

Due 10/11/13 @ 11:59pm



Custom Cupcakes Online Store

**Project** **Description**

Your Collaborative Project team is making a bid to create an online web store for Custom Cupcakes (a massive cupcakery) called **CustomCupcakes.com**. The online store will allow users to order cupcakes with a variety of cupcake flavors, fillings, frostings, and toppings. As a part of your bid, your team needs to submit a functional prototype to Custom Cupcakes to demonstrate your team’s talent and hopefully win the bid. Your job is to create a 3-tier architecture comprised of:

1. SQL Database
2. PHP Middleware
3. Web client

Custom Cupcakes has provided a “rough sketch” of how they envision the website to look. They understand that your bid is only a **functional prototype** so they are NOT expecting your application to look like a finalized product. However, they are expecting your team to meet all project requirements and, at a minimum, implement their rough sketch.

**Project Requirements**

From the website, users will be able to handle the following requirements:

**Unregistered Customer User**

1. Create a user account. *See UI Requirements section for more details*

**Registered Customer User**

1. Create an order of cupcakes
   1. An order consists of 1 or more cupcakes.
2. Design/Create a cupcake
   1. Choose 1 cupcake flavor
   2. Choose 0 or 1 cupcake filling
   3. Choose 1 icing
   4. Pick 0-N toppings
   5. Select quantity
   6. Add to order
3. View Order
4. Review the details of a cupcake design
5. Edit and update the details of a cupcake design
6. View Order’s Total Cost
7. Submit order
8. Save cupcake design as a “favorite.” A “favorite” is one single cupcake design with a user defined name.

**Employee User**

1. View sale analytics on the following:
   1. Cupcake flavors sold (as a pie chart).
   2. Filling Flavors sold (as a pie chart).
   3. Icing Flavors sold (as a pie chart).
   4. Topping sold (as a bar graph).

DB Requirements

**Create a Database:**

Create a database in MySql that is sufficiently robust to store all data needed for the project. Minimally, it should handle customers, orders, favorites, and employee information. Where appropriate, add code to enforce primary keys. You may add foreign key constraints as well.

**Populate your Database:**

Use the menu.json file to extract data to populate your database for cupcakes, fillings, frosting, and toppings. In addition, use the 3 csv files for pre existing user information and favorite cupcake designs. You need to be able to parse both the .json and .csv files to collect the proper information for populating your DB.

**SQL Queries:**

You’re responsible for creating the SQL queries which provide the UI developers with the cupcake flavors, fillings, frosting, and toppings. When returning the result of your queries, you’ll need to work with the UI developers for outputting the desired HTML using the PHP *echo* command.

**Create an API:**

Create a PHP API which provides the following:

1. Returns JSON of a registered user’s “favorite” cupcake designs.
2. Receives JSON for adding a registered user’s new “favorite” cupcake design.
3. Returns JSON which contains the sales information for cupcake flavors, fillings, frosting, and toppings.

This API should be a PHP Class that has appropriate functionality to achieve the above three tasks. Your solution should have other php files that use this class as a common interface to your datastore.

**UI Requirements**

The following requirements will be used to inform the UI development and the database design. Your UI will need to handle everything here and the database will need to store it.

**Login and Account Creation Screen:**

This will be the first page that registered and unregistered visitors can view. This screen will provide the following functionality:

1. **User Login:** Allows a registered user to enter an email address and password and allow them to login to make an order.
   1. Before submitting information to the server, client side validation will occur to ensure the following:
      1. email address is not empty a actually a valid email address
      2. That the password field is not empty and is greater than or equal to 8 characters.
   2. Upon **successful** user authentication, the user will automatically be redirected to the order page.
   3. If user authentication **fails**, the user will remain on the login page and a error will be displayed to the user explaining why authentication failed.
2. **Account Creation:** Allows an unregistered user to create an account.
   1. This form includes the following fields and validation:
      1. Join our mailing list
      2. First Name - cannot be empty
      3. Last Name - cannot be empty
      4. Email - must match the format of a traditional email address, cannot be empty
      5. Password - cannot be empty and must be greater than or equal to 8 characters.
      6. Telephone Number - cannot be empty and must be equal to 10 digits. Use js validation to strip all non digits from this field and check if the value is equal to 10 digits.
      7. Address - cannot be empty
      8. City - cannot be empty
      9. State - cannot be empty
      10. Zipcode - cannot be empty and needs to contain at least 5 digits.
   2. If any of the previous fields fail to validate, provide a meaningful error message to the user which describes to the user why the field(s) failed.

**Order Screen:**

Provides a registered customer the ability to design a cupcake and add it to their order, edit their order, save a cupcake design as a favorite, select a pre-existing favorite cupcake and add it to an order.

1. **Favorites Menu:** A menu that shows a registered user’s past cupcake designs. Functionality includes:
   1. Retrieving a user’s favorites via XMLHTTPRequest. The server will return a JSON string representing the user’s favorite cupcake designs.
   2. Using JSON, dynamically populate the user’s favorites menu by creating HTML elements and inserting them into the DOM tree using the DOM api.
   3. Allows the user to select a favorite cupcake and have it be applied to the current **Cupcake Editor**. This will allow the user the chance to quickly order quantities of their favorite cupcake or make small modifications.
   4. When a user saves a new favorite cupcake, this menu will update to reflect the recent addition.
2. **Cupcake Editor:** Allows the user to choose/update a cupcake design by specifying the cupcake, filling, and frosting flavor along with toppings. Functionality includes:
   1. Allow the user to make a cupcake, filling, frosting, and topping selection(s).
   2. Clear all selected toppings
   3. Reset the entire cupcake back to default
   4. Select cupcake quantity
   5. Adding the cupcake design and quantity to the order
   6. Adding the cupcake design to a user’s favorites
   7. All items (cupcake flavor, filling, frosting, toppings) will be populated using PHP by reading information from the DB and then echoing out html.
3. **Order Menu:** Allows the user to view their current order, remove a cupcake design from the order, edit a cupcake design, and/or submit their cupcake order. Functionality includes:
   1. When a user adds a cupcake design via the **Cupcake Editor**, the design and quantity will be added.
   2. Allows the user to delete/remove a cupcake design.
   3. Allows the user to select the cupcake design in order to make modification to the design and/or quantity using the **Cupcake Editor**.
   4. Allows the user to submit their order where it will be processed by the server.
   5. All items added via the **Cupcake Editor** will be dynamically generated using Javascript.
4. When a user successfully submits their order, navigate the user to the Employee Analytics screen. *\*Obviously a real application wouldn’t do this, but for this assignment it is okay.\**

**Analytics Screen:**

Provides the employee an analytical view of which cupcake flavors, fillings, frostings, and toppings are selling the best. Functionality includes:

1. This page should not require a login. *\*Obviously, once again, a real application wouldn’t do this.\**
2. Use an XMLHTTPRequest, from your client, to fetch JSON, from your server, which provides the sales information about all cupcake flavors, fillings, frostings, and toppings.
3. Cupcake Flavor Pie Chart - a pie chart diagramming the cupcake flavor sales.
4. Cupcake Filling Pie Chart - a pie chart diagramming the cupcake filling sales.
5. Cupcake Frosting Pie Chart - a pie chart diagramming the cupcake frosting sales.
6. Cupcake Toppings Bar Graph - a bar graph diagramming the cupcake toppings sales.

Create the graphs using [Chart.js](http://www.chartjs.org/docs/) (or your own preferred graphing library). Follow the instructions from the documentation for creating the graphs. You’ll need to become slightly familiar with the <canvas/> element in order to implement the graphs. See a past [CSE 3345 lecture](http://lyle.smu.edu/~craley/3345/sp13/lectures/L17-Canvas.pptx) for info.

**Project Resources**

The following [resources](http://lyle.smu.edu/~craley/3345/hwk/a6/A6Resources.zip) were provided by Custom Cupcake:

1. Artwork for cupcake flavors and frostings.
2. Paper prototypes
3. Menu and User DB information.

**Additional Requirements**

* This project is designed to give you experience for your collaborative project. So you are required to use Git,GitHub, and your production server to host this project. This means you need to create a separate repository for this assignment. Additionally, with Apache, creating a separate virtualHost would be highly recommended and encouraged. This will allow you to separate your production code for this homework assignment from your production code for the semester project.
* Document your php interface thoroughly. While you don’t need to expose a public API necessarily, your private API should be sufficiently documented so that a new developer to your company could get up to speed very quickly. Documentation exists inside and outside of the actual php files. You could research a php api documentation engine that could extract comments in the code to a beautiful web api. The other option is for you to code it yourself.
* Provide a ER Diagram for the database schema. Indicate relationships as needed.
* Bonus: Set up Apache so that the analytics documents and api documentation can be accessed from ONLY an SMU IP address (129.119.\*.\*). In the documentation, explain how you accomplished this.

**Deliverables**

Submit all deliverables by October 11th @ 11:59pm to the **Assignment 6** link on the CSE 3345 Blackboard page**.** Each team only needs one submission, so a single team member can upload for the entire team. You will need to provide the following:

1. a README file that includes:
   1. A link to your GitHub repo (We will be examining your code so make it is pretty and well documented).
   2. A link to your Custom Cupcake website.
   3. Indicate whether or not you attempted the Bonus.
2. ER Diagram
3. A SQL Script file to create the database and tables. It should include other constraints such as primary key and foreign key constraints as well.