# SJAM Bakery

Team Members: Navreet Dhillon, Cameron Hazelwood, Kyungbin Son

## Brief Description of Program

Bakeries consist of mouth watering sweets that attract many customers of all age and gender. These delicious treats can uplift and comfort individuals who had a long and exhausting day. However, sometimes ordering can be time consuming when the customer has wait in a long line. **SJAM Bakery** is a simple, yet convenient program that simulates an online ordering system for a bakery that allows the customer to just click and choose. With many different features all in one single program, it is an unique and user-friendly system.

## Product Description

The program can calculate the total cost of the ordered pastries, and also display the list of ingredients. This program is very useful as it instantly provides the information requested by the consumer.

### Features Included in the Program:

#### MENU Tab Features

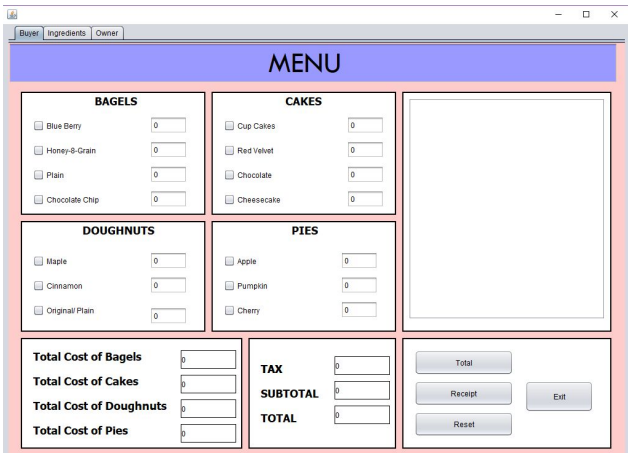
- ➢ Place the quantity of the pastry/pastries to order
- ➢ Separate display of the total cost of each type pastry (ex. Total Cost of Bagels, Total Cost of Cakes)
- ➢ Total cost of all of the pastries before and after taxation
- ➢ Receipt with the date, and the time of purchase
- ➢ Option to EXIT, or RESET the purchases made

#### INGREDIENTS Tab

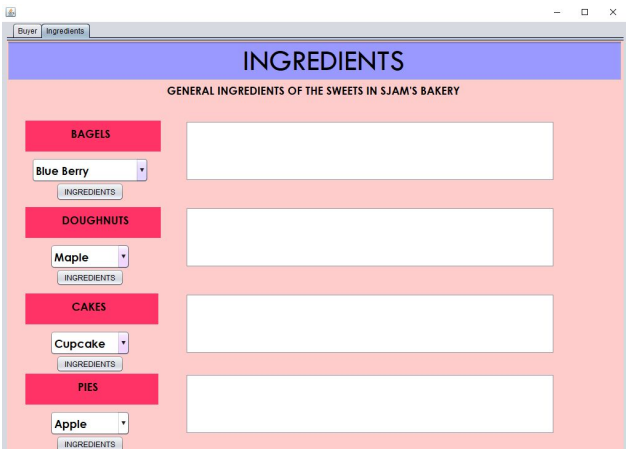
- ➢ Option to check the list of ingredients of the chosen type of pastry

## Design

The design and user interface of the program is clean and simple. It is laid out so that the user of the program can easily understand and use. As suggested by research, appetite increasing colours of light pink and blue were used.




First Tab




Second Tab

## Sample Code

### Inheritance - *How our team implemented inheritance in our project.*

```
}*/

public void addIngredients(String ingredient) {
    if (!this.ingredientsList.contains(ingredient)) {
       this.ingredientsList.add(ingredient);
    }
}

public ArrayList<String> getIngredients() {
    return this.ingredientsList;
}
public String describe() {

    String a = "HackTheBakery have " + quantity + " " + type + "\n" + "Ingredients: ";

    for (int i = 1; i < ingredientsList.size(); i++) {
        String c [] ;
         a += "\n" + i  + ") " + ingredientsList.get(i)  + " \n";
    }

    return a;



   }
}
```

In this line of code, our team created a class called **BakingGoods** that would have properties such as the price, temperature, and the ingredients list(array list).

```java
package Bakeries;

import Bakeries.BakingGoods;


public abstract class  Pies extends BakingGoods {
    public Pies(String type , String ingredient) {
        super(type + "Pies", 360, 20, 1);
        super.addIngredients("flour");
        super.addIngredients("oil");
        super.addIngredients("sprinkle");
        super.addIngredients(ingredient);
    }

    public Pies (BakingGoods bakingGoods) {
        super(bakingGoods);
    }

}
```

In this line of code, we created a class called **Pies** that extended from baking goods. It then added ingredients such as flour, oil and sprinkle.

```java
package Bakeries;
public class ApplePies extends Pies {
    public ApplePies (){
        super ("Apple" , "Apples");
    }
}
```

We then created a class name **ApplePies** that inherited from Pies and it had its own feature as it was made of flour, oil, sprinkle, and **apples.**

## Ingredients Tab - *Using Inheritance to describe the ingredients*

```java
    private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {
        Pies s = new ApplePies();
        Pies p1 = new PumpkinPies();
        Pies p2 = new CherryPies();

        if (piesDropMenu.getSelectedItem() == "Apple"){
            PiesIngredients.setText(s.describe());
        }

        if (piesDropMenu.getSelectedItem() == "Pumpkin"){
            PiesIngredients.setText(p1.describe());
        }
        if (piesDropMenu.getSelectedItem() == "Cherry"){
            PiesIngredients.setText(p2.describe());
        }

    }
```

This example code is called when the INGREDIENTS button is clicked for pies section of the ingredients tab. Then it creates an object for the different types of pies (apple, pumpkin, cherry). When the user chooses a kind of pie from the drop menu, it calls the inheritance code that describes the ingredients.

## Testing

To test the project, our teammates tried the program ourselves to continuously check if the program provided the desirable output. To add on, we presented the program to our peers with no coding experience. The users all agreed that the program was very simple to understand and convenient to use.

## Future Features

1. **Health Status**

To improve the program, we plan on adding another tab with information of how these baked goods impact the health of the consumers. It will display the calories, how much exercising is needed to burn the calories, and other information such as the expiry date of the pastries.

2. **Review Section**

We will also add a tab where the customers can add reviews of the pastries in the bakery so future customers can keep in mind which baked good they should invest in.