Assignment 1: Cake Baby Bakery

The purpose of this readme is to understand the Cake Baby Bakery's web application development. For that reason I would follow the same steps followed in the code to better explain it. We have to mention that all this code explained is the script code as all the functionality has been built using Javascript.

1. Variables – Build objects

First of all we need to declare the objects that we need for this application. In this case, we have declared all the ingredients as an Ingredient object having each one a name, measurement and cost properties. This is an example:

```
var flour = new cakeBakery.Ingredients("Flour",  "cup", 1);
```

Then, we have created an array of Products that we have called products. We also have created an array with all the ingredients for the chocolate cake and another one with the cookies' ingredients. Then, we have created the Products object having the name and ingredients properties each and we have declared the chocolate cake and cookies as the following example:

```
var chocolateCake= new cakeBakery.Products("Chocolate Cake",
ingredientsChocolateCake);
```

To finish that, we have pushed both products in the array products, so we have our chocolate cake and cookies stored in a products array and each of its products elements have its name and ingredients.

2. Build some HTML into the page

First of all, we need to recover our main div that is going to be the whole page that we will create. This div container is the only one that we have in our html code as it was included there in the original code and it is called cakeHook. We have (getElementById) using Javascript and rename it to myHookDiv and we would attach every div we will create to it using the function appendChild.

In this section we also have included another div called title to be the title of the webpage.

3. function renderRecipe(aProduct)

First of all we have created the div needed to display the recipe of the product we would like to show. Using the appendChild function we have attached it to our main div myHookDiv. We also have created another div to show the name of the product we are showing.

This function needs to have an argument passed. This argument is going to be the product to show. We have called it aProduct. We have thinked it is better to save code and be more organized. So, to use this function we have to code:

```
renderRecipe(products[0]);
renderRecipe(products[1]);
```

In this case, we have in products[0] position the chocolate cake with its name and ingredients and in products[1] the cookies also with its name and ingredients. So, we would show the recipe of the chocolate cake first using the renderRecipe function and after that, the cookies recipe.

```
for (var i = 0; i < aProduct.ingredients.length; i++){

var productIngredients = document.createElement("p");
productIngredients.innerHTML = aProduct.ingredients[i].name + " "
+aProduct.ingredients[i].quantity + " " +aProduct.ingredients[i].measurement;

container.appendChild(productIngredients);
        }
```

Using this for, we would obtain all the ingredients for the product we are showing and display them creating a p html element each iteration for each ingredient.
Then, when we have the entire recipe, we would show a button that allows us to add to the master list of ingredients and calculate the total of the day using the onclick Javascript property and the function we have called addtoMasterList() and calculateTotal().

4. function addtoMasterList(aProduct)

Before creating this function we have to create the div needed to display the master list that we have called container and the title for it. Moreover, we have to initialize the variables we would use in this function.

The main function of this function is to go through all the ingredients of the product

we are evaluating using the for iteration and identify all of them using the if's sentences.

```
for (var i = 0; i < aProduct.ingredients.length; i++){

        //This is an example of the identification

        if(aProduct.ingredients[i].name==flour.name){
```

For each ingredient identified we would create a p hmtl element to show the total quantity used during the day, measurement and its name. To calculate the total quantity we would add to a variable the quantity of this iteration using the sentence:

```
flourused+=aProduct.ingredients[i].quantity;
```

5. function calculateTotal (aProduct)

As previously, we first have to create the div element needed to display the total cost of the day.

Then, we would go through all the ingredients of the aProduct and calculate its total cost using the following code:

```
var ingredientsproductTotal = aProduct.ingredients[i].quantity *
aProduct.ingredients[i].cost;
costmeasurement.push(ingredientsproductTotal);
```

Then, we would push it into the array costmeasurement, and we would calculate all the sum of this array and add to the total of the day.

```
for(var j = 0; j < costmeasurement.length; j++){

        totaloftheday += costmeasurement[j];

                                }
```

6. Graduate extension

To do the graduate extension we have included all the script in the function:

```
(function (cakeBakery){

        })(window.cakeBakery=window.cakeBakery||{});
```

We also have created reusable objects:

```
cakeBakery.Ingredients = function(name,measurement,cost){
                        this.name=name;
                        this.measurement=measurement;
                        this.cost=cost;
                }
cakeBakery.Products= function(name,ingredients){
                        this.name=name;
                        this.ingredients=ingredients;
                }
```

So, to create the ingredients or products we just need to code like the following:

```
var sugar = new cakeBakery.Ingredients("Sugar", "cup",  0.5);
var cookies= new cakeBakery.Products("Cookies", ingredientsCookies);
```

7. CSS

To make our submission better we have included some CSS using inline CSS in the style section of our html code and we also have included some CSS in the script code using the code as the following:

```
var container2 = document.createElement("div");
                container2.style['background-color']='#ffe7f3';
                container2.style['border-style']='solid';
                container2.style['border-color']='#ff80c0';
```

Moreover, we have included some Google fonts using the link added in the <head> section to make our fonts beautiful.