**Final Report**

**for**

**Nom Nom Express**

**By**

**Angela Ziesel,**

**Kamila Diyanova,**

**Kento Hyono,**

**Manny Sandhu**

Contents

[1. Project Overview 3](#_Toc148894768)

[2. Project Specifications 3](#_Toc148894769)

[2.1 Function Requirements 3](#_Toc148894770)

[2.2 Non-Function Requirements 3](#_Toc148894771)

[2.3 Group Roles 3](#_Toc148894772)

[3. Usability Guide 4](#_Toc148894773)

[4. Problems and Challenges 8](#_Toc148894774)

[5. Skills Learned 9](#_Toc148894775)

[6. Proposed Changes 9](#_Toc148894776)

[7. Code Snippets 9](#_Toc148894777)

**Table of Figures**

[Figure 1: Login Window 4](#_Toc148894778)

[Figure 2: Registration Page 5](#_Toc148894779)

[Figure 3: Main Page (logged in) 5](#_Toc148894780)

[Figure 4: Account Info Page 6](#_Toc148894781)

[Figure 5: Order Cart Page 7](#_Toc148894782)

[Figure 6: Admin Page 8](#_Toc148894783)

## Project Overview

[assigned to: Kento, talk to Shohreh about this section (how big, detailed etc)]

## Project Specifications

### 2.1 Function Requirements

* R01 – The client must be able to log into an admin account
* R02 – The admin account must be able to add products to the web site
* R03 - Must be able to display products by categories
* R04 - Customers must register to place orders
* Must provide a fully functioning shopping cart utility where a customer can:
  + R05 - display the selected items currently in the cart
  + R06 - add selected products to the cart
  + R07 - delete products (individually or all) from the cart
  + R08 - adjust the quantity of a selected product currently in the cart
  + R11 (CR01) - Create a unique file for each order that contains the order confirmation information so that at a future date, the information in the file can be sent to the client as an email message
  + R12 - include a secure, online payment process
* R13 - Allow customers to view their order history
  + R50 (CR02)
    - A. All users must be shown a new privacy page informing them of this new law, what personal information is being collected on them and how that information will be used.
    - B. Users must either accept the terms or decline them.
    - C. Users must be able to change their mind on accepting or declining the terms later on.
    - D. Users that decline the terms will not be allowed to login to the main site.

### 2.2 Non-Function Requirements

* R09 - preserve the contents of the cart if the user’s session is disconnected accidentally
* R10 - have the cart accessible regardless from any machine the customer is logging in
* R14 - Provide an intuitive, consistent look and feel to the user interface
* R15 - Use free open-source software
* R16 - Have the system fully tested and up and running in 8 weeks (this is a firm deadline)

### 2.3 Group Roles

[assigned to: Kento, talk to Shohreh if needed]

## Usability Guide

1. User accesses URL in browser of their choice [include production address]
   * + - 1. Site prompts user to login

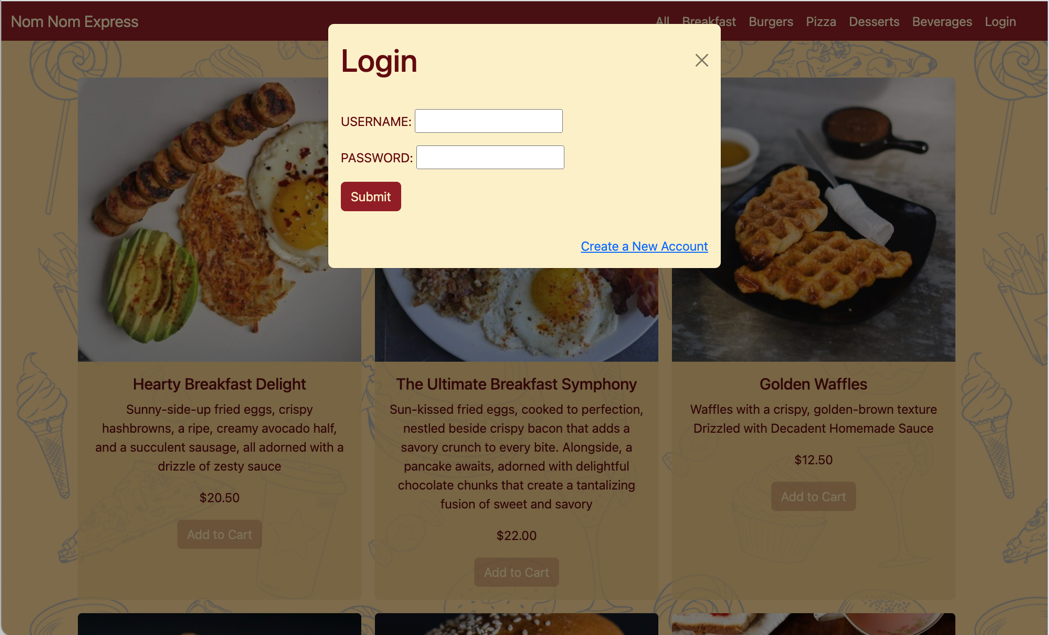


Figure : Login Window

* New User:

1. User selects  link
2. A General Data Protection Regulation window is displayed, User can Accept or Decline the privacy policy
   * User clicks  button
     + 1. Window closes, redirected to main page, prompted to Login
   * User clicks  button
     1. Window closes, redirects to Registration page

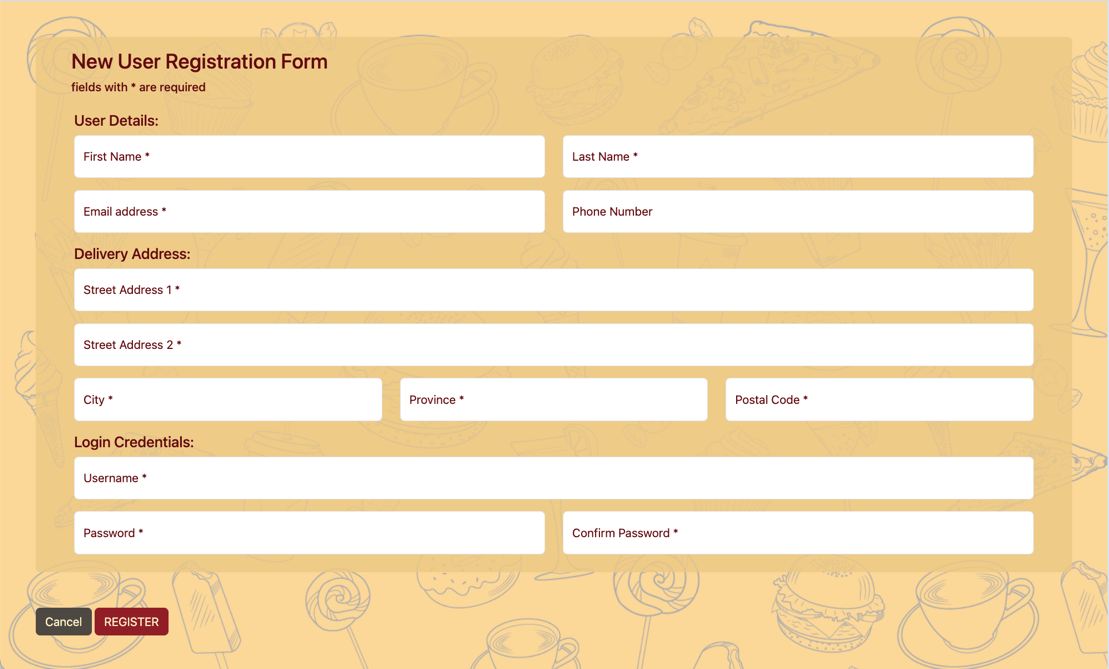


Figure : Registration Page

* + 1. Enter information into appropriate fields, click  button, correct any invalid entries or missing fields
  + Existing User:

1. User enters username and password and clicks  button, correct any invalid entry
2. User is logged in

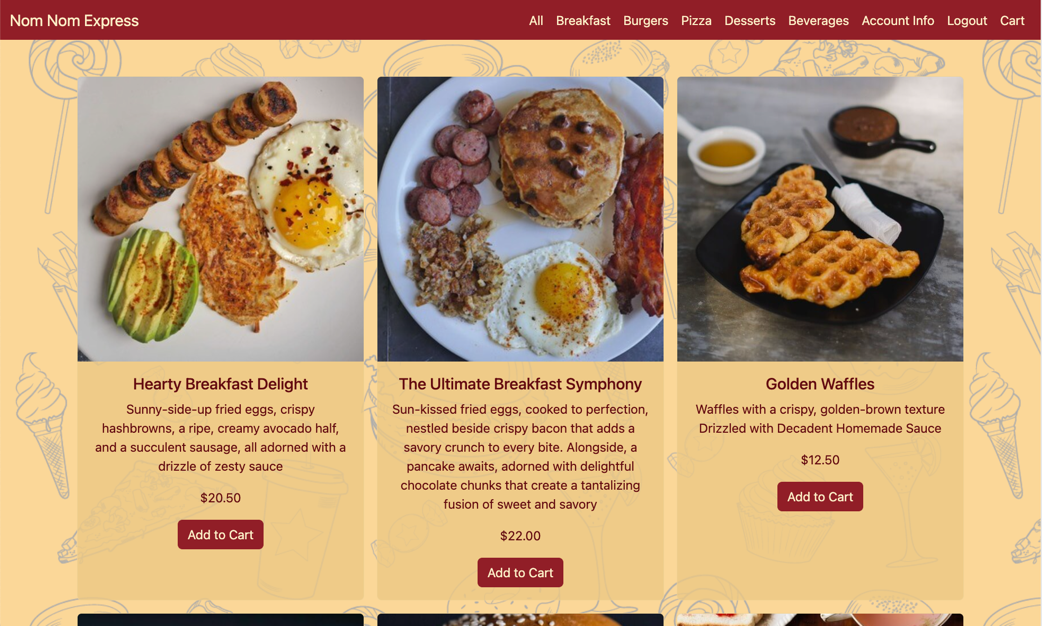


Figure : Main Page (logged in)

* + - * Customer Account:

1. Site redirects to main page, with updated navigation bar and activated Add to Cart buttons
2. Menu items can be filtered by clicking on the category in the navigation bar
3. Account Info

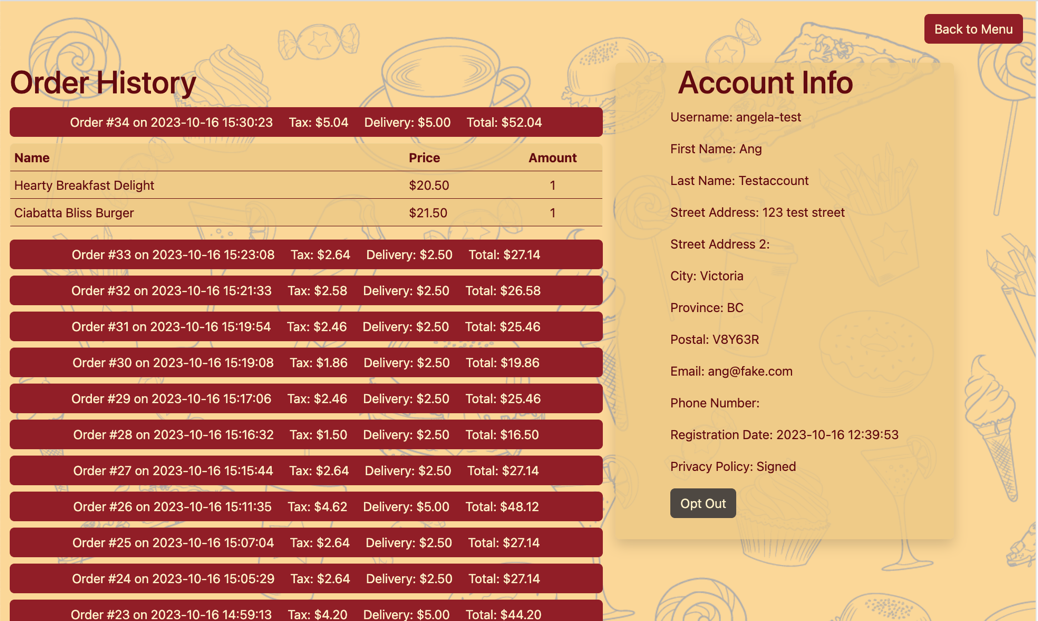


Figure : Account Info Page

* + 1. User can review Account info and history by clicking  link in the navigation bar. Order History is displayed on the left, clicking on an order row will expand the details to display each item ordered. Account Info is displayed on the right, showing all the registration information
    2. Click on  button to decline privacy policy. User will no longer be able to add items to cart

1. Add item to cart
   * 1. Menu items can be added to cart by clicking the  button. If clicked more than once, quantity of item in cart is increased
2. Display cart
   * 1. Cart can be displayed by clicking on  link in navigation bar



Figure : Order Cart Page

1. Update order cart
   * 1. Quantity of item can be adjusted by clicking the  or  buttons
     2. Item can be deleted by clicking on the  button
     3. Cart can be emptied by clicking on the  button, page will update to display message ‘No Items in the Cart’
2. Order can be placed by clicking on the  button
   * 1. Payment window opens, User enters credit card information and clicks Pay button
     2. Site redirects to Order Complete page, displaying Order details and thank you message to User
     3. Site creates an Order Confirmation file in Orders folder simulating an Order Confirmation email
3. User clicks  button to return to main page

* Admin Account:

1. Site redirects to Admin page, displaying Store Info, Add Menu Item form, and list of existing Menu Items that can be edited or enabled/disabled

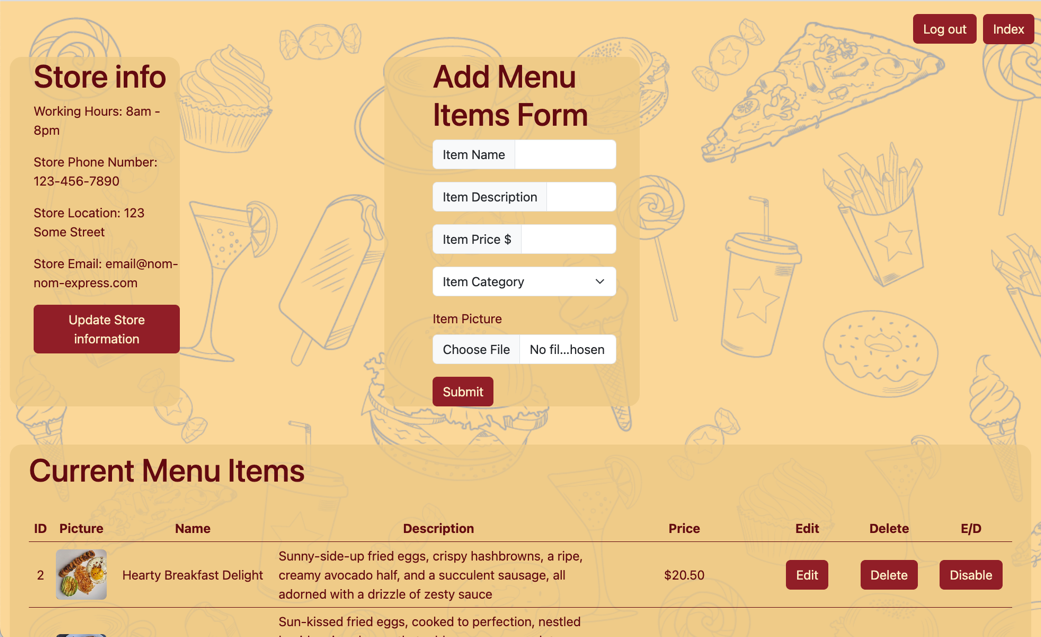


Figure : Admin Page

1. Update Store info
   * 1. Clicking Update Store information button
     2. Window opens with Update Form, User enters new information, clicks  button
     3. Window closes, new information displayed in Store info section
2. Add new Menu item
   * 1. Enter item information into Add Menu Items Form, select picture from file, click  button
     2. Site adds item to database and displays on main page
3. Update Current Menu Items
   * 1. Click  button for a window to modify the data fields for that menu item. Update appropriate field and click  button. Item is updated in database and changes reflect on main page
     2. Clicking  button asks for confirmation before the item is deleted. Click  button to confirm ( button to cancel). Item is deleted from database and removed from main page
     3. Clicking  button asks for confirmation before the item is disabled. Click  button to confirm ( button to cancel). Item remains in database but is not visible/available for order on main page
     4. Clicking  button (on item previously disabled) asks for confirmation before the item is enabled. Click  button to confirm ( button to cancel). Item is visible/available for order on main page.
4. User logs out
   * + - 1. Click  button from navigation bar
         2. Site redirects to main page and prompts for login

## Problems and Challenges

*< Problems and challenges you faced as a group during the project. This should include a description of each problem and how you overcame it>*

[Final write up assigned to: Manny]

Technical issues

1. Database Forward Engineer not working (overcame by writing our own sql)
2. Gitlab (how did we collaborate when gitlab down / collaboration, learnings came late)
3. Consistency (Plan out naming conventions as a team, including filenames and global variable names)

Connection issues

1. Deepblue connection
2. Remote Connection to School labs

Learnings

1. MS Project software learned late
2. Use Case writeups (more detailed, more clear idea of what it should do/look like up front)

## Skills Learned

* 1. Our team developed a deep understanding of PHP as a middleware tool that allows interact with our database. We used PHP to create dynamic and responsive website, ensuring a robust connection between our frontend and the backend database.
  2. We accomplished the seamless integration of Stripe into our project, providing a secure and user-friendly solution for online payment processing.
  3. We developed the skill of creating comprehensive use cases and other project-related documentation. Creating use cases helped us to clearly define and document the functional requirements of our project and ensure that all team members understand what the system should do, thus reducing misunderstandings and miscommunications.
  4. Our team became skilled at using GitLab for collaboration. We learned how to work together efficiently by creating, merging, and managing different versions of our code, making our teamwork smoother.
  5. We got better at understanding and working with code written by different team members, even when they had their unique ways of writing code. This made our code collaboration more effective.
  6. We gained practical experience in managing MySQL, learned how to effectively use and manage MySQL database to store, organize, and retrieve data efficiently, which is crucial for our project's success.
  7. We improved our understanding of all stages a project goes through, better anticipating needs further down, saving time and avoiding rework.

## Proposed Changes

1. Naming Conventions: In future if we work together as a team, we will decide how we name things like files and variables. This way, we'll ensure that the names are consistent and clear for everyone. When everyone knows what things are called, it makes it easier for us to understand and work on the project.
2. Stable Leadership: We'll aim to have more stable roles and leadership within our team. Changing roles every week caused disruptions, so we'll try to avoid that. So, next time, we'll try to have more consistent leaders and roles in our team. This means that the same people will lead and do specific tasks for longer periods, which should make things run more smoothly.
3. In the future, we'll spend more time talking and planning before we start. This helps us understand things better and make sure we're on the right track from the beginning. One important part of this planning is creating use cases. These will help us outline how the system should work and what it should do from a user's point of view.

## Code Snippets

1. Back-end / middleware:

Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание

The peace of code checks if a POST request with an 'action' parameter is received (indicates that a user wants to adjust the quantity of an item in their cart). It fetches the current quantity of the item from the database. Depending on the user's 'action' (increase, decrease, or no change), it calculates the new quantity, then updates the database with the new quantity for that item. If the update is successful, the user is redirected to the cart page, reflecting the updated item quantity, if not - it will display an error message.

Our team successfully implemented dynamic cart management, allowing users to easily adjust the quantity of items in their cart. This enhances the user experience by giving them more control over their shopping cart. This code demonstrates a solid understanding of interacting with the database using MySQL and PHP. We have included an error handling to provide informative error messages in case something goes wrong, which can help us with debugging.

We proud of this code as it effectively manages the shopping cart, enhancing the user experience and demonstrating proficiency in key back-end /middle-ware development skills.

[assigned to: Manny]

1. Middle-ware:

[assigned to: Kento]

1. Front-end: