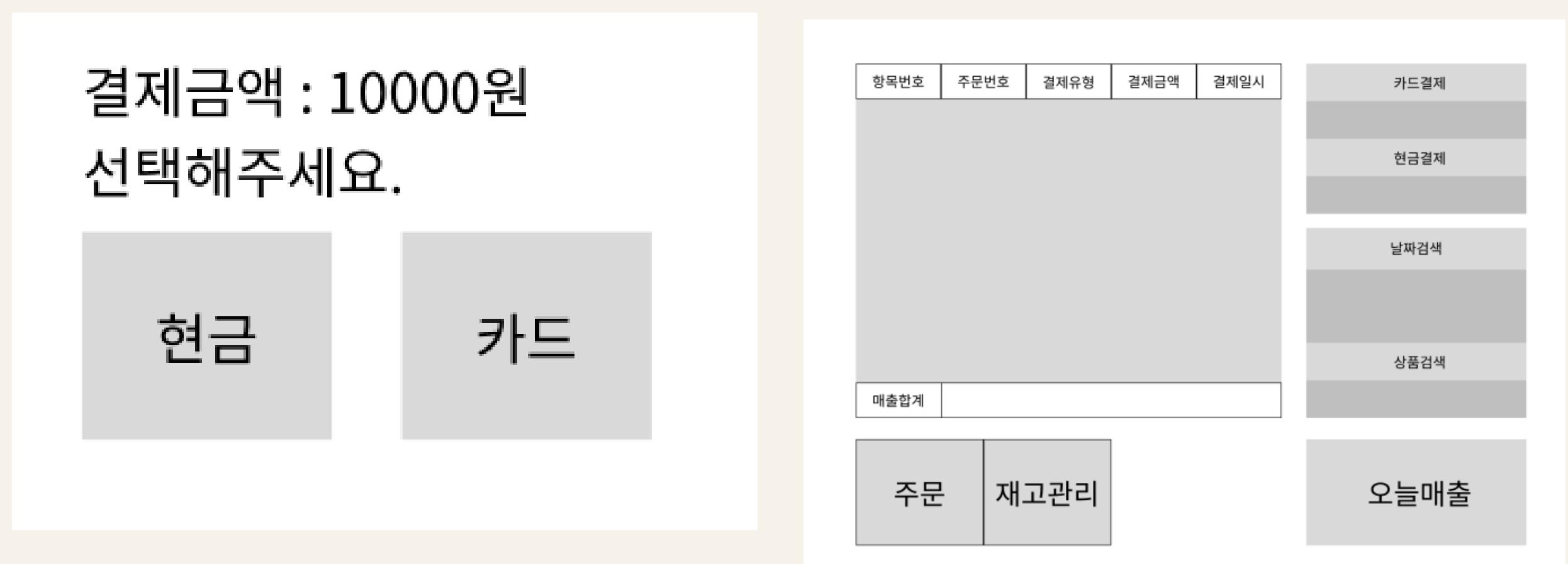
# 화면설계



결제금액 : 10000원  
선택해주세요.

현금

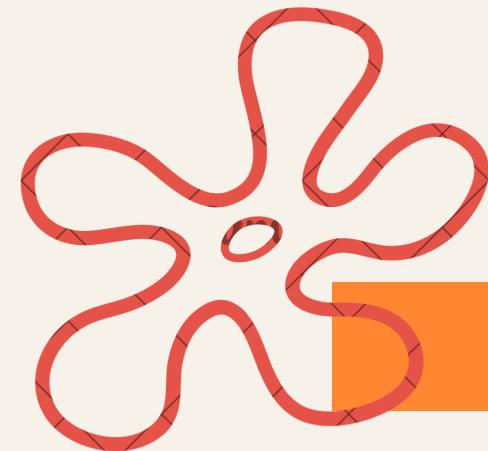
카드



DB

# Diagram





# 1. 주문 관리

고찬우 조정진



# 주문 관리

각 메뉴(메인,사이드,드링크)페이지 열기



# 주문 관리

```
참조 3개
private void Add_Form_To_Panel(Form form) // Form타입의 매개변수 form을 입력받는 메소드
{
    // 전달받은 품을 패널에 추가하고 화면에 표시.
    form.TopLevel = false; // 품을 부를때 TopLevel이면 다른패널에서 못씀, 그래서 false로 정의
    form.FormBorderStyle = FormBorderStyle.None; // form의 테두리 없애기
    form.Dock = DockStyle.Fill; // 패널에 꽉차게 하기
    panel1.Controls.Clear(); // 이미 추가된 품이 있으면 제거.
    panel1.Controls.Add(form); // 패널에 추가
    form.Show(); // 화면에 표시
}
참조 1개
private void Btn_Main_Menu_Click(object sender, EventArgs e) // 메인 메뉴 버튼을 누르면 버거 품 불러오기
{
    // OrderForm.Main 네임스페이스의 Burger 클래스에 접근하여 인스턴스를 생성.
    OrderForm.product_menu.Burger burger = new OrderForm.product_menu.Burger(this);

    // burger를 패널에 추가하고 화면에 표시.
    Add_Form_To_Panel(burger);
}
참조 1개
private void Btn_Side_Menu_Click(object sender, EventArgs e) // 메인 메뉴 버튼을 누르면 사이드 품 불러오기
{
    OrderForm.product_menu.Side side = new OrderForm.product_menu.Side(this);
    Add_Form_To_Panel(side);
}
참조 1개
private void Btn_Drink_Menu_Click(object sender, EventArgs e) // 메인 메뉴 버튼을 누르면 드링크 품 불러오기
{
    OrderForm.product_menu.Drink drink = new OrderForm.product_menu.Drink(this);
    Add_Form_To_Panel(drink);
}
#endifregion
```

```
namespace OrderForm.product_menu
{
    참조 4개
    public partial class Burger : Form
    {
        private Main_form main_form;
        참조 1개
        public Burger(Main_form main_form)
        {
            InitializeComponent();
            this.main_form = main_form;
        }
    }
}
```



# 주문 관리

메뉴 선택 시 Listview 에 값 보여주기



# 주문 관리

```
public void Load_Products_by_menu_id(int menu_id)
{
    try
    {
        if (db.OpenConnection())
        {
            Console.WriteLine("db 접속됨");
            string query = $"SELECT m.menu_name, i.stock, m.price " +
                $"FROM menu m " +
                $"INNER JOIN inventory i ON m.menu_id = i.menu_id " +
                $"WHERE m.menu_id = '{menu_id}'";
            MySqlCommand command = new MySqlCommand(query, db.GetConnection());
            MySqlDataReader reader = command.ExecuteReader();

            // ListView에 데이터 추가
            while (reader.Read())
            {
                string productName = reader.GetString("menu_name");
                int productPrice = reader.GetInt32("price");

                ListViewItem existingItem = Burger_Order_Listview.FindItemWithText(productName);

                if (existingItem != null)
                {
                    // 이미 존재하는 제품인 경우 price와 quantity를 증가시킴
                    int currentQuantity = int.Parse(existingItem.SubItems[1].Text); // 현재 quantity 가져오기
                    int newQuantity = currentQuantity + 1; // quantity 증가
                    existingItem.SubItems[1].Text = newQuantity.ToString(); // 증가된 quantity 설정

                    int currentPrice = int.Parse(existingItem.SubItems[2].Text); // 현재 price 가져오기
                }
            }
        }
    }
}
```

```
    else
    {
        // 새로운 제품인 경우 ListView에 추가
        ListViewItem item = new ListViewItem(productName);
        item.SubItems.Add("1"); // quantity를 항상 1로 설정
        item.SubItems.Add(productPrice.ToString());
        Burger_Order_Listview.Items.Add(item);
    }

    Update_Total_Price();
}

reader.Close();
}

catch (Exception ex)
{
    MessageBox.Show("데이터베이스 연결 오류: " + ex.Message);
}
finally
{
    // DB 연결 닫기
    if (db != null)
    {
        db.CloseConnection();
    }
}
```



# 주문 관리

Listview에 선택된 메뉴 취소

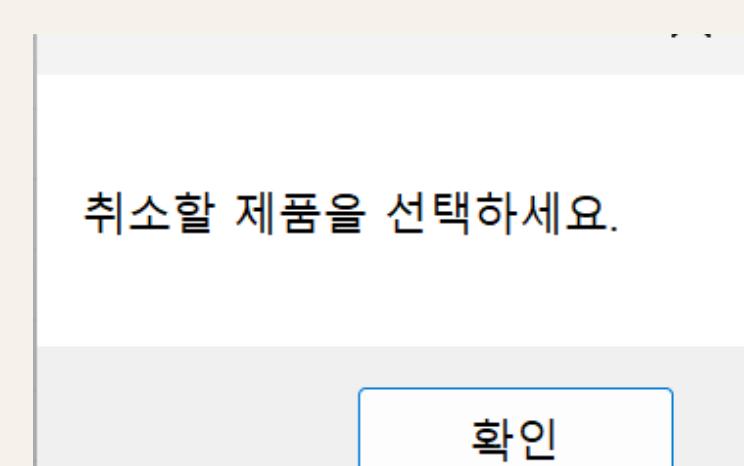
```
private void Delete_Selected_Menu()
{
    if (Burger_Order_Listview.SelectedItems.Count > 0)
    {
        ListViewItem selectedItem = Burger_Order_Listview.SelectedItems[0];
        int currentQuantity = int.Parse(selectedItem.SubItems[1].Text);
        if (currentQuantity > 1)
        {
            currentQuantity--;
            selectedItem.SubItems[1].Text = currentQuantity.ToString();
        }
        else
        {
            Burger_Order_Listview.Items.Remove(selectedItem);
        }
        Update_Total_Price();
    }
    else
    {
        MessageBox.Show("취소할 제품을 선택하세요.");
    }
}
```

메뉴	수량	가격
불고기와퍼	1	5500
와퍼	2	5000



메뉴	수량	가격
불고기와퍼	1	5500
와퍼	1	5000

메뉴	수량	가격
와퍼	1	5000



# 주문 관리

Listview에 선택된 메뉴 전체 취소

```
private void Btn_Cancel_all_Click(object sender, EventArgs e)
{
    Burger_Order_Listview.Items.Clear();
    Update_Total_Price();
}

참조 1개

private void Btn_Cancel_Selection_Click(object sender, EventArgs e)
{
    Delete_Selected_Menu();
}
```

메뉴	수량	가격
와퍼	1	5000
불고기와퍼	1	5500
게살와퍼	1	5500
새우버거	1	6500
게살버거	1	7500
집게리아...	1	8000



메뉴	수량	가격



# 주문 관리

총 금액 계산

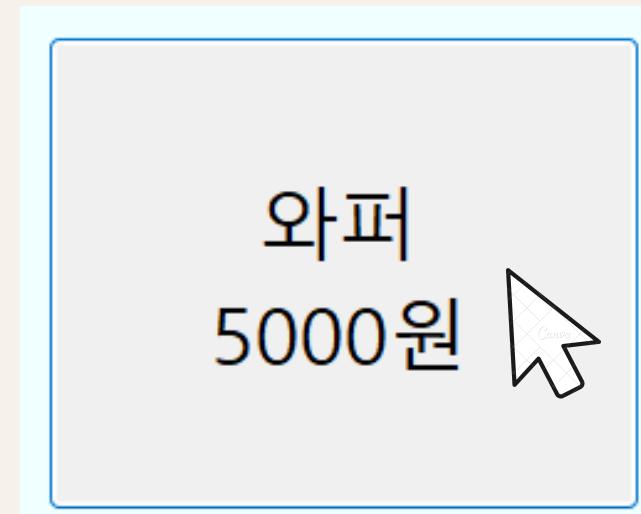
```
public void Update_Total_Price()
{
    int totalPrice = 0;

    foreach (ListViewItem item in Burger_Order_Listview.Items)
    {
        int quantity = int.Parse(item.SubItems[1].Text);
        int price = int.Parse(item.SubItems[2].Text);
        totalPrice += quantity * price;
    }

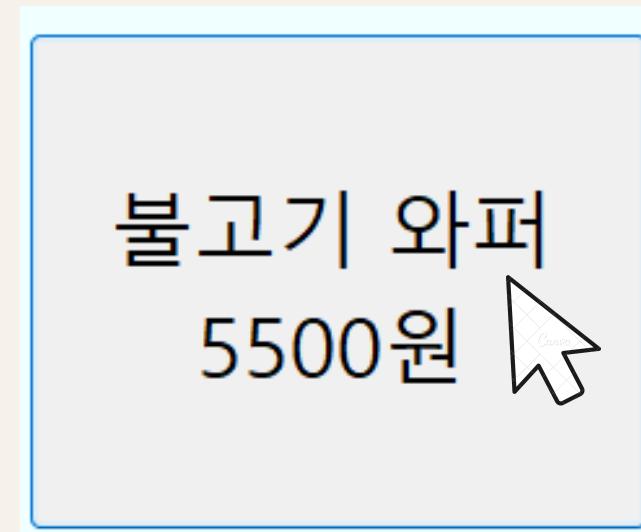
    Lb_Total_Price.Text = totalPrice.ToString();
}

참조 1개

public string Lb_Total_Price_P
{
    get { return Lb_Total_Price.Text; }
}
```



메뉴	수량	가격
와퍼	1	5000
총 금액		<b>5000</b>

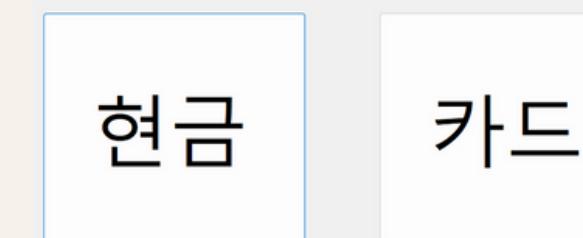


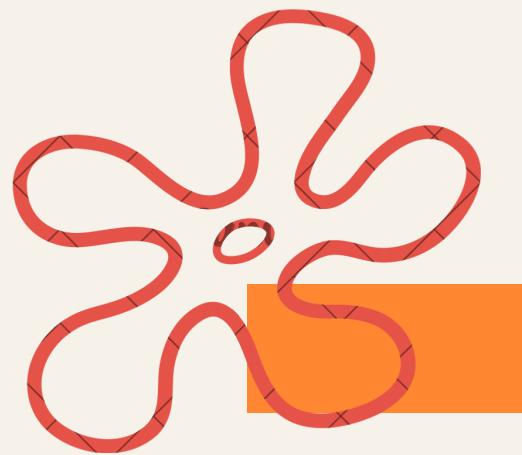
메뉴	수량	가격
와퍼	1	5000
불고기와퍼	1	5500
총 금액		<b>10500</b>



결제금액 **10500**

선택해주세요.





## 2. 결제

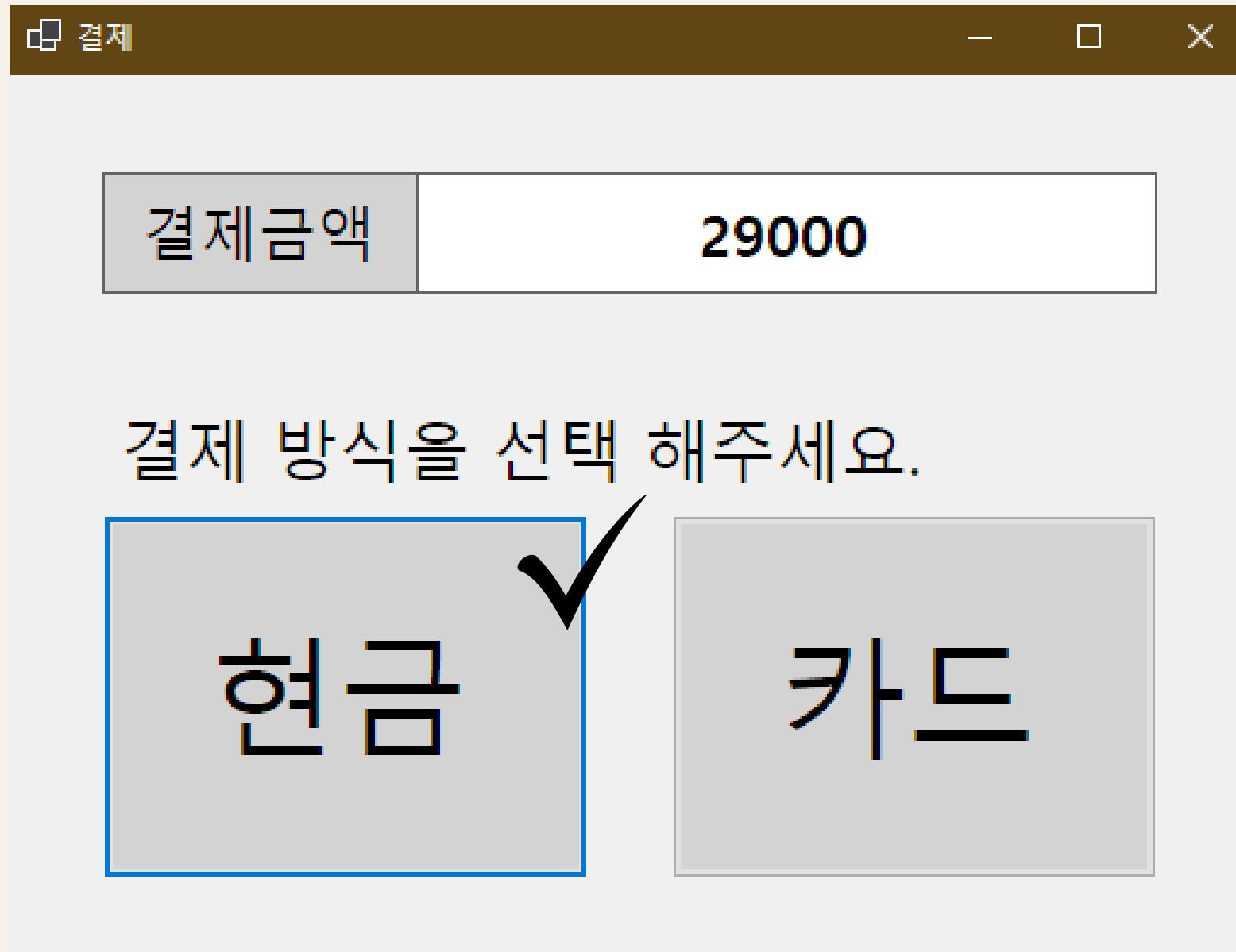


신지훈

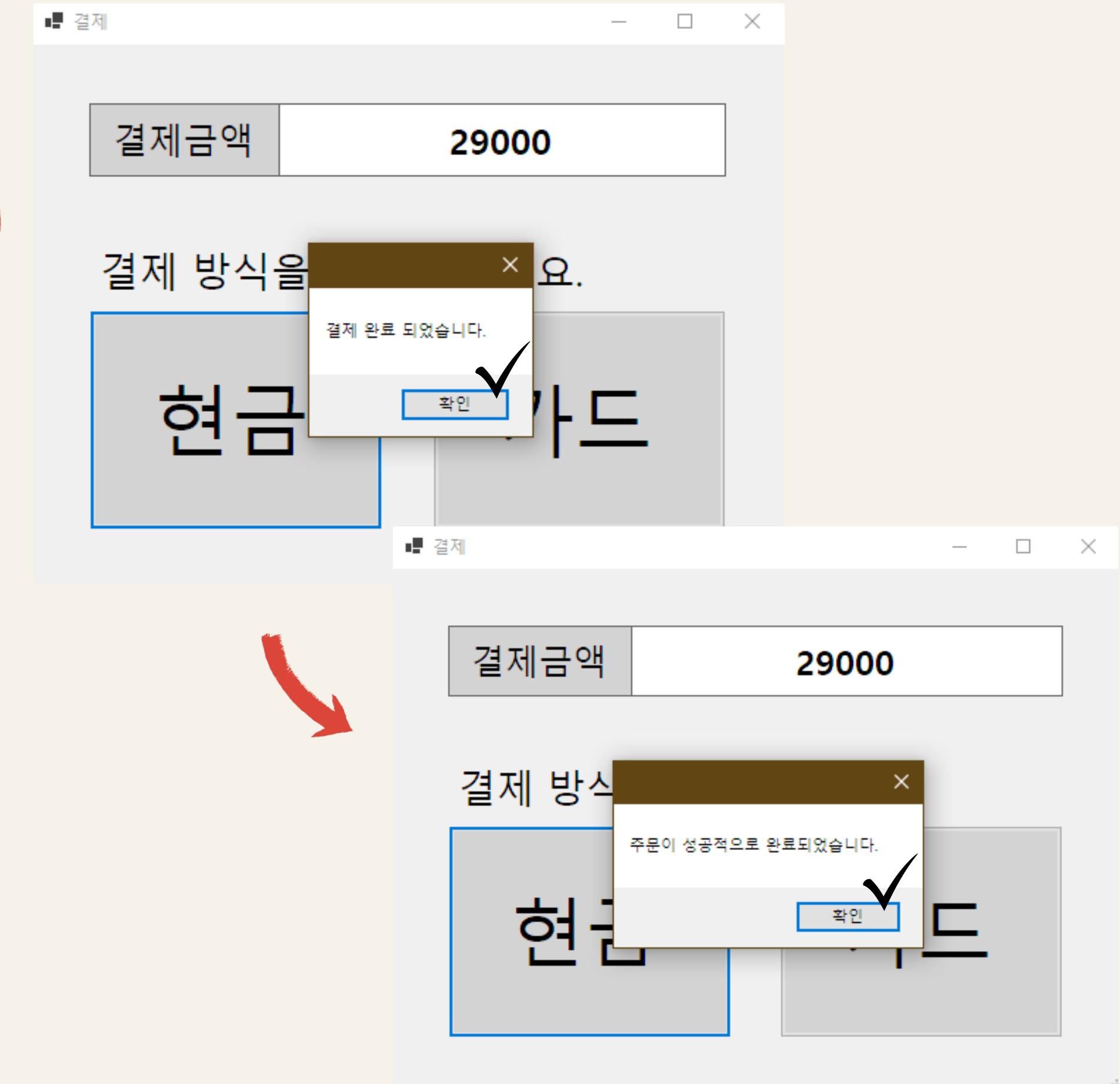


# 결제

## 1. 진행방식



결제 방식을 선택 해주세요.



# 결제

## 2. 주문 정보 저장

```
private void AddPaymentToDatabase(string payment_method, int orderId)
{
    try
    {
        if (db.OpenConnection())
        {

            string query = "INSERT INTO payment(payment_method, payment_amount, payment_time, order_id) " +
                "VALUES (@payment_method, @payment_amount, @payment_time, @order_id)";
            MySqlCommand command = new MySqlCommand(query, db.GetConnection());

            command.Parameters.AddWithValue("@order_id", orderId);
            command.Parameters.AddWithValue("@payment_method", payment_method);
            command.Parameters.AddWithValue("@payment_amount", Lb_Pay_Price.Text);
            command.Parameters.AddWithValue("@payment_time", DateTime.Now);

            command.ExecuteNonQuery();
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show("Error: " + ex.Message);
    }
    finally
    {
        if (db != null)
        {
            db.CloseConnection();
        }
    }
}
```

사용자가 주문 후 결제를 하고  
주문 완료시 주문 진행 때 저장된  
주문 번호와 결제 유형,  
총 결제 금액, 결제 시각을 DB에  
저장해주는 쿼리문

#	payment_id	order_id	payment_method	payment_amount	payment_time
1	1	1	현금	166,500	2024-03-11 17:12:53
2	2	2	카드	197,000	2024-03-11 17:47:30
3	3	3	현금	24,500	2024-03-10 17:51:23
4	4	4	카드	11,500	2024-03-10 17:51:30
5	5	5	현금	6,500	2024-03-12 17:51:37



# 결제

## 3. 주문 진행(주문 번호 생성)

```
// order 테이블에 order_id와 order_time을 저장하는 메서드  
참조 2개  
private int Save_Order()  
{  
    int orderId = 0; // 초기 주문 ID 설정 0으로 초기화  
    try  
    {  
        if (db.OpenConnection())  
        {  
  
            // order 테이블에 있는 order_time 컬럼에 NOW를 넣으면 현재시간이 등록된다  
            string query = $"INSERT INTO `order` (order_time) VALUES (@order_time)";  
            MySqlCommand command = new MySqlCommand(query, db.GetConnection());  
            command.Parameters.AddWithValue("@order_time", DateTime.Now);  
            command.ExecuteNonQuery(); // 쿼리 실행문  
  
            // 새로 생성된 order_id 가져오기  
            query = "SELECT LAST_INSERT_ID()";  
            // MySQL에서 마지막으로 삽입된 자동 증가(primary key) 열의 값을 가져오는 쿼리  
  
            command = new MySqlCommand(query, db.GetConnection());  
            // command.ExecuteScalar() 메서드를 통해 쿼리를 실행하고  
            // 그 결과로 반환된 첫 번째 행의 첫 번째 열의 값을 가져온다  
            // 그다음 Convert.ToInt32()를 사용하여 정수로 변환한 다음 orderId 변수에 넣는다.  
            // 새로운 주문이 들어왔을때 생성된 order_id 값을 가져오는데 사용  
            orderId = Convert.ToInt32(command.ExecuteScalar());  
        }  
    }  
}
```

주문 진행시 주문 메뉴가  
저장된 주문 번호(order id)와  
주문 시각을 저장하는 쿼리문을  
작성해줍니다.



#	order_id	order_time
1	1	2024-03-11 17:12:52
2	2	2024-03-11 17:47:30
3	3	2024-03-10 17:51:23
4	4	2024-03-10 17:51:30
5	5	2024-03-12 17:51:37



# 결제

## 4. 주문 정보 전달

```
public void SelectInventoryForOrderMenu(int orderId)
{
    try
    {
        if (db.OpenConnection())
        {
            string query = $"SELECT menu_id, quantity" +
                $"FROM order_menu " +
                $"WHERE order_id = '{orderId}'";
            MySqlCommand command = new MySqlCommand(query, db.GetConnection());
            command.ExecuteNonQuery();
            MySqlDataReader reader = command.ExecuteReader();

            while (reader.Read())
            {
                int menuId = reader.GetInt32("menu_id");
                int quantity = reader.GetInt32("quantity");
            }
        }
    }
}
```

주문 번호를 기반으로 order\_menu 테이블에서 주문된 메뉴의 메뉴 ID와 수량을 검색하고 추출 후 불러옵니다.  
그 후 order\_menu 테이블에 주문 번호와 메뉴 ID, 수량을 저장합니다.

```
private void Save_Order_Menu(int orderId, int menuId, int quantity)
{
    try
    {
        if (db.OpenConnection())
        {
            // order_menu 테이블에 위에서 정의한 orderId, menuId, 새로 받은 quantity 정보를 저장한다
            string query = $"INSERT INTO order_menu (order_id, menu_id, quantity) VALUES ({orderId}, {menuId}, {quantity})";
            MySqlCommand command = new MySqlCommand(query, db.GetConnection());
            command.ExecuteNonQuery(); // 쿼리 실행
        }
    }
}
```



# 결제

## 5. 결제 시스템

```
private void button1_Click(object sender, EventArgs e)
{
    try
    {
        if (db.OpenConnection())
        {
            int orderId = Save_Order(); // Save_Order 메서드를 호출한다
            AddPaymentToDatabase("현금", orderId);
            MessageBox.Show("결제 완료 되었습니다.");

            if (orderId > 0) // 0보다 클때
            {
                Dictionary<string, int> menuIdMap = GetAllMenuIds();
                // 각 메뉴별로 주문된 수량과 메뉴 ID를 가져와서 order_menu 테이블에 저장
                foreach (ListViewItem item in main_Form.Burger_Order_Listview.Items)
                {
                    string menuName = item.SubItems[0].Text;
                    int quantity = int.Parse(item.SubItems[1].Text);

                    // 메뉴 이름으로부터 메뉴 ID를 가져오는 메서드 호출
                    int menuId = menuIdMap[menuName];
                    SelectInventoryForOrderMenu(orderId);

                    // order_menu 테이블에 주문 정보 저장
                    Save_Order_Menu(orderId, menuId, quantity);

                    UpdateInventoryStock(menuId, quantity);
                }
            }
        }
    }
}
```

전달 받은 주문 정보를 기반으로  
결제 유형 선택 후 결제를 진행  
합니다.

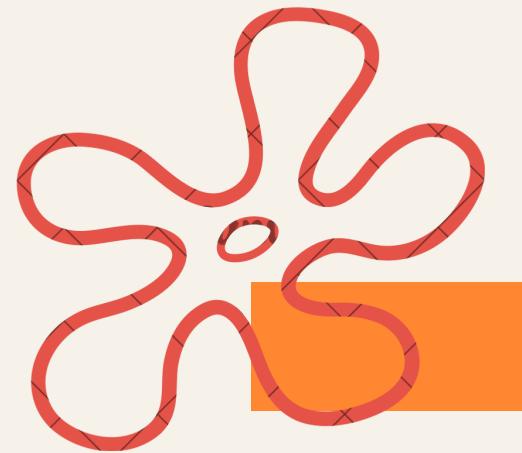
```
        MessageBox.Show("주문이 성공적으로 완료되었습니다.");
        // 주문 완료 후 ListView 초기화
        main_Form.Burger_Order_Listview.Items.Clear();
        main_Form.Update_Total_Price();

    }
    else
    {
        MessageBox.Show("주문을 처리하는 동안 오류가 발생했습니다.");
    }
}

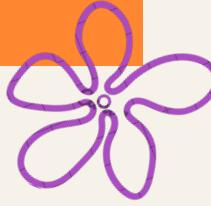
catch (Exception ex)
{
    MessageBox.Show("데이터베이스 연결 오류: " + ex.Message);
}
finally
{
    if (db != null)
    {
        db.CloseConnection();
    }
}

this.Close();
```





## 3.재고관리



정은희 정종철



# 재고관리

집게킹 포스기

■ 재고관리

- □ ×

전체 버거 사이드 음료

	상품이름	가격	수량
▶	21치즈스틱	3500	10
	게살버거	7500	46
	게살와퍼	5500	6
	고래버거	7000	21
	네겟킹8조각	4500	39
	바삭킹	3500	25
	불고기와퍼	5500	8
	새우버거	6500	7
	쉐이킹프라이	3000	9
	쉬림프와퍼	6000	19
	스프라이트	2000	10
	스프라이트 제로	2000	18
	아메리카노	2500	21
	언니언링	3000	7

재고 관리

현재수량

수량변경

입력

재고 변동 확인

주문 매출관리



# 재고관리

재고관리

재고 관리			
전체	버거	사이드	음료
상품이름	가격	수량	
▶ 21치즈스틱	3500	10	
게살버거	7500	46	
게살와퍼	5500	6	
고래버거	7000	21	
너겟킹8조각	4500	39	
바삭킹	3500	25	
불고기와퍼	5500	8	
새우버거	6500	7	
쉐이킹프라이	3000	9	
쉬림프와퍼	6000	19	
스프라이트	2000	10	
스프라이트 제로	2000	18	
아메리카노	2500	21	
언니언링	3000	7	

재고 변동 확인

주문 매출관리

전체	버거	사이드	음료
상품이름	가격	수량	
▶ 21치즈스틱	3500	7	
게살버거	7500	4	
게살와퍼	5500	5	
고래버거	7000	4	
너겟킹8조각	4500	12	
바삭킹	3500	7	
불고기와퍼	5500	14	
새우버거	6500	4	
쉐이킹프라이	3000	7	
쉬림프와퍼	6000	5	
스프라이트	2000	10	
스프라이트 제로	2000	9	
아메리카노	2500	5	
언니언링	3000	6	
와퍼	5000	-2	

전체	버거	사이드	음료
상품이름	가격	수량	
▶ 21치즈스틱	3500	7	
너겟킹8조각	4500	12	
바삭킹	3500	7	
쉐이킹프라이	3000	7	
언니언링	3000	6	
코콜솔로	2000	7	
코코넛쉬림프	4500	7	
크리미모짜볼	4000	6	
해쉬브라운	2500	7	

전체	버거	사이드	음료
상품이름	가격	수량	
▶ 스프라이트	2000	10	
스프라이트 제로	2000	9	
아메리카노	2500	5	
코카콜라	2000	6	
코카콜라 제로	2000	5	



# 재고관리

DataGridview에 전체 재고 테이블 연결

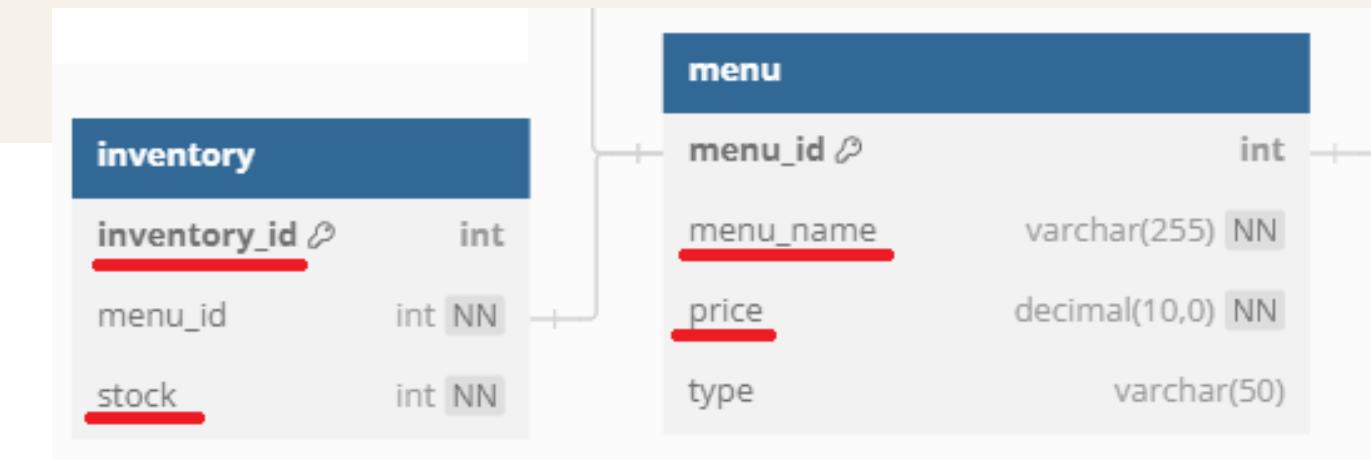
```
public void dataSelect() // 전체 재고를 보여준다
{
    MySqlConnection conn = new MySqlConnection(Connection);
    try
    {
        conn.Open();

        // columns 초기화
        inventory_dataGridView.Columns.Clear();

        // DataGridView에 칼럼
        inventory_dataGridView.Columns.Add("ID", "ID");
        inventory_dataGridView.Columns["ID"].Visible = false;
        inventory_dataGridView.Columns.Add("상품이름", "상품이름");
        inventory_dataGridView.Columns.Add("가격", "가격");
        inventory_dataGridView.Columns.Add("수량", "수량");

        //DataGridView에 DB연결
        MySqlCommand cmd = new MySqlCommand("SELECT inventory.inventory_id, menu.menu_name, menu.price, inventory.stock FROM " +
            "inventory INNER JOIN menu ON inventory.menu_id = menu.menu_id ORDER BY menu.menu_name;", conn);

        MySqlDataReader reader = cmd.ExecuteReader();
    }
}
```



```
while (reader.Read())
{
    object[] row = {
        reader["inventory_id"].ToString(),
        reader["menu_name"].ToString(),
        reader["price"].ToString(),
        reader["stock"].ToString()
    };
    inventory_dataGridView.Rows.Add(row);
}

inventory_dataGridView.Columns["ID"].Width = 30;
inventory_dataGridView.Columns["상품이름"].Width = 145;
inventory_dataGridView.Columns["가격"].Width = 145;
inventory_dataGridView.Columns["수량"].Width = 145;

}
catch (Exception ex)
{
    // columns 초기화
    MessageBox.Show("Error: " + ex.Message);
}
finally
{
    if (conn.State == ConnectionState.Open)
        conn.Close();
}
```



# 재고관리

DataGridview에 버거 테이블 연결

```
public void burger_Select() // 사이드 메뉴를 보여준다
{
    MySqlConnection conn = new MySqlConnection(Connection);

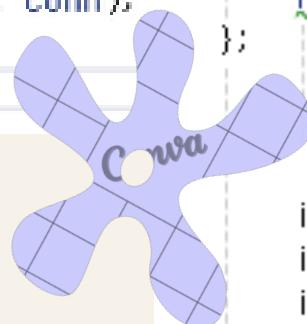
    try
    {
        conn.Open();

        // columns 초기화
        inventory_dataGridView.Columns.Clear();

        // DataGridView에 칼럼
        inventory_dataGridView.Columns.Add("ID", "ID");
        inventory_dataGridView.Columns["ID"].Visible = false;
        inventory_dataGridView.Columns.Add("상품이름", "상품이름");
        inventory_dataGridView.Columns.Add("가격", "가격");
        inventory_dataGridView.Columns.Add("수량", "수량");

        MySqlCommand cmd = new MySqlCommand("SELECT inventory.inventory_id, menu.menu_name, menu.price, inventory.stock FROM " +
            "inventory INNER JOIN menu ON inventory.menu_id=menu.menu_id WHERE menu.type='버거' ORDER BY menu.menu_name", conn);

        MySqlDataReader reader = cmd.ExecuteReader();
    }
}
```



```
while (reader.Read())
{
    object[] row = {
        reader["inventory_id"].ToString(),
        reader["menu_name"].ToString(),
        reader["price"].ToString(),
        reader["stock"].ToString()
    };
    inventory_dataGridView.Rows.Add(row);
}

inventory_dataGridView.Columns["ID"].Width = 30;
inventory_dataGridView.Columns["상품이름"].Width = 160;
inventory_dataGridView.Columns["가격"].Width = 150;
inventory_dataGridView.Columns["수량"].Width = 150;
}

catch (Exception ex)
{
    // 예외처리
    MessageBox.Show("Error: " + ex.Message);
}
finally
{
    if (conn.State == ConnectionState.Open)
        conn.Close();
}
```

An orange arrow pointing to the right, indicating the continuation of the code block.

# 재고관리

	상품이름	가격	수량
▶	21치즈스틱	3500	10
	게살버거		46
	게살와퍼		6
	고래버거	7000	21
▶	너겟킹8조각	4500	39
	바삭킹	3500	25
	불고기와퍼	5500	8
	새우버거	6500	7
	쉐이킹프라이	3000	9
	슈림프와퍼	6000	19
	스프라이트	2000	10
	스프라이트 제로	2000	18
	아메리카노	2500	21
	언니언링	3000	7

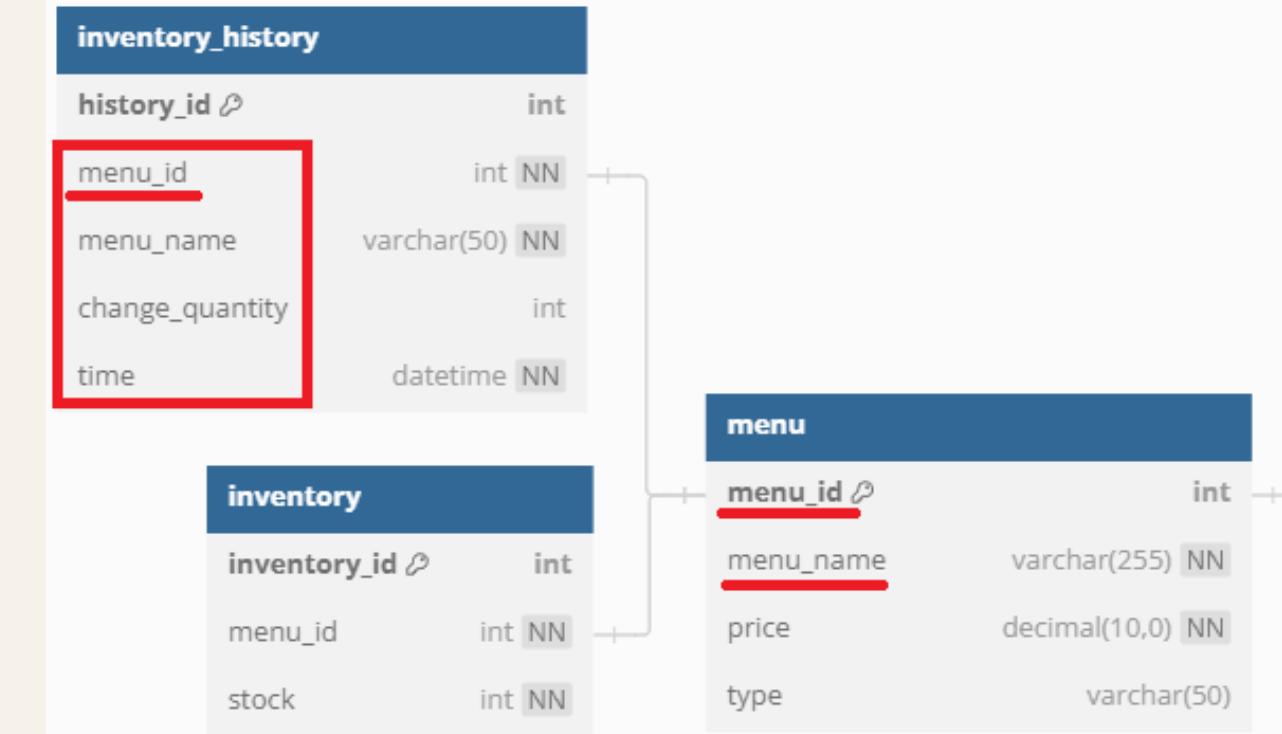
```
// 현재수량을 currentstock_label에 표시  
참조 1개  
private void inventory_dataGridView_CellClick  
    (object sender, DataGridViewCellEventArgs e)  
{  
    if (e.RowIndex != -1)  
    {  
        currentstock_label.Text  
            = Convert.ToString(inventory_dataGridView.CurrentRow.Cells[3].Value);  
    }  
}
```

```
// inventory 테이블 업데이트  
string updateSql =  
    "UPDATE inventory SET stock = stock + @stock " +  
    "WHERE inventory_id = @inventory_id";  
MySqlCommand updateCmd = new MySqlCommand(updateSql, conn, transaction);  
updateCmd.Parameters.AddWithValue("@stock", quantityToAdd);  
updateCmd.Parameters.AddWithValue("@inventory_id", inventoryId);  
updateCmd.ExecuteNonQuery();
```



# 재고관리

	상품이름	변동량	시간
▶	너겟킹8조각	10	2024-03-14 오후...
	해쉬브라운	10	2024-03-13 오후...
	스프라이트	5	2024-03-13 오후...
	21치즈스틱	1	2024-03-13 오후...
	언니언링	1	2024-03-13 오후...
	코울슬로	10	2024-03-13 오후...
	코코넛쉬림프	10	2024-03-13 오후...
	스프라이트 제로	15	2024-03-13 오후...
	코카콜라 제로	15	2024-03-13 오후...
	코카콜라	30	2024-03-13 오후...
	크리미모짜볼	10	2024-03-13 오후...
*	플랑크톤버거	19	2024-03-13 오후...



```

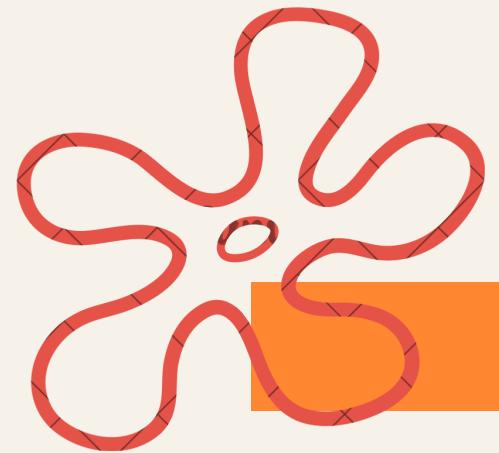
// menu_name 바탕으로 menu_id를 가져오기
int menuId = Convert.ToInt32(selectMenuIdCmd.ExecuteScalar());
string selectMenuIdSql =
    "SELECT menu_id FROM menu WHERE menu_name = @menu_name";
MySqlCommand selectMenuIdCmd = new MySqlCommand(selectMenuIdSql, conn, transaction);
selectMenuIdCmd.Parameters.AddWithValue("@menu_name", menuName);

// inventory_history 테이블 업데이트
string insertHistorySql =
    "INSERT INTO inventory_history (menu_id, menu_name, change_quantity, time) " +
    "VALUES (@menu_id, @menu_name, @change_quantity, @time);";
MySqlCommand insertHistoryCmd = new MySqlCommand(insertHistorySql, conn, transaction);
insertHistoryCmd.Parameters.AddWithValue("@menu_id", menuId);
insertHistoryCmd.Parameters.AddWithValue("@menu_name", menuName);
insertHistoryCmd.Parameters.AddWithValue("@change_quantity", quantityToAdd);
insertHistoryCmd.Parameters.AddWithValue("@time", DateTime.Now);
insertHistoryCmd.ExecuteNonQuery();

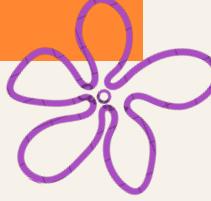
// inventory_history 시간 내림차순으로 보여주기
MySqlCommand cmd = new MySqlCommand(
    "SELECT * FROM inventory_history order by time desc", conn);

```





## 4. 매출관리



강현욱 박준재



# 매출관리

집게킹 포스기

매출관리

주문번호	결제방법	결제금액	결제시간
1	현금	166500	2024-03-11 오후 5:12:53
2	카드	197000	2024-03-11 오후 5:47:30
3	현금	24500	2024-03-10 오후 5:51:23
4	카드	11500	2024-03-10 오후 5:51:30
5	현금	6500	2024-03-12 오후 5:51:37
6	카드	9500	2024-03-11 오후 6:08:57
7	현금	50000	2024-03-11 오후 6:14:17
8	카드	25000	2024-03-12 오후 1:59:53
9	카드	21000	2024-03-12 오후 2:00:11
10	현금	7000	2024-03-12 오후 2:04:50
11	현금	98500	2024-03-12 오후 3:11:12
12	카드	33500	2024-03-12 오후 3:11:27
13	현금	16000	2024-03-12 오후 3:23:36
14	현금	5000	2024-03-12 오후 3:30:02
총 합계		1104000원	

현금합계  
679500원

카드합계  
424500원

2024년 3월						
일	월	화	수	목	금	토
25	26	27	28	29	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31	1	2	3	4	5	6

오늘: 2024-03-13

오늘매출

주문 재고관리



# 매출관리

집게킹 포스기

전체 내역 조회!!

```
참조 2개
public sale_payment()
{
    InitializeComponent();
    listSelect();
}

#region 판매한 금액 list
참조 1개
public void listSelect()
{
    int totalCash = 0;
    int totalCard = 0;

    MySqlConnection conn = new MySqlConnection(Connection);
    conn.Open();
    MySqlCommand cmd = new MySqlCommand("SELECT * FROM payment", conn);
    MySqlDataReader reader = cmd.ExecuteReader();
    while (reader.Read())
    {
        ListViewitem item = new ListViewitem(reader["order_id"].ToString());
        item.SubItems.Add(reader["payment_method"].ToString());
        item.SubItems.Add(reader["payment_amount"].ToString());
        item.SubItems.Add(reader["payment_time"].ToString());

        if (reader["payment_method"].ToString() == "현금")
        {
            totalCash += Convert.ToInt32(reader["payment_amount"].ToString());
        }
        if (reader["payment_method"].ToString() == "카드")
        {
            totalCard += Convert.ToInt32(reader["payment_amount"].ToString());
        }

        listView1.Items.Add(item);
    }
    textBox1.Text = totalCash.ToString() + "원";
    textBox2.Text = totalCard.ToString() + "원";
}
```

주문번호	결제방법	결제금액	결제시간
1	현금	166500	2024-03-11 오후 5:12:53
2	카드	197000	2024-03-11 오후 5:47:30
3	현금	24500	2024-03-10 오후 5:51:23
4	카드	11500	2024-03-10 오후 5:51:30
5	현금	6500	2024-03-12 오후 5:51:37
6	카드	9500	2024-03-11 오후 6:08:57
7	현금	50000	2024-03-11 오후 6:14:17
8	카드	25000	2024-03-12 오후 1:59:53
9	카드	21000	2024-03-12 오후 2:00:11
10	현금	7000	2024-03-12 오후 2:04:50
11	현금	98500	2024-03-12 오후 3:11:12
12	카드	33500	2024-03-12 오후 3:11:27
13	현금	16000	2024-03-12 오후 3:23:36
14	현금	5000	2024-03-12 오후 3:30:02

현금합계  
632000원  
카드합계  
424500원

현금 따로  
카드 따로  
확인!!



# 매출관리

집게킹 포스기

```
while (reader.Read())
{
    ListViewItem item = new ListViewItem(reader["order_id"].ToString());
    item.SubItems.Add(reader["payment_method"].ToString());
    item.SubItems.Add(reader["payment_amount"].ToString());
    item.SubItems.Add(reader["payment_time"].ToString());

    if (reader["payment_method"].ToString() == "현금")
    {
        totalCash += Convert.ToInt32(reader["payment_amount"].ToString());
    }
    if (reader["payment_method"].ToString() == "카드")
    {
        totalCard += Convert.ToInt32(reader["payment_amount"].ToString());
    }

    listView1.Items.Add(item);
}

textBox1.Text = totalCash.ToString() + "원";
textBox2.Text = totalCard.ToString() + "원";

// 총원 담기
int totalPrice = 0;

foreach (ListViewItem item in listView1.Items)
{
    int price = int.Parse(item.SubItems[2].Text);
    totalPrice += price;
}

label4.Text = totalPrice.ToString() + "원";
```

주문번호	결제방법	결제금액	결제시간
1	현금	166500	2024-03-11 오후 5:12:53
2	카드	197000	2024-03-11 오후 5:47:30
3	현금	24500	2024-03-10 오후 5:51:23
4	카드	11500	2024-03-10 오후 5:51:30
5	현금	6500	2024-03-12 오후 5:51:37
6	카드	9500	2024-03-11 오후 6:08:57
7	현금	50000	2024-03-11 오후 6:14:17
8	카드	25000	2024-03-12 오후 1:59:53
9	카드	21000	2024-03-12 오후 2:00:11
10	현금	7000	2024-03-12 오후 2:04:50
11	현금	98500	2024-03-12 오후 3:11:12
12	카드	33500	2024-03-12 오후 3:11:27
13	현금	16000	2024-03-12 오후 3:23:36
14	현금	5000	2024-03-12 오후 3:30:02

총 합계 1104000원

조회된 내역의 결제금액  
총 합계 출력!!



# 매출관리

집게킹 포스기

```
private void Today_Payment_Click(object sender, EventArgs e)
{
    listView1.Items.Clear();
    int totalCash = 0;
    int totalCard = 0;

    MySqlConnection conn = new MySqlConnection(Connection);
    conn.Open();
    MySqlCommand cmd = new MySqlCommand("SELECT * FROM payment WHERE DATE(payment_time) = CURDATE()", conn);
    MySqlDataReader reader = cmd.ExecuteReader();
    while (reader.Read())
    {
        ListViewItem item = new ListViewItem(reader["order_id"].ToString());
        item.SubItems.Add(reader["payment_method"].ToString());
        item.SubItems.Add(reader["payment_amount"].ToString());
        item.SubItems.Add(reader["payment_time"].ToString());

        if (reader["payment_method"].ToString() == "현금")
        {
            totalCash += Convert.ToInt32(reader["payment_amount"].ToString());
        }
        if (reader["payment_method"].ToString() == "카드")
        {
            totalCard += Convert.ToInt32(reader["payment_amount"].ToString());
        }

        listView1.Items.Add(item);
    }
    textBox1.Text = totalCash.ToString() + "원";
    textBox2.Text = totalCard.ToString() + "원";
}
```

오늘매출			
주문번호	결제방법	결제금액	결제시간
29	현금	5000	2024-03-13 오전 10:32:...
30	현금	5000	2024-03-13 오전 10:34:...
31	현금	5000	2024-03-13 오전 10:36:...
32	현금	5000	2024-03-13 오전 10:38:...
33	현금	5000	2024-03-13 오전 10:39:...
34	현금	22500	2024-03-13 오전 10:40:...
35	현금	10000	2024-03-13 오후 2:43:41
36	현금	65000	2024-03-13 오후 2:48:39
37	현금	28500	2024-03-13 오후 2:49:47
38	현금	28500	2024-03-13 오후 2:51:13
39	현금	8500	2024-03-13 오후 2:52:05
40	현금	13000	2024-03-13 오후 2:56:51

오늘 매출 확인!!



# 매출관리

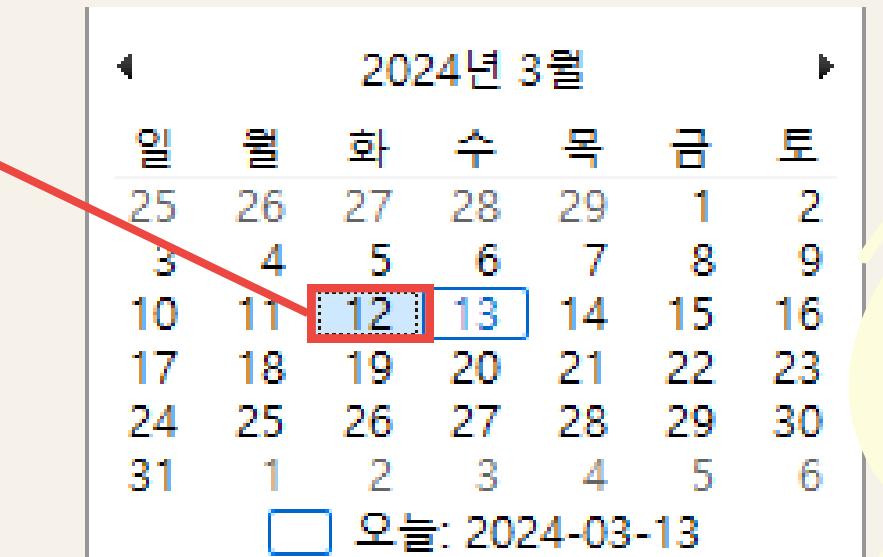
집게킹 포스기

```
private void monthCalendar1_DateSelected(object sender, DateRangeEventArgs e)
{
    DateTime selectedDate = monthCalendar1.SelectionStart.Date;
    listView1.Items.Clear();
    int totalCash = 0;
    int totalCard = 0;

    MySqlConnection conn = new MySqlConnection(ConnectionString);
    conn.Open();
    MySqlCommand cmd = new MySqlCommand("SELECT * FROM payment WHERE DATE(payment_time) = @selectedDate", conn);
    cmd.Parameters.AddWithValue("@selectedDate", selectedDate);
    MySqlDataReader reader = cmd.ExecuteReader();
    while (reader.Read())
    {
        ListViewItem item = new ListViewItem(reader["order_id"].ToString());
        item.SubItems.Add(reader["payment_method"].ToString());
        item.SubItems.Add(reader["payment_amount"].ToString());
        item.SubItems.Add(reader["payment_time"].ToString());

        if (reader["payment_method"].ToString() == "현금")
        {
            totalCash += Convert.ToInt32(reader["payment_amount"].ToString());
        }
        if (reader["payment_method"].ToString() == "카드")
        {
            totalCard += Convert.ToInt32(reader["payment_amount"].ToString());
        }

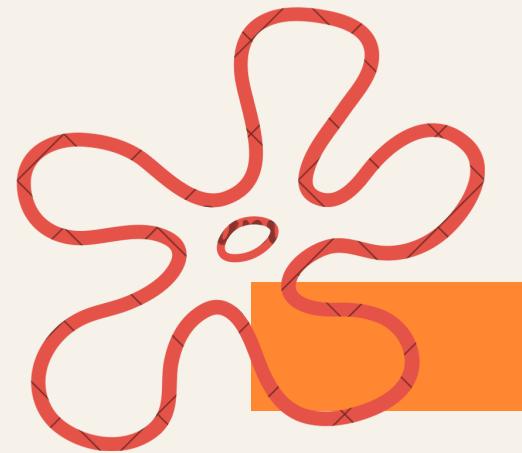
        listView1.Items.Add(item);
    }
    textBox1.Text = totalCash.ToString() + "원";
    textBox2.Text = totalCard.ToString() + "원";
}
```



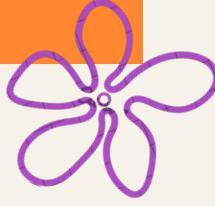
주문번호	결제방법	결제금액	결제시간
5	현금	6500	2024-03-12 오후 5:51:37
8	카드	25000	2024-03-12 오후 1:59:53
9	카드	21000	2024-03-12 오후 2:00:11
10	현금	7000	2024-03-12 오후 2:04:50
11	현금	98500	2024-03-12 오후 3:11:12
12	카드	33500	2024-03-12 오후 3:11:27
13	현금	16000	2024-03-12 오후 3:23:36
14	현금	5000	2024-03-12 오후 3:30:02
15	현금	29000	2024-03-12 오후 3:52:45
16	현금	9000	2024-03-12 오후 4:54:43
17	카드	16500	2024-03-12 오후 4:54:53
18	카드	13500	2024-03-12 오후 5:05:42
19	현금	8500	2024-03-12 오후 5:05:48
20	현금	11500	2024-03-12 오후 5:22:24

총 합계 597500원

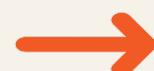
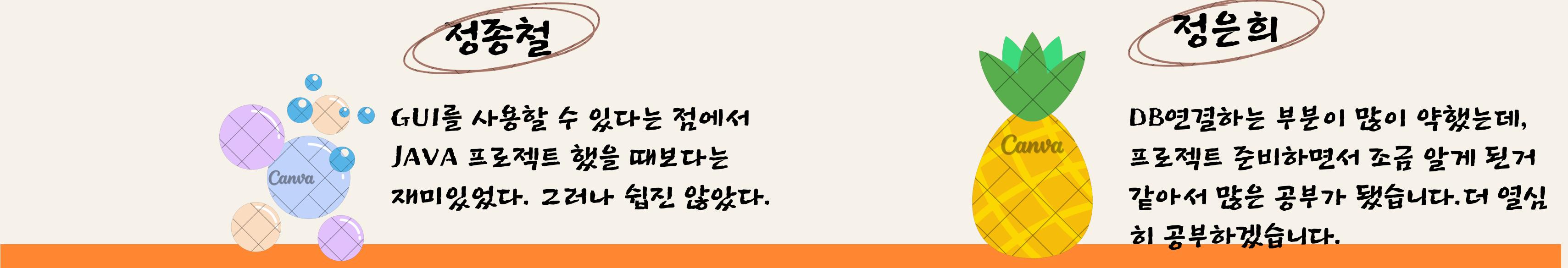
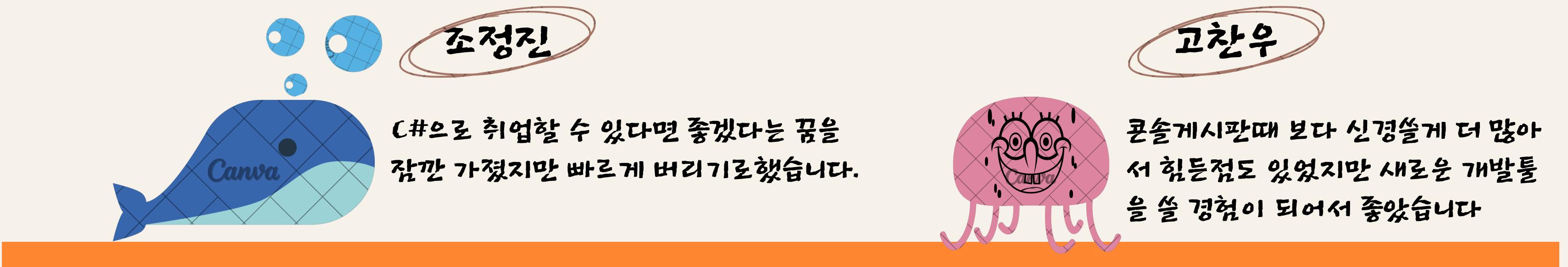


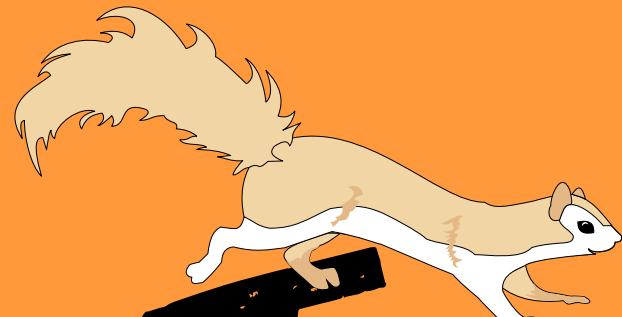


시연









# THANK YOU!

감사합니다

