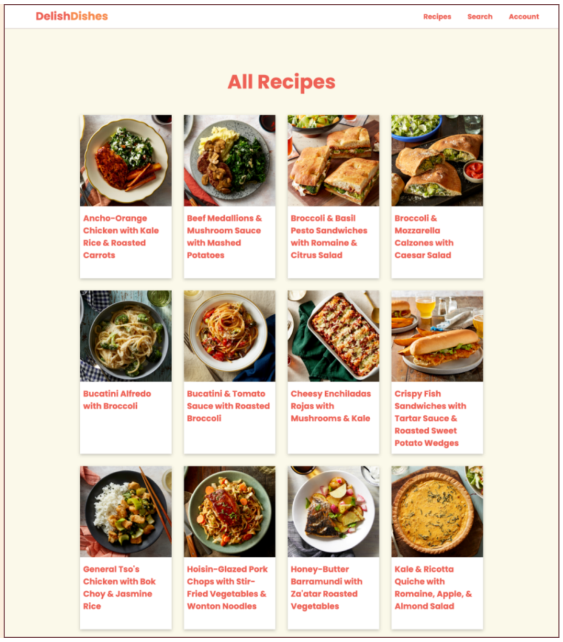
Scripting II Case Study

Recipes web application built using PHP, MySQL, HTML, CSS, and JavaScript



All Recipes

- Ancho-Orange Chicken with Kale Rice & Roasted Carrots
- Beef Medallions & Mushroom Sauce with Mashed Potatoes
- Broccoli & Basil Pesto Sandwiches with Romaine & Citrus Salad
- Broccoli & Mozzarella Calzones with Caesar Salad
- Bucatini Alfredo with Broccoli
- Bucatini & Tomato Sauce with Roasted Broccoli
- Cheesy Enchiladas Rojas with Mushrooms & Kale
- Crispy Fish Sandwiches with Tartar Sauce & Roasted Sweet Potato Wedges
- General Tso's Chicken with Bok Choy & Jasmine Rice
- Hoisin-Glazed Pork Chops with Stir-Fried Vegetables & Wonton Noodles
- Honey-Butter Barramundi with Za'atar Roasted Vegetables
- Kale & Ricotta Quiche with Romaine, Apple, & Almond Salad

The Overview

The goal of this project was to create a fully responsive recipe web application that dynamically displays recipes stored in a database, using PHP and MySQL as the primary programming and data management languages. The final solution allows users to create new recipes, edit or delete existing recipes, and view or search for recipes stored in a database. Before coding this project, I designed each screen in Figma. I referenced these designs when I was writing the code for these pages.

Context and Challenge

Background / Timeline / Purpose

I created this project for my Scripting II class, and the timeline for the project was ten weeks. This class taught me the fundamentals of PHP and MySQL. Throughout the semester, I implemented PHP and MySQL along with HTML, CSS, and JavaScript to build a dynamic, responsive recipe web application.

The Problem

I was used to building static web pages using HTML, but I understood the importance of learning how to build dynamic web pages. Building dynamic web pages makes it easier to make changes to the content shown on any given screen, creating a faster and less repetitive process. PHP and MySQL allowed me to fetch and display information from a database and create components that could be used across several web pages. Being a novice at PHP and MySQL before starting this project presented several difficulties, but I was able to overcome these issues to build my web application.

Goals and Objectives

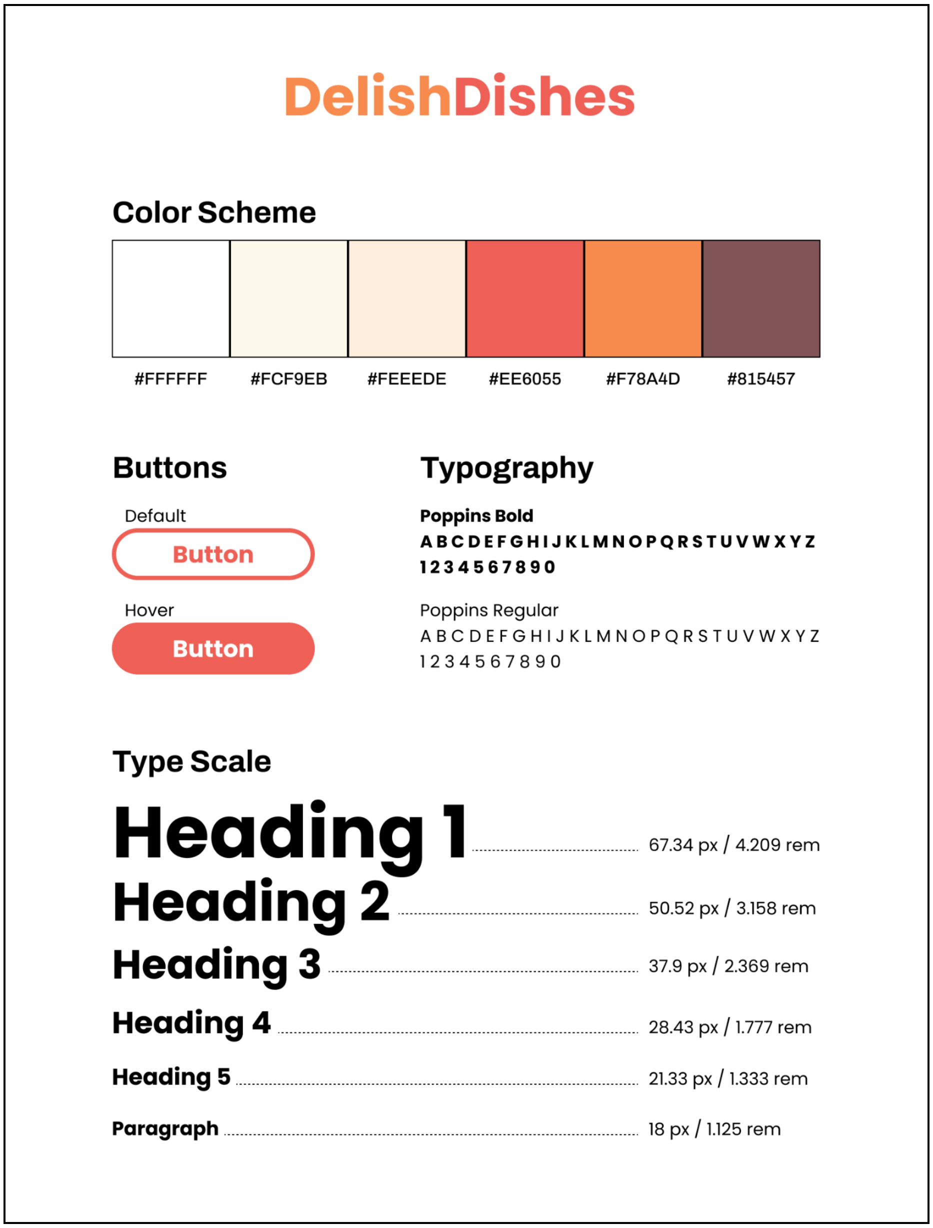
The project will be considered successful if the code functions smoothly. The following features will be implemented on the final recipe web application and function as intended:

- Fetch and display forty recipes from a database
- Create new recipe
- Edit or delete existing recipes
- Search for recipes, showing results or a “No Results Found” message depending on the user’s search query

Process and Insight

Creating a Style Tile

The style tile below shows how I wanted my recipes web application to look. After doing some research, I realized that many recipe websites used bright orange colors. I decided to take a similar approach and incorporate bright colors throughout my design.



DelishDishes

Color Scheme

#FFFFFF	#FCF9EB	#FEEEDE	#EE6055	#F78A4D	#815457

Buttons

Default

Button

Hover

Button

Typography

Poppins Bold

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

1 2 3 4 5 6 7 8 9 0

Poppins Regular

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

1 2 3 4 5 6 7 8 9 0

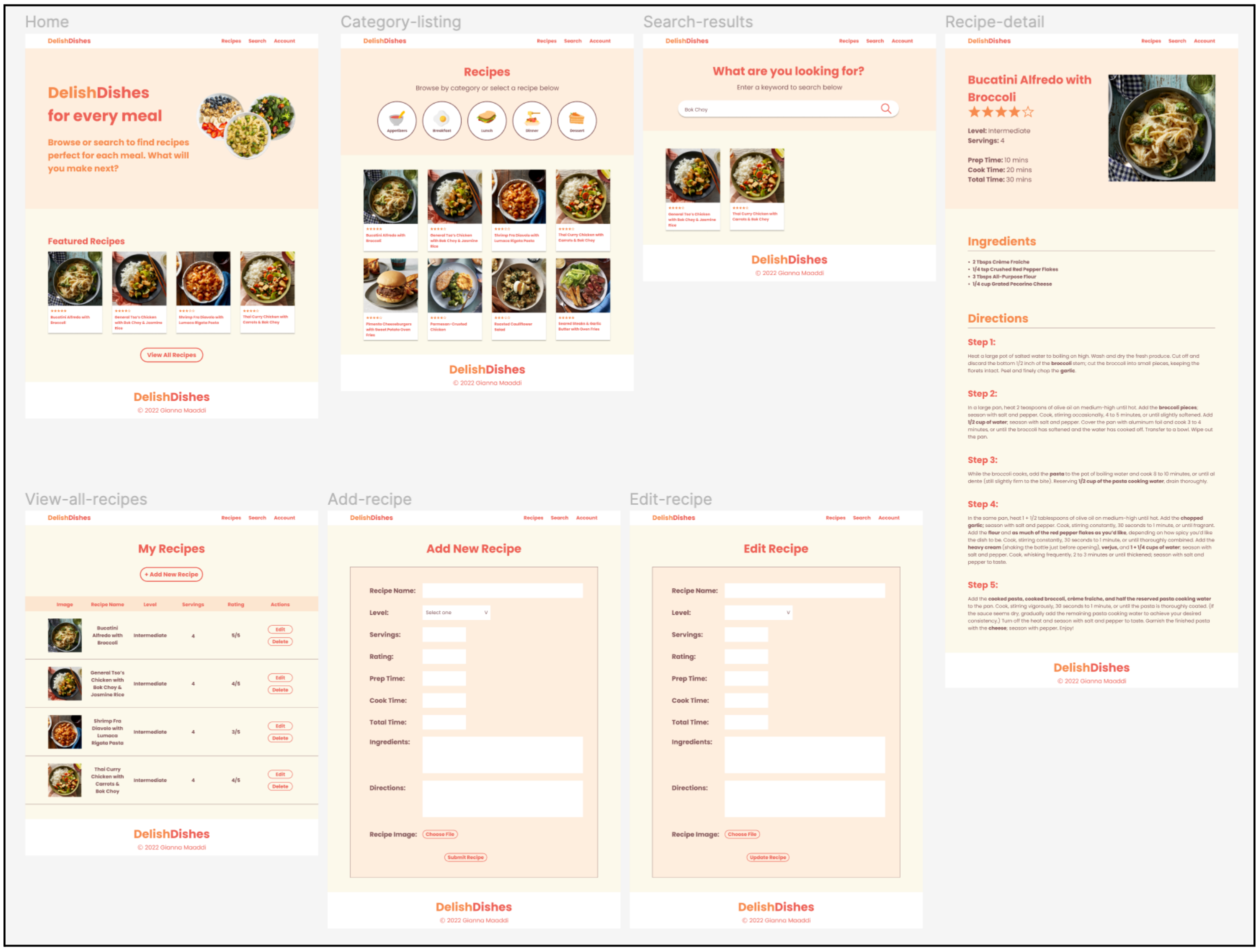
Type Scale

Heading 1	67.34 px / 4.209 rem
Heading 2	50.52 px / 3.158 rem
Heading 3	37.9 px / 2.369 rem
Heading 4	28.43 px / 1.777 rem
Heading 5	21.33 px / 1.333 rem
Paragraph	18 px / 1.125 rem

Style tile for the recipes web application, DelishDishes

Designing the Web Application

The image below shows the initial designs I created for each web page using Figma. I did not want to display too much information on the homepage, but I wanted to showcase some of the recipes, so I created a "Featured Recipes" section. I decided that I wanted to create card components to display each recipe because they would showcase beautiful images of the food and display the recipe titles in a clean, visually appealing manner. When a user clicks on any of these card components, they would be taken to the recipe detail page, which contains additional information about the recipe, including the level of difficulty, servings, prep/cook time, ingredients, and directions. To add a new recipe or edit an existing recipe, users would be taken to a form that they can access from the "My Recipes" page. With forty recipes in my database, I wanted to ensure that the content had a strong organizational structure. I believe that incorporating a table of recipes and rows of card components achieved this goal.



Screenshot of the initial Figma designs for each page

Coding the Web Application: Approach

Once I created the initial designs, I started to write the code for my project. For each page I wanted to create, I built out the pages in HTML first then added CSS to add styling and PHP/MySQL to add functionality. Once I completed a page, I moved on to a different page where I repeated the process: writing the HTML followed by CSS/JavaScript and PHP/MySQL. I decided to create the admin screens first because I wanted to get more familiar with PHP and MySQL before focusing on the client-facing screens that would require more styling. I set up my database and dynamically displayed the recipes from my database on the admin recipes screen seen **here**. Once I got the first piece of PHP functionality to work, it was not as difficult to implement PHP in other areas of the project. I added the edit and delete functionality relatively easily.

Coding the Web Application: Struggles

I ran into an issue at one point with the search page seen **here**. Although I was able to display recipes on the page when a user's search query found a match, I struggled to display a "No results found" message when the search returned no results. Eventually, I realized that I created different PHP variables with the same name, which overrode what I wanted to display. This mistake was a result of working quickly to finish a project without taking the time to write clean code. After realizing this, I decided to take the time to write clean code as I went along because this could save me time when I needed to debug.

Once I finished adding the functionality to my web application, I reviewed what I built, which is when I noticed several other issues ranging in severity. First, even though my selected font, Poppins, was correctly displaying on my web pages, I received several error messages when I inspected the site with Google Chrome's DevTools. I decided to remove the .woff2 files in my CSS document and link to the Poppins font online instead, which resolved this issue.

Also, I realized that my web application did not look as visually appealing on smaller screens. Since I selected somewhat large fonts, these took up a lot of screen real estate. I also noticed that rows of card components were not centered on smaller screens. I resolved these issues in a set of CSS media queries. Although these changes were time-consuming, my previous knowledge of CSS greatly contributed to my success.

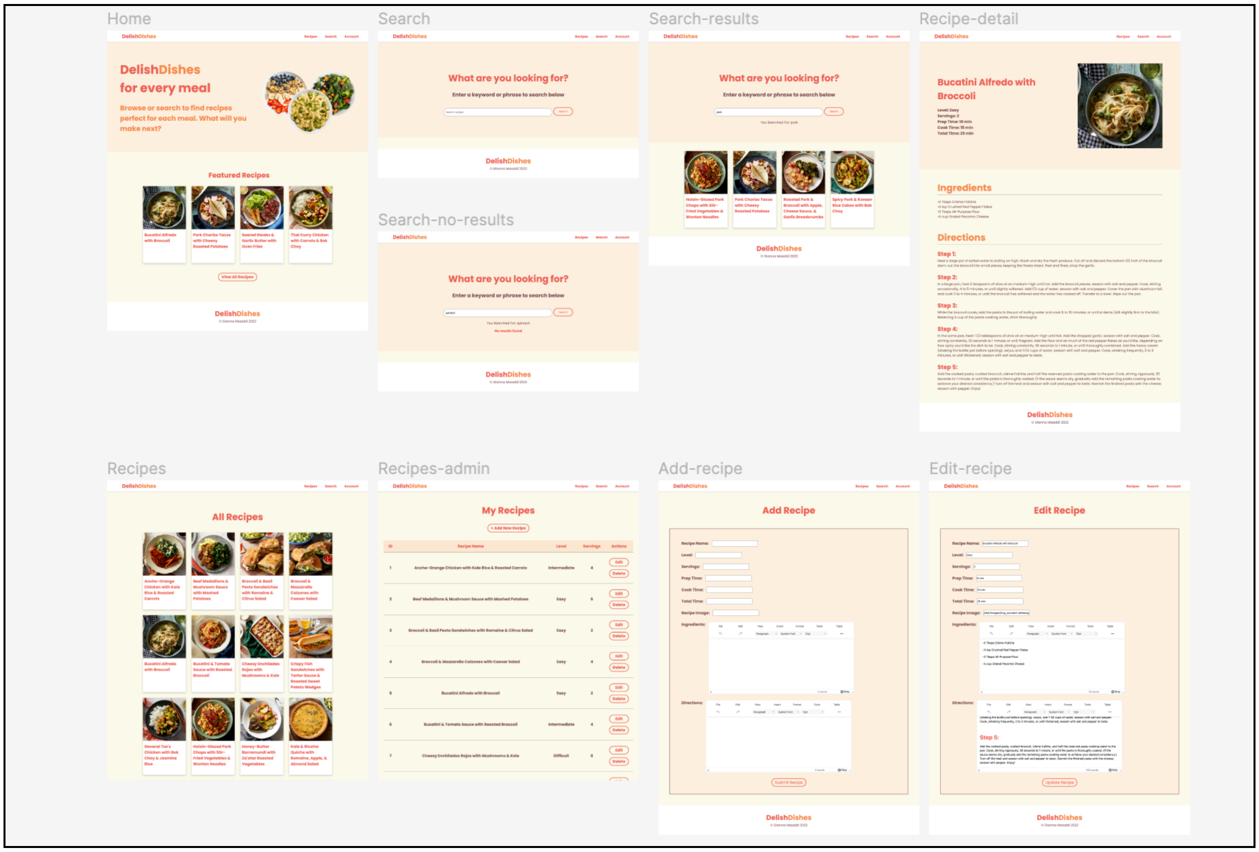
The Solution

Link to the Final Solution

To view the final solution, click **here**.

Changes from the Original Designs

The final designs for my recipes web application are shown below. I stayed true to my original designs for the most part, but I made some changes as I coded my project. For example, the client-facing recipes screen no longer contains recipe categories because I decided that this would be unnecessary when all I wanted was for users to be able to click on a card component that would take them to that recipe's detail page. I also decided to remove the recipe images from the admin recipes page because many recipes had long titles and little space, making each row taller than I would have liked. The final major change was removing the rating system for each recipe. I decided that this was another unnecessary component out of the scope of this project because I would be assigning arbitrary values to forty recipes.



Screenshot of the final designs for each page

Strengths of the Final Design

The forms make it easy for users to add or update their recipes, and the recipe table and card components present the recipes in a clean, organized, visually appealing way.

The Results

The final web application was ultimately a success because I created a visually appealing recipes web application that incorporated PHP and MySQL. Learning these new languages and implementing them brought on a series of challenges, especially with the search page. The final solution was very similar to the initial designs I created in Figma, and I adjusted when I felt that certain ideas were out of scope for this project. In the future, I could try to implement some of these ideas, including a filter for different recipe categories and a star rating system for the recipes. At the end of ten weeks, I built a fully functioning, dynamic, and responsive web application.