TECHNICAL SPECIFICATION REPORT

CIS 243: PizzaParlor Application Upgrade

TWO BITS & A BYTE

CLAUDIA DU PLESSIS

PAUL FISCHETTI

BILL SELING

Edmonds Community College
20000 68th Ave W
Lynnwood, WA 98036

Phone (425) 640-1459
Fax: (813) 374-2023
info@edcc.edu

Technical Specification Report

Table of Contents

Technical Specification Report

CIS 243: PizzaParlor Application Upgrade

INTRODUCTION

This document defines the software, architectural design, and test plan for the next iteration of the CIS 243 PizzaParlor web site—upgrading it to an e-commerce ready, database-driven web application, and increased functionality.

Stakeholders

The PizzaParlor web site has a long and storied history, going back several quarters. Originally designed and operated by Marti Baker to bring in a few bucks to maintain low computer lab fees (that's her story and she's sticking to it), it quickly was closed down following several complaints that the virtual Canadian Bacon truly wasn't Canadian and the Hawaiian pizza really didn't come from virtual Hawaii.

The following stakeholders are involved in the current development effort to upgrade the site.

- Tessa Mero, Project Sponsor and Technical Advisor
- Bill Seling, Technical Lead
- Claudia Du Plessis, Lead Developer
- Paul Fischetti, Lead Developer
- Aron T Norberg, IT Systems Support
- Mohammad Thalib, IT Systems Support
- EdCC Student Body (CIS 243 class), Consumers
- Marti Baker, Technical Advisor

PURPOSE AND SCOPE STATEMENT

The scope of this project is to upgrade the CIS 242 PizzaParlor web site from a self-contained form-based application to a database-driven web application, adding increased functionality, including the ability to send notification emails to customers and staff, and the addition of an administration section (including basic reporting on transactions), to be implemented by June 17, 2014.

In Scope

- Creation of MySQL database, tables, queries, and any stored procedures to support new functionality
- Summary page: Write Customer and Order information to database
- Send confirming emails from Summary page
  - Confirming email to customer
  - Email to PizzaParlor that notifies them to make pizza
- Add Confirmation page (between the Order and Summary pages) with appropriate coding to allow customer opportunity to change Order or Delivery Information
- Creation of Administration pages and reports
  - Create Administration Login page

- Create Administrative Menu Options page
    - Incomplete Orders (report) – (Outstanding customer orders?)
    - Complete Orders (report)
    - Daily Summary (report)
    - Customer Listing (report)
    - Add User
    - User Status (Admins) – Activate/inactivate Admins
- Create page and coding for creating Admins; first time Admins must change password at first login
- Create two reports (to be displayed in web page)
    - Outstanding Customer Orders
    - Customer Listing

Out of Scope

- System upgrades to server(s)
- User training beyond previewing functionality at walkthrough
- Graphics changes
- UI redesign beyond changes noted as In Scope
- Any item not specified as In Scope

PROJECT PLAN

The project will be completed in five phases and ready for implementation by June 17, 2014.

- Phase I: Complete *Technical Specification Report*
- Phase 2: Complete site mockup, including new page and admin section
- Phase 3: Complete database design, including ability to send emails
- Phase 4: Complete integration; manual functional/non-function, static; and user acceptance testing
- Phase 5: Successfully implement site

TECHNICAL SPECIFICATIONS

Several “moving” parts require coding and integration to complete the Pizza Parlor upgrade successfully, including coding web pages for a variety of web browsers, server-side coding, and database scripts and queries.

Web Languages

Web pages will be updated to HTML5 (from XHTML 1.0 Transitional), and incorporate JavaScript (and possibly some jQuery code as needed) to ensure transactions are accurately captured and sent to the database, including client-side validation. The basic theme of the current user interface will remain unchanged and new pages will adopt the same theme. CSS3.0 will be used to enforce page styles.

PHP will be used server-side to handle delivering the page form data to the database, after performing server-side validation.

RDBMS Selection

MySQL v5.6.12 (minimum) has been selected as the RDBMS for development, unit testing, and integration testing. The server is part of the WAMP suite.

Hardware and Software Requirements

Development and unit testing will occur on Windows 7 equipped PCs running the WAMP server which includes Apache v2.4.4 (Win32) and PHP v5.4.16.

Development PCs are expected to run Windows7 Professional with Service Pack 1 and be current with all patches, service packs, and updates.

FUNCTIONAL SPECIFICATIONS

The functional specifications section describes what the site must do and what data is required for it to be successful.

Functional Requirements – Pizza Orders

The following functional requirements are required for the pizza ordering functions of the PizzaParlor upgrade:

- The customer must be able to open the Order page
- The customer must be able to specify pizza size (Order page)
- The customer must be able to specify crust type (Order page)
- The customer must be to select Build Your Own or a Specialty Pizza (Order page)
  - If Build Your Own customer must be able to select toppings:
    - Pineapple
    - Canadian Bacon
    - Pepperoni
    - Chicken
    - Sausage
    - Tomatoes
  - If Specialty Pizza, customer must be able to select:
    - Hawaiian
    - Veggie
    - Meat Lovers
- The customer must be able to specify Delivery Information (Order page)
  - First Name (required)
  - Last Name (required)
  - Address (required)
  - Apartment (optional)
  - City (required)
  - State (required)
  - Zip (required)
  - Phone (required)
- The customer must be able to submit pizza size, crust type, build-your-own or specialty pizza, and delivery information to a Confirmation page.
- Confirmation Page must allow customer to change order or delivery information (Confirmation page)
- Customer must be able to submit Confirmation page data to Summary Page
- The Summary page must list the order, delivery information, and total price of order.
- Emails with order information must be sent to customer and PizzaParlor.

Functional Requirements – PizzaParlor Administration

The following functional requirements are required for the administrative functions of the PizzaParlor upgrade:

- Admin must be able to login into admin portion of site with login/pw combo
- With only one Admin user in Admin login table, the /admin/adminFirstLogin.php displays. All fields must be complete to proceed beyond this page.
- Admin Menu page displays after creating new Admin user.
  - Menu1
- Admins may run reports on sales, customers, and Admin users
  - Sales
    - Incomplete Orders for today
    - Completed Orders for today
    - Order Summary for today
  - Customers
    - Customer Listing
  - Users (Admin for now; could be other types later)
    - Active Admin Users
    - Inactive Admin Users
- Admins must be able to add other admins
  - New user form displays (All fields required)
  - Confirmation form displays

Information Requirements

Information required for the PizzaParlor update can be thought of as belonging to specific groups, e.g. store info, pizzas (pizza options and prices), customers, orders, and administration.

PizzaParlor_Info

- PizzaParlorID (in case we get more than one location)
- Company_Name
- Customer Address
- Customer P.O. Box
- Customer City
- Customer State
- Customer Zip
- Customer Phone
- AppVersionID

Pizza

- Pizza_Size: (12”, 16”, 20”, Price)
- Pizza_Crust (Hand-tossed, Deep dish)
- Pizza_Toppings: (Pineapple, Canadian Bacon, Pepperoni, Chicken, Sausage, Tomatoes, Price)
- Specialty_Pizza (Hawaiian, Veggie, Meat Lovers, Price)

Customer

- CustomerID
- Customer First Name
- Customer Last Name
- Customer Address
- Customer Apt

- Customer City
- Customer State
- Customer Zip
- Customer Phone

Orders

- Order_ID (can be primary key or make one up)
- Order_Date
- Order_Time
- Order_Subtotal (computed from pizza options)
- Order_Tax (8.5% or whatever)
- Order_Total (computed from subtotal & tax)
- Order_Complete (Y/N)
- Emails_Sent (Y/N)

Administration

- AdminID
- Admin_FName
- Admin_LName
- Admin_Username
- Admin_PW
- Admin_Active

Site Plan

The site plan, sometimes referred to as a sitemap, is a graphical representation of the PizzaParlor application, including the new administrative functions. See “Appendix C” for the upgraded sitemap.

Wireframes

Wireframes for the upgraded PizzaParlor help the reader of this document visualize the current pages and the pages to be added. Please understand that wireframes sometimes change after construction begins and the developers start trying to fit real elements and graphics to the pages. The wireframe are included in “Appendix D” of this document.

File Structure

See “Appendix E” for a file structure diagram. Currently, the files include:

/ (PizzaParlor root folder)
/css folder (pizzaStyle.css)
/graphics folder (index.jpeg, pizza.jpeg, pizzaSlice.jpg, productsmain.jpg, spacer.PNG)
/js folder (cookies.js, test.js)
/order.php
/confirmation.php
/summary.php

/admin (root admin folder off PizzaParlor root folder)
/admin/index.php
/admin/addAdmin.php
/admin/addAdminConfirmation.php
/admin/adminFirstLogin.php
/admin/adminMenu.php
/admin/reportIncompleteOrders.php

/admin/reportCompletedOrders.php
/admin/reportOrderSummary.php
/admin/reportCustomerListing.php
/admin/userStatusAdmin.php

TEST PLAN

Our testing focuses prominently on ensuring the PizzaParlor web application meets the minimum requirements as interpreted by our team. Extra functionality beyond the requirements expresses the team's enthusiasm for the project and it is hoped that should such extra functionality fall short of expectations, it will be viewed in a positive light.

Test Cases and Scenarios

Manual testing techniques (including elements of static, functional, and non-functional testing) will be used to execute use/test case scenarios against the application. Testing will focus primarily on Happy Path, but will also include some exception testing. See "Appendix B" for use/test case scenarios.

Bug Reports

Bugs will be documented using screenshots and text in Word documents and stored in our team folder on Canvas Bugs folder. The Test Manager will establish the status, priority and severity levels for bugs after reviewing the issue. See "Appendix F" for bug report requirements and a sample bug report template.

Testing Roles and Responsibilities

The following testing roles and responsibilities are in effect for the rest of Spring Quart 2014.

Name	Role/Responsibility	Email
Bill Seling	Test Manager	w.seling9619@edmail.edcc.edu
Paul Fischetti	Unit Test	p.fischetti1959@edmail.edcc.edu
Claudia Du Plessis	Unit Test	c. duplessisxxxx@edmail.edcc.edu
Tessa Mero	Program Manger	tessa.mero@email.edcc.edu

Test Reports

Test reporting will not be conducted on a formal basis. The Test Manager will share bug results with the team and Program Manager as needed.

Training

Training on the PizzaParlor application, for the purposes of testing, will not be required.

Hardware

Testing will conducted primarily on personal computers running Microsoft Windows 7 and XP operating systems. A limited amount of testing will be conducted using an Apple iMac computer. Mobile and other handheld devices will not be tested for this iteration.

Software
The most current version of Internet Explorer, Sari, Chrome, and Firefox will be used to manually test the PizzaParlor application. Microsoft and Apple operating systems will be running the browsers.

Data
No formal test data is required to test this version of the application forms and database tables. Regular text strings, including special characters for exception testing, will be used for testing.

Tools
Testing will be manual (including elements of static, functional, and non-functional testing). No tools will be required to test this iteration of the app. However, if time permits, the team is interested in learning about any tools that would help automate testing.

Test Schedule
The following test schedule will be in force for testing this iteration. Also see “Appendix A” for the Project Schedule.

Testing Phases and Tentative Dates	
Phase	**Tentative Dates**
Test Plan	Due **05/20/2014**.
Test Cases	Due **05/20/2014**.
Smoke Testing	Due **06/10/2014** – Smoke testing will be considered complete when the PizzaParlor application is installed, configured and pronounced ready for testing (the home page displays) on the Test Manager’s computer.
Installation, Functional and Non-functional Testing	Scheduled for **06/10/2014**. Installation testing will only be performed to the extent that ensures that the solution is ready for functional and non- functional testing.
User Acceptance Testing (UAT)	UAT is tentatively scheduled following the end of Functional and Non-functional Testing, which is scheduled for **06/17/2014.**
Regression Testing	Scheduled as needed as bug fixes occurs prior to user acceptance.
Go-Live	Tentatively scheduled for **06/17/2014**.

Approach to Testing
Manual testing (including elements of static, functional, and non-functional testing) will be used to test the PizzaParlor application within the given time limits. Most testing will focus on validating the operation and accuracy of the application as it relates to the requirements. There will be informal testing with regards to validation, security and performance testing.

Regression testing will be performed whenever a bug fix required to successfully implement the product has been issued during the testing phase. The scope of the regression testing will be determined at time of regression testing.

Bug Handling

Any defects, deficiencies, issues, or bugs will all be reported as bugs. According to *MSF for CMMI Process Improvement 4.2* (download at http://www.microsoft.com/en-us/download/details.aspx?id=24900)**,** a bug is a work item for a potential problem with the product and should accurately describe the problem so that the reader understands the full impact of the bug. The bug write-up should list the steps for reproducing the bug. All bug write-ups for PizzaParlor will strive to meet this standard.

All bugs will be tracked, including spelling errors; any perceived usability deficiencies, etc. and stored on Canvas in the team's Bug folder. The Test Manager will set initial bug status values wherever possible. The Program Manager will have ultimate authority over bug status, priority and severity level settings.

Setting Bug States

A bug can be in one of four states. A bug's state changes according to the work to be performed and does not necessarily follow a liner progression from beginning to end.

Setting Bug States	
Status	**Description**
Proposed	A proposed status means a bug is waiting for approval from the project's Test or Program Managers. Proposed bugs are triaged and either approved to become Active, or Closed. A Proposed bug is closed if it is Deferred, Rejected, or found to be a Duplicate of another bug.
Active	When a bug is set to Active, it indicates that the project's Test or Program Managers believe that a problem exists and some sort of resolution must be achieved.
Resolved	A bug is set to Resolved when the issue has been addressed by the person responsible for the fix. A bug resolution must be verified by the person who initially authored the bug (usually a tester or Test Manager). If the resolution is correct, the bug is closed, otherwise it is reactivated so that work towards a fix can continue.
Closed	When a bug is set to Closed, it means that no further work is to be done for the current product version. A bug is set to Closed after the resolution has been verified. A closed bug may be reset to an Active status if regression testing reveals a reoccurrence of the issue.

Setting Bug Priority Levels

Setting a bug's priority is a subjective take on the bug's importance rating. Valid values are High, Medium, and Low.

Setting Bug Priority Levels		
Ranking	**Priority**	**Description**
1	**High**	The bug must be fixed as soon as possible. The bug may be blocking further progress in this area; the area may be critical to the business unit. Any bug that could prevent the group from being able to do its work fits this state.
2	**Medium**	The bug should be fixed soon, probably before the go-live date. This area is important to the business unit. Any bugs that may affect the ability of the group to do its work fit into this state.
3	**Low**	The bug should be fixed if time allows. It may be somewhat trivial. May be postponed. The business unit could get by with a bug in this area or use a workaround until the bug could be fixed.

Setting Bug Severity Levels

Setting a bug's Severity Level is a subjective take on the impact of not fixing the bug. Valid values are Critical, High, Medium, and Low.

Setting Bug Severity Levels		
Ranking	**Name**	**Description**
1	**Critical**	This may be a "show-stopper" that requires immediate attention. A critical bug may need to be resolved for testing to continue. This could also be a defect that impacts production.
2	**High**	This bug could potentially represent a major impact to the customer. Functionality or a system component might be broken. This would most likely need to be resolved quickly.
3	**Medium**	This bug could potentially have a significant impact on the customer. Functionality or a system component might not be working properly. This would most likely need to be resolved before release of the current version or a patch might need to be developed.
4	**Low**	This bug might have a minor impact on the customer. Functionality or a system component might impose some loss of functionality, but there may be an acceptable and easily reproducible workaround. This should be fixed if there is time, but it can be deferred.

APPENDIX A – PROJECT SCHEDULE

The following schedule is dependent upon availability of resources (students' time). Changes to course schedule, sharing of scripts, etc. may cause schedule to change.

Deliverable	Due Date	Responsible
Technical Specification Report	May 20, 2014	Bill Seling
Site Mockup	May 27, 2014	Claudia Du Plessis
Confirmation Page	May 27, 2014	Paul Fischetti
Administration Page	June 03, 2014	Claudia Du Plessis
Database Design	June 03, 2014	Paul Fischetti
Emails	June 03, 2014	Claudia Du Plessis
Testing	June 10, 2014	Bill Seling
Implementation	June 17, 2014	Claudia Du Plessis, Paul Fischetti, Bill Seling

APPENDIX B – USE CASE SCENARIOS

The following use case scenarios are designed to test the functionality of the PizzaParlor application using manual testing techniques (including elements of static, functional, and non-functional testing). Most testing will focus on performing Happy Path testing (assuming we can go through the site normally). As time allows, exception testing will be performed where abnormal values and unusual actions are taken. This phase may also include Exploratory-testing techniques (off-script) to discover additional bugs.

/order.php Use Case Scenarios (Happy Path)

Use Case	Scenario	Result w/Comment
OP001	Check all text on page for correct spelling and grammar and consistency. **Expected Result:** Grammar and spelling are correct. Font styles are consistent and appropriate.	
OP002	Verify all links on page are active and display expected page in the expected window type. **Expected result:** All links function correctly and pages display as expected.	
OP003	Select **pizza size** 12, 16, or 20 (radio button). **Expected result:** button is correctly selected and size appears to right of form under Order Summary section and order totals appear normal.	
OP004	Select **crust type**: Hand-tossed or Deep dish (radio button). **Expected result:** button is correctly selected and crust type appears to right of form under Order Summary section and order totals appear normal.	
OP005	Pass 1: Select **Build your own** (toppings). Select each check box. **Expected result:** Each topping appears to right of form under Order Summary section and order totals appear normal. Pass 2: Select specialty pizza (radio button). **Expected result:** button is correctly selected and pizza appears to right of form under Order Summary section and order totals appear normal.	
OP006	Complete Delivery Information section. Complete each field with normal values. **Expected result:** Fields accept values.	

Use Case	Scenario	Result w/Comment
OP007	Click **Order your Pizza**. **Expected result:** Confirmation page displays.	
OP008	Reload page and execute steps OP001 – 0P002. At OP003, perform Pass 2 test for specialty pizza. Continue with tests OP004 and OP006. **Expected results:** Confirmation page displays.	
OP009	Review information on Confirmation page includes all information from Order page. **Expected result:** All information from Order page displays on Confirmation page.	
OP010	Check database tables to make sure data has been written is expected formats. **Expected result:** All tables have successfully updated and data is formatted correctly.	

/order.php Use Case Scenarios (Exception)

Use Case	Scenario	Result w/Comment
OP001	Check all text on page for correct spelling and grammar and consistency. **Expected Result:** Grammar and spelling are correct. Font styles are consistent and appropriate.	
OP002	Verify all links on page are active and display expected page in the expected window type. **Expected result:** All links function correctly and pages display as expected.	
OP003	Do not select any pizza options and click submit button. **Expected result:** Message displays asking user to select pizza size (first set of radio buttons on form).	
OP004	Select pizza size, but no other options and click submit button. **Expected result:** Message displays asking user to select crust type (second set of radio buttons on form).	
OP005	Select pizza size and crust type, but no other options and click submit button. **Expected result:** Message displays asking user to select build-your-own or specialty pizza.	
OP006	Select pizza size and crust type, select build-your-own or specialty pizza, and then click submit button. **Expected result:** Message displays asking user to complete Delivery Information.	
OP007	With pizza order correctly filled-out, type a First Name, but leave other fields empty, and then click **Order your pizza**. **Expected result:** user is prompted for other required Delivery Information fields.	
	With pizza order correctly filled-out, type a First Name and Last Name, but leave other fields empty, and then click **Order your pizza**. **Expected result:** user is prompted for other required Delivery Information fields.	

Use Case	Scenario	Result w/Comment
	With pizza order correctly filled-out, type a First Name, Last Name, and Address, but leave other fields empty, and then click **Order your pizza**. **Expected result:** user is prompted for other required Delivery Information fields.	
	With pizza order correctly filled-out, type a First Name, Last Name, Address, and City, but leave other fields empty, and then click **Order your pizza**. **Expected result:** user is prompted for other required Delivery Information fields.	
	With pizza order correctly filled-out, type a First Name, Last Name, Address, City, and State, but leave other fields empty, and then click **Order your pizza**. **Expected result:** user is prompted for other required Delivery Information fields.	
	With pizza order correctly filled-out, type a First Name, Last Name, Address, City, State, Zip, but leave Phone field empty, and then click **Order your pizza**. **Expected result:** user is prompted for Phone Number.	
	Select all possible order options, complete delivery info, and then click Clear. **Expected result:** All fields are reset to starting values (empty).	
	Check DB tables for incorrect data at each: **Expected result:** Data has been written in its expected format.	

/confirmation.php Use Case Scenarios (Happy Path)

Use Case	Scenario	Result w/Comment
CO001	Check all text on page for correct spelling and grammar and consistency. **Expected Result:** Grammar and spelling are correct. Font styles are consistent and appropriate.	
CO002	Verify all links on page are active and display expected page in the expected window type. **Expected result:** All links function correctly and pages display as expected.	
CO003	Verify order details passed from Order page. **Expected result:** Order details match.	
CO004	Verify delivery information passed from Order page. **Expected result:** Delivery information matches.	
CO005	Click button (or link) that allows customer to change Order or Delivery Information. **Expected result:** Customer is returned to Order Page, all data previously entered is present, and customer can change data and resubmit. Any changes are reflected on the Confirmation page.	
CO006	Click button (or link) to submit order and delivery information. **Expected result:** Summary page displays. Order and Delivery Information match Order and Confirmation pages. Emails sent to customer and PizzaParlor staff.	
	Check database tables to make sure data has not been written yet and emails have not been sent. **Expected result:** Data is written and emails sent at orderSummary.php.	

There are no exceptions to test for the /confirmation.php page.

/orderSummary.php Use Case Scenarios (Happy Path)

Use Case	Scenario	Result w/Comment
OS001	Check all text on page for correct spelling and grammar and consistency. **Expected Result:** Grammar and spelling are correct. Font styles are consistent and appropriate.	
	Verify all links on page are active and display expected page in the expected window type. **Expected result:** All links function correctly and pages display as expected.	
OS002	Verify countdown timer displays correct time (if present). **Expected result:** Correct delivery time is displayed.	
OS003	Check database tables to make sure data has been written is expected formats. **Expected result:** All tables have successfully updated and data is formatted correctly.	
OS004	Verify all links on page are active and display expected page in the expected window type. Expected result: All links function correctly and pages display as expected.	

There are no exceptions to test for the /orderSummary.php page.

/emailCustomer.php Customer Email Use Case Scenarios

Use Case	Scenario	Result w/Comment
SE001	Verify PizzaParlor receives email. **Expected result**: PizzaParlor receives email.	
SE002	Check all email text for correct spelling, grammar, consistency, display, etc. **Expected Result:** Grammar and spelling are correct. Font styles are consistent and appropriate. Sentences and paragraphs break appropriately.	
SE003	Verify any links in email display expected page. **Expected result:** Any links function correctly and pages display as expected.	
SE004	Verify Customer To: email address is correct. **Expected result:** To: email address matches value from Order form.	
SE005	Verify From line is correct. **Expected result:** email From line matches value from design.	
SE006	Verify email Customer Subject line is correct. **Expected result:** email Subject line matches value from design.	
SE007	Verify email Body is correct. **Expected result:** email body matches design.	
SE008	Check database table to make sure “email sent” bit is enables. **Expected result:** “Email sent” bit correctly enabled.	

There are no exceptions to test for the /emailCustomer.php page.

/emailPizzaParlor.php PizzaParlor Email Use Case Scenarios

Use Case	Scenario	Result w/Comment
SE001	Verify PizzaParlor receives email. **Expected result**: PizzaParlor receives email.	
SE002	Check all email text for correct spelling, grammar, consistency, display, etc. **Expected Result:** Grammar and spelling are correct. Font styles are consistent and appropriate. Sentences and paragraphs break appropriately.	
SE003	Verify any links in email display expected page. **Expected result:** Any links function correctly and pages display as expected.	
SE004	Verify Customer To: email address is correct. **Expected result:** To: email address matches value from Order form.	
SE005	Verify From line is correct. **Expected result:** email From line matches value from design.	
SE006	Verify email Customer Subject line is correct. **Expected result:** email Subject line matches value from design.	
SE007	Verify email Body is correct. **Expected result:** email body matches design.	
SE008	Check database table to make sure “email sent” bit is enables. **Expected result:** “Email sent” bit correctly enabled.	

There are no exceptions to test for the /emailPizzaParlor.php page.

/admin/index.php Use Case Scenarios (Happy Path)

Starting condition: one user in Admin login table.

Use Case	Scenario	Result w/Comment
ADL001	Check all text on page for correct spelling and grammar and consistency. **Expected Result:** Grammar and spelling are correct. Font styles are consistent and appropriate.	
ADL002	Verify all links on page are active and display expected page in the expected window type. **Expected result:** All links function correctly and pages display as expected.	
ADL003	Call Admin login form. Enter user name of **Admin** and Admin password. **Expected result:** Login form displays and fields allow text entry.	
ADL004	Click admin page **Log in** button. **Expected result for FIRST ADMIN LOGIN:** adminFirstLogin page displays. This form asks you to create a new administrator.	
ADL005	Complete all adminFirstLogin fields. **Expected result:** Fields allow text entry.	
ADL006	Click **adminFirstLogin** Log in button. **Expected result:** Login form displays. Check database table and verify creation of new user.	
ADL007	At Admin log in form, enter newly created user name of **Admin** and Admin password. **Expected result:** Login form displays and fields allow text entry.	
ADL008	Click admin page **Log in** button. **Expected result for newly created Admin:** adminMenu page displays (with links to various options).	
ADL009		

/admin/index.php Use Case Scenarios (Exception)

Use Case	Scenario	Result w/Comment
ADLE001	Check all text on page for correct spelling and grammar and consistency. **Expected Result:** Grammar and spelling are correct. Font styles are consistent and appropriate.	
ADLE002	Verify all links on page are active and display expected page in the expected window type. **Expected result:** All links function correctly and pages display as expected.	
ADLE003	Without typing any values for username or password, click **Log in**. **Expected result:** Message displays asking user to enter a valid username and password.	
ADLE004	Type valid username, but leave password blank and click **Log in**. **Expected result:** Message displays asking user to enter a valid username and password. (*Do not offer up any hints by saying just enter a valid password; would indicate username was valid to unauthorized user.*)	
ADLE005	Type valid username, but enter invalid password and click **Log in**. **Expected result:** Message displays asking user to enter a valid username and password. (*Do not offer up any hints by saying just enter a valid password; would indicate username was valid to unauthorized user.*)	
ADLE006	Type an invalid username, but enter a valid password and click **Log in**. **Expected result:** Message displays asking user to enter a valid username and password. (*Do not offer up any hints by saying just enter a valid password; would indicate username was valid to unauthorized user.*)	
ADLE007	Type a valid username and password, and then click **Log in**. **Expected result:** If there is only one Admin in login table, the /admin/adminFirstLogin.php page appears. Otherwise, the /admin/admin Menu.php page displays.	
ADLE008		

/admin/adminFirstLogin.php Use Case Scenarios (Happy Path)

Use Case	Scenario	Result w/Comment
ADFL001	Check all text on page for correct spelling and grammar and consistency. **Expected Result:** Grammar and spelling are correct. Font styles are consistent and appropriate.	
ADFL002	Verify all links on page are active and display expected page in the expected window type. **Expected result:** All links function correctly and pages display as expected.	
ADFL003	Call Admin login form. Enter user name of **Admin** and Admin password. **Expected result:** Login form displays and fields allow text entry.	
ADFL004	Click admin page **Log in** button. **Expected result for FIRST ADMIN LOGIN:** adminFirstLogin page displays. This form asks you to create a new administrator.	
ADFL005	Complete all adminFirstLogin fields. **Expected result:** Fields allow text entry.	
ADFL006	Click **adminFirstLogin** Log in button. **Expected result:** Login form displays. Check database table and verify creation of new user.	
ADFL007	At Admin log in form, enter newly created user name of **Admin** and Admin password. **Expected result:** Login form displays and fields allow text entry.	
ADFL008	Click admin page **Log in** button. **Expected result for newly created Admin:** adminMenu page displays (with links to various options).	
ADFL009	Check database tables to make sure data has been written is expected formats. **Expected result:** All tables have successfully updated and data is formatted correctly.	

/admin/adminFirstLogin.php Use Case Scenarios (Exception)

Use Case	Scenario	Result w/Comment
ADFLE001	Check all text on page for correct spelling and grammar and consistency. **Expected Result:** Grammar and spelling are correct. Font styles are consistent and appropriate.	
ADFLE002	Verify all links on page are active and display expected page in the expected window type. **Expected result:** All links function correctly and pages display as expected.	
ADFLE003	Without filling in any fields, click **Log in**. **Expected Result:** Message displays that all required fields must be completed or equivalent.	
ADFLE004	Type in an Administrator First Name, but leave all other fields blank, and then click **Log in**. **Expected Result:** Message displays that all required fields must be completed or equivalent.	
ADFLE005	Type in an Administrator First Name and Last Name, but leave all other fields blank, and then click **Log in**. **Expected Result:** Message displays that all required fields must be completed or equivalent.	
ADFLE006	Type in an Administrator First Name, Last Name, and User Name, but leave both password fields blank, and then click **Log in**. **Expected Result:** Message displays that all required fields must be completed or equivalent.	
ADFLE007	Type in an Administrator First Name, Last Name, and User Name, and New Password field, but leave Confirm Password field blank, and then click **Log in**. **Expected Result:** Message displays that passwords must match.	
ADFLE008	Type in an Administrator First Name, Last Name, and User Name, and New Password field, and type a non-matching value in the Confirm Password field, and then click **Log in**. **Expected Result:** Message displays that passwords must match.	

/admin/adminMenu.php Use Case Scenarios (Happy Path)

Use Case	Scenario	Result w/Comment
ADM001	Check all text on page for correct spelling and grammar and consistency. **Expected Result:** Grammar and spelling are correct. Font styles are consistent and appropriate.	
ADM002	Verify all links on page are active and display expected page in the expected window type. **Expected result:** All links function correctly and pages display as expected.	

There are no exceptions to test for the /admin/adminMenu.php page.

/admin/reportIncompleteOrders.php Use Case Scenarios (Happy Path)

Use Case	Scenario	Result w/Comment
AD001	Check all text on page for correct spelling and grammar and consistency. **Expected Result:** Grammar and spelling are correct. Font styles are consistent and appropriate.	
AD002	Verify all links on page are active and display expected page in the expected window type. **Expected result:** All links function correctly and pages display as expected.	
AD003	Page results are correct	
AD004	Click order complete. Expected result: Order moves from this page to Completed orders page.	
AD005	Check database tables to make sure data has been written is expected formats. **Expected result:** All tables have successfully updated and data is formatted correctly.	
AD006		
AD007		

There are no exceptions to test for the /admin/reportIncompleteOrders.php page.

/admin/reportCompletedOrders.php Use Case Scenarios (Happy Path)

Use Case	Scenario	Result w/Comment
ADCO001	Check all text on page for correct spelling and grammar and consistency. **Expected Result:** Grammar and spelling are correct. Font styles are consistent and appropriate.	
ADCO002	Verify all links on page are active and display expected page in the expected window type. **Expected result:** All links function correctly and pages display as expected.	
ADCO003	Page results are correct	

There are no exceptions to test for the /admin/reportCompletedOrders.php page.

/admin/reportOrderSummary.php Use Case Scenarios (Happy Path)

Use Case	Scenario	Result w/Comment
ADOS001	Check all text on page for correct spelling and grammar and consistency. **Expected Result:** Grammar and spelling are correct. Font styles are consistent and appropriate.	
ADOS002	Verify all links on page are active and display expected page in the expected window type. **Expected result:** All links function correctly and pages display as expected.	
ADOS003	Page results are correct	

There are no exceptions to test for the /admin/reportOrderSummary.php page.

/admin/reportCustomerListing.php Use Case Scenarios (Happy Path)

Use Case	Scenario	Result w/Comment
ADCL001	Check all text on page for correct spelling and grammar and consistency. **Expected Result:** Grammar and spelling are correct. Font styles are consistent and appropriate.	
ADCL002	Verify all links on page are active and display expected page in the expected window type. **Expected result:** All links function correctly and pages display as expected.	
ADCL003	Page results are correct	

There are no exceptions to test for the /admin/reportCustomerListing.php page.

/admin/addAdmin.php Use Case Scenarios (Happy Path)

Assumes tester has successfully logged on as Admin.

Use Case	Scenario	Result w/Comment
ADAA001	Check all text on page for correct spelling and grammar and consistency. **Expected Result:** Grammar and spelling are correct. Font styles are consistent and appropriate.	
ADAA002	Verify all links on page are active and display expected page in the expected window type. **Expected result:** All links function correctly and pages display as expected.	
ADAA003	Type an Admin First Name. **Expected result:** Field accepts data as designed.	
ADAA004	Type an Admin Last Name. **Expected result:** Field accepts data as designed.	
ADAA005	Type an Admin Login Name. **Expected result:** Field accepts data as designed; only unique login names are allowed.	
ADAA006	Type an Admin Password. **Expected result:** Field accepts data as designed.	
ADAA007	Type a Confirm Password value. **Expected result:** Field accepts data as designed.	
ADAA008	Click **Add New User**. **Expected result:** Field accepts data as designed.	
ADAA009	Check database tables to make sure data has been written is expected formats. **Expected result:** All tables have successfully updated and data is formatted correctly.	

/admin/addAdmin.php Use Case Scenarios (Exception)

Use Case	Scenario	Result w/Comment
ADAAE001	Check all text on page for correct spelling and grammar and consistency. **Expected Result:** Grammar and spelling are correct. Font styles are consistent and appropriate.	
ADAAE002	Verify all links on page are active and display expected page in the expected window type. **Expected result:** All links function correctly and pages display as expected.	
ADAAE003	Click **Add New User** without filling-in any fields. **Expected result:** Tester is prompted to complete required fields.	
ADAAE004	Type an **Admin First Name** without filling-in any other required fields, and then click **Add New User**. **Expected result:** Tester is prompted to complete required fields.	
ADAAE005	Type an **Admin First Name,** and **Admin Last Name** without filling-in any other required fields, and then click **Add New User**. **Expected result:** Tester is prompted to complete required fields.	
ADAAE006	Type an **Admin First Name, Admin Last Name,** and **Admin Login Name** without filling-in any other required fields, and then click **Add New User**. **Expected result:** Tester is prompted to type in a password and confirming password.	
ADAAE007	Type an **Admin First Name, Admin Last Name, Admin Login Name,** and **Admin Password** without filling-in **Confirm Password**, and then click **Add New User**. **Expected result:** Tester is prompted that passwords do not match.	
	Type an **Admin First Name, Admin Last Name, Admin Login Name,** and **Admin Password**. Type a non-matching password into **Confirm Password**, and then click **Add New User**. **Expected result:** Tester is prompted that passwords do not match.	

/admin/addAdminConfirmation.php Use Case Scenarios (Happy Path)

Use Case	Scenario	Result w/Comment
ADC001	Check all text on page for correct spelling and grammar and consistency. **Expected Result:** Grammar and spelling are correct. Font styles are consistent and appropriate.	
ADC002	Verify all links on page are active and display expected page in the expected window type. **Expected result:** All links function correctly and pages display as expected.	
ADC003	Check database tables to make sure data has been written is expected formats. **Expected result:** All tables have successfully updated and data is formatted correctly.	
ADC004	All results on page are correct.	

There are no exceptions to test for the /admin/addAdminConfirmation.php page.

/admin/usersStatusAdmin.php Use Case Scenarios (Happy Path)

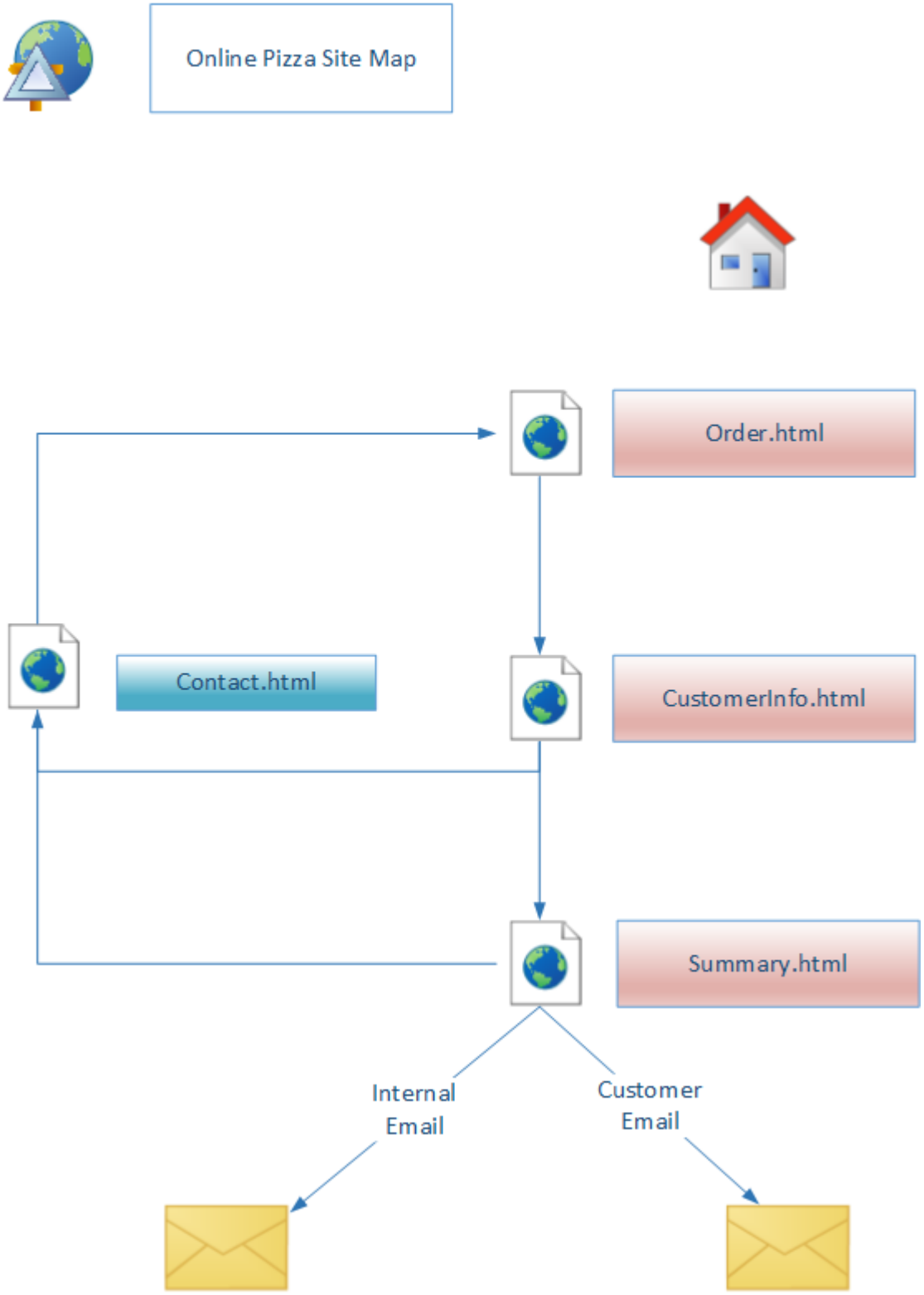
Use Case	Scenario	Result w/Comment
AD001	Check all text on page for correct spelling and grammar and consistency. **Expected Result:** Grammar and spelling are correct. Font styles are consistent and appropriate.	
AD002	Verify all links on page are active and display expected page in the expected window type. **Expected result:** All links function correctly and pages display as expected.	
AD003	Click inactivate. **Expected result:** Admin moves from active to inactive. Selected checkbox is disabled.	
AD004	Click activate. **Expected result:** Admin moves from inactive to active. Selected checkbox is disabled.	
AD005	All results on page are correct.	
AD006	Check database tables to make sure data has been written is expected formats. **Expected result:** All tables have successfully updated and data is formatted correctly.	

There are no exceptions to test for the /admin/usersActiveAdmin.php page.

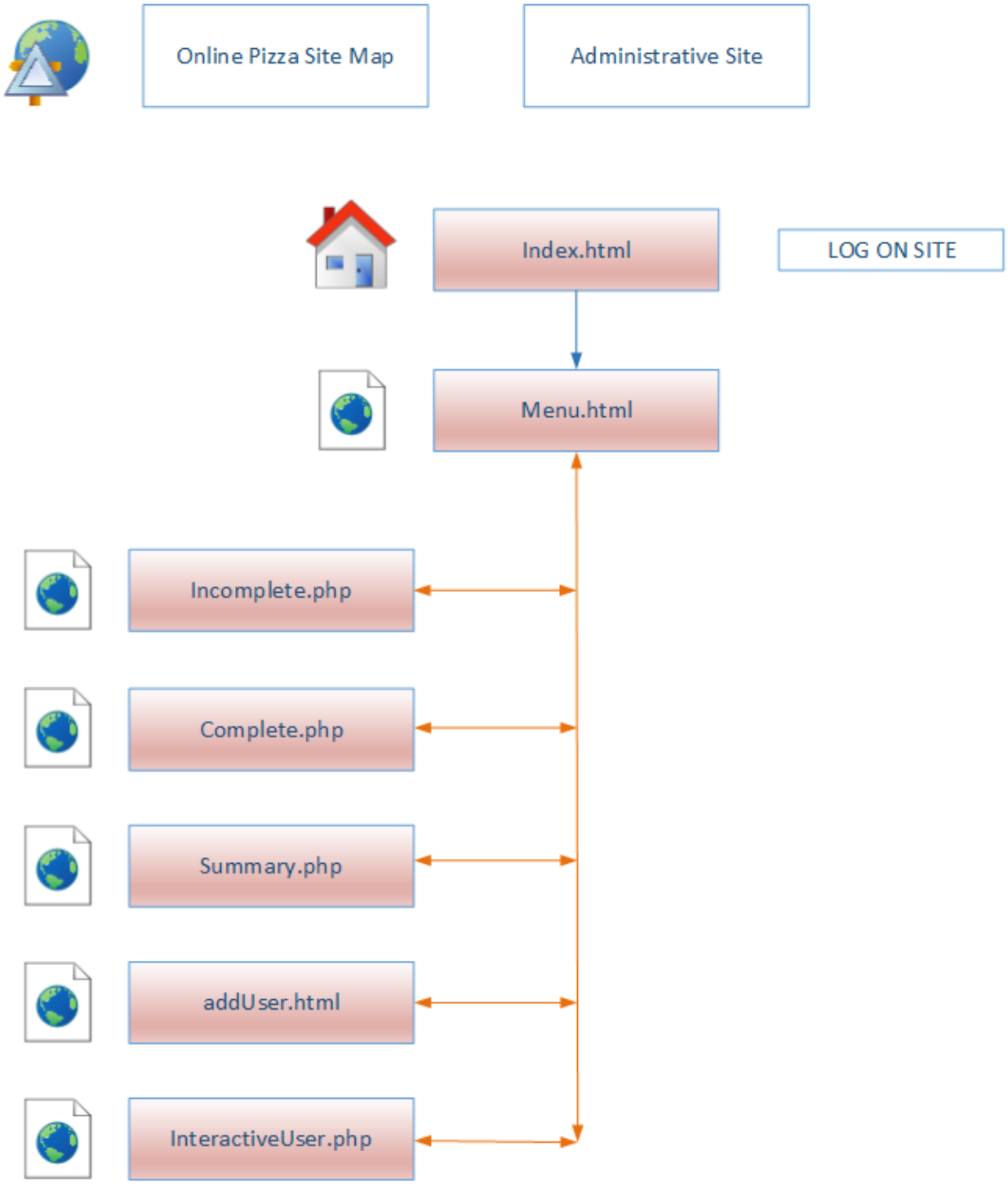
APPENDIX C – SITE MAP (by Paul Fischetti)

This sitemap is a graphical to-be representation of the upgraded PizzaParlor application.

PizzaParlor Sitemap



PizzaParlor Administration Sitemap

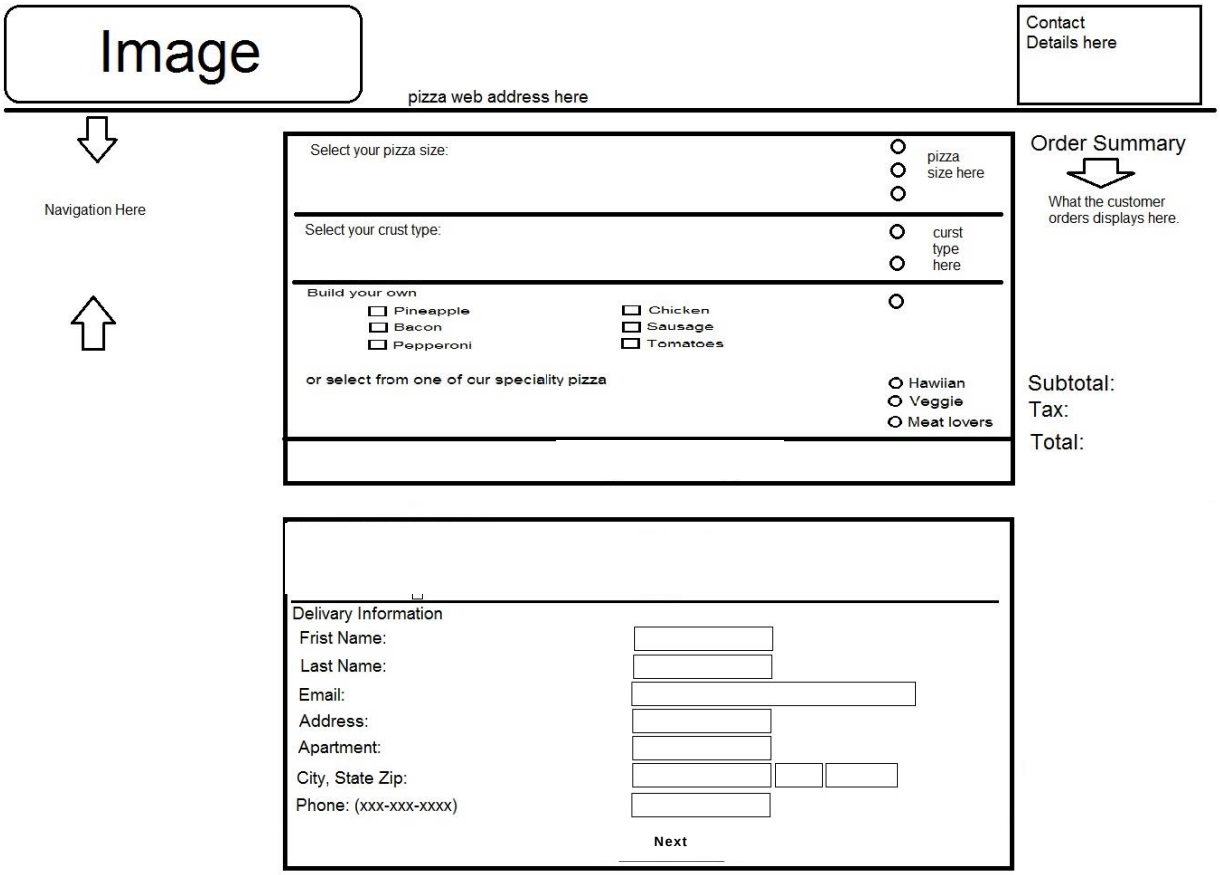


APPENDIX D – WIREFRAMES (by Claudia Du Plessis)

The following wireframes show the current and proposed pages for the upgraded PizzaParlor application. Wireframes suggest how certain elements may appear on a page; the actual design may change during the development process.

/order.php

The customer builds their pizza on the order.php page; the total price of the order, including tax, displays on the right side of the page. Then, the complete the delivery information and clicks **Next** to continue to a confirmation page (confirmation.php) OR clicks Cancel (clears fields).

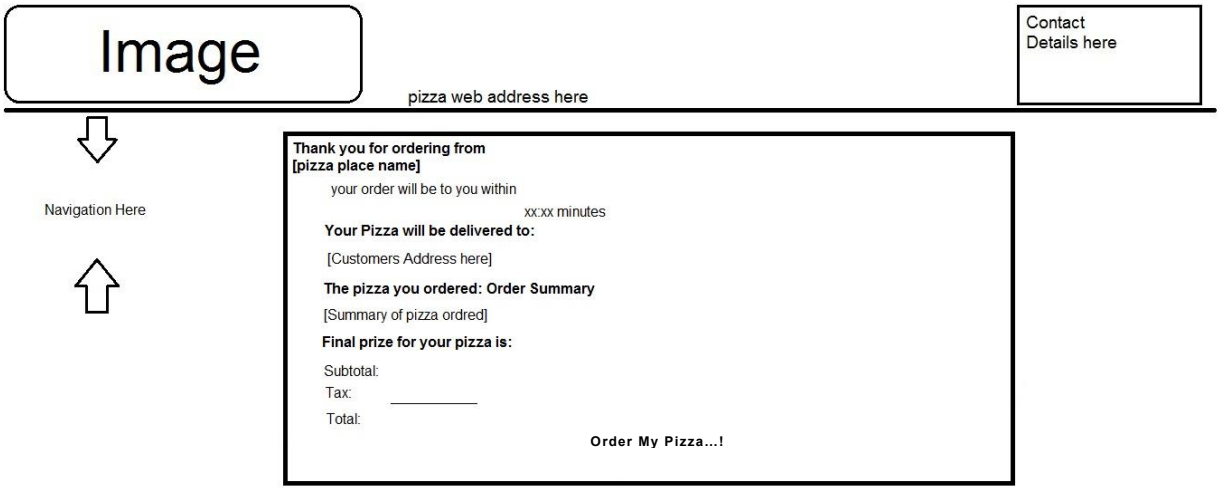


/confirmation.php

At the confirmation.php page, the customer clicks **Back** (returns to order.php page to change order or delivery info) or clicks **Next** (continues to summary.php page) or **Cancel** (clicking **Cancel** returns customer to order.php page and clears all fields).

/summary.php

The summary.php page displays an order summary, including cost, the delivery info, and the expected time for delivery. Customer clicks **Order my pizza!** All info and computed fields are written to the database, and emails are sent to the customer and PizzaParlor. Message or pop-up tells customer that email has been sent, thank you, etc.



/emailCustomer.php Confirming Email Sent to Customer

When the customer clicks **Order your pizza** on confirmation.php, an email summarizing all the info about the order is sent to email entered back in the delivery info section.

CONFIRMING EMAIL TO CUSTOMER

TO: xxxx@xxx.xxx
FROM: PizzaParlor
SUBJECT: <OrderID> Pizza Order

BODY: Includes order, delivery info, cost, and estimated time summaries. Marketing info like half-off pizzas ordered Tuesdays or something.

/emailPizzaParlor Order-in Email Sent to PizzaParlor

When the customer clicks **Order your pizza** on confirmation.php, an email summarizing all the info about the order is sent to the PizzaParlor email address.

ORDER-IN EMAIL TO PIZZAPARLOR

TO: PizzaParlor
FROM: xxxx@xxx.xxx
SUBJECT: <OrderID> Pizza Order

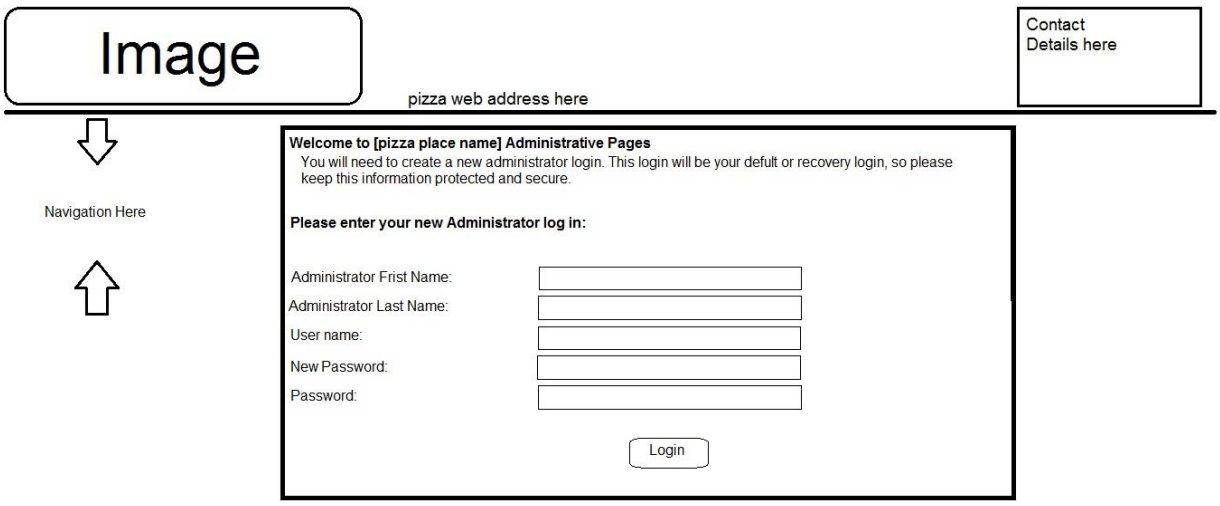
BODY: Includes order, delivery info, cost, and estimated time summaries.

/admin/index.php

The /admin folder lies at root of PizzaParlor folder. Home page is /admin/index.php. Enter valid **User name** and **Password**, and then click **Log in**. The admin/adminFirstLogin.php or admin/adminMenu.php page displays depending on state of admin login table.

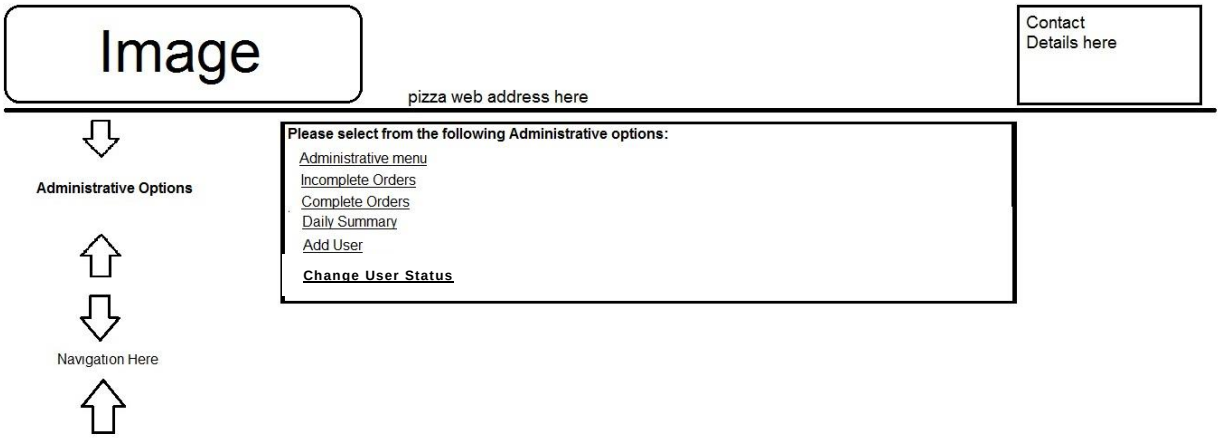
/admin/adminFirstLogin.php

This page displays if the Admin login table contains only one Admin. User is forced to add a second admin.



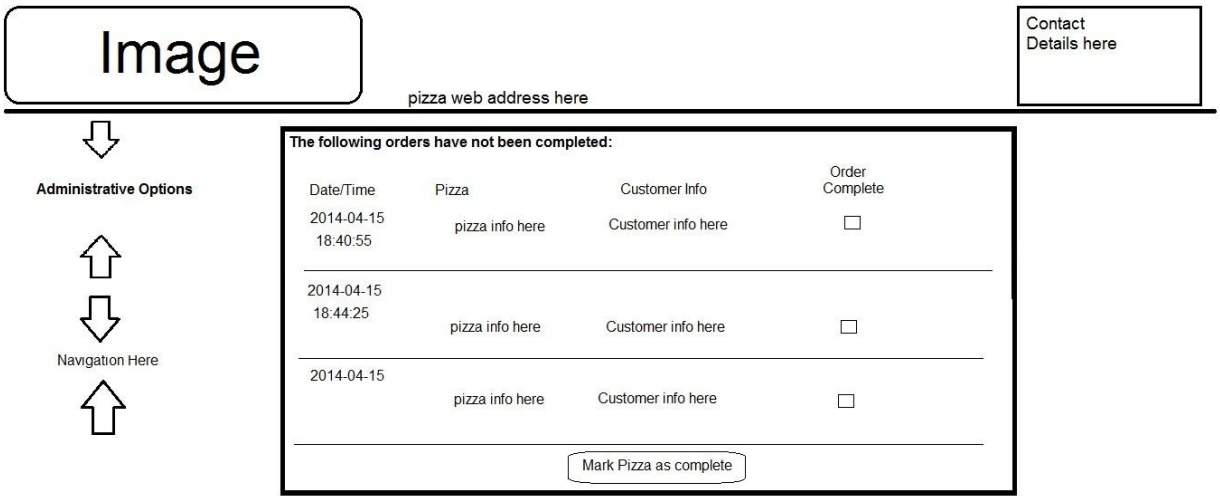
/admin/adminMenu.php

Admin menu displays after admin login (and admin login table containing at least two admins).



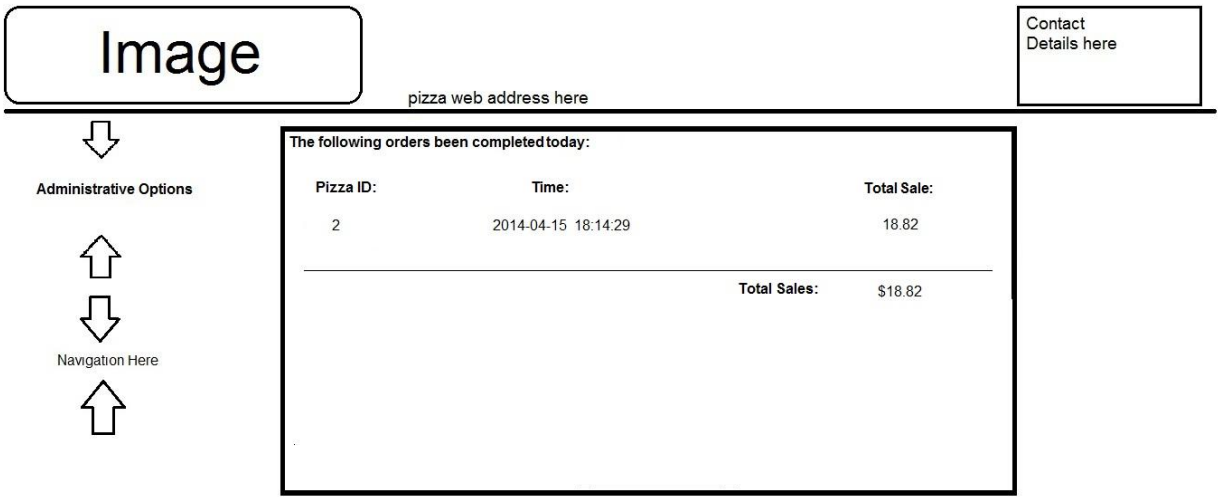
/admin/reportIncompleteOrders.php

This page is arrived at through the admin menu. It shows outstanding pizza orders. An admin clicks **Order Complete** and clicks **Mark Pizza as complete** to complete order. Page refreshes and removes checked pizza from outstanding orders. (Sets orderComplete bit)



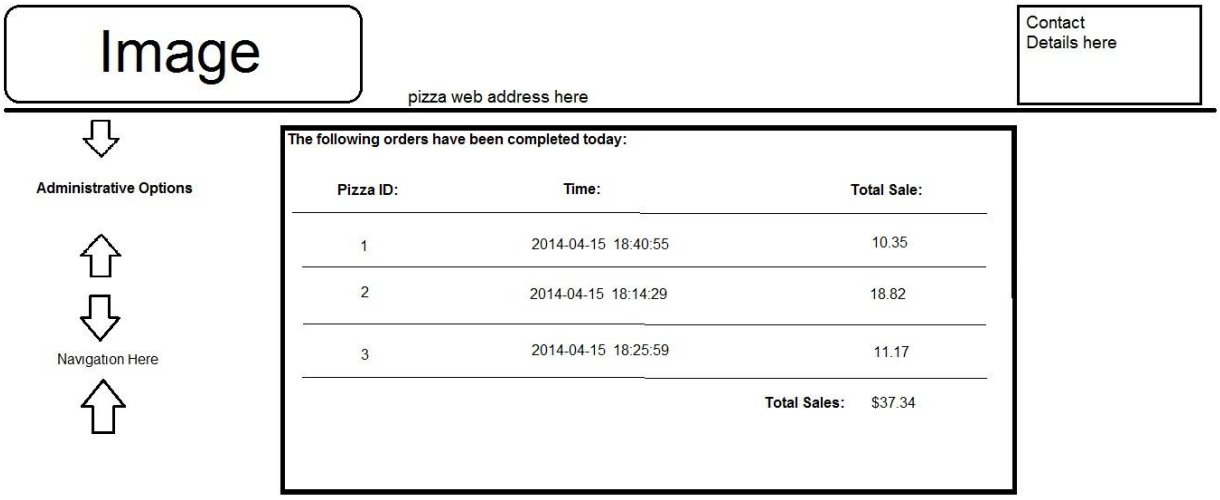
/admin/reportCompletedOrders.php

This page displays all pizzas for today where orderComplete bit is set to Yes. Lists total sales.



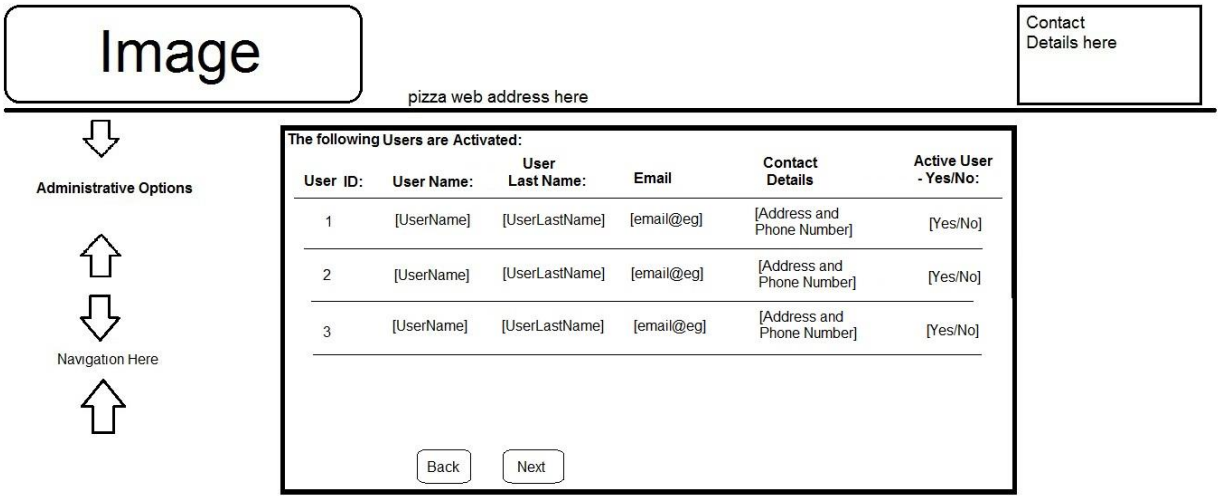
/admin/reportOrderSummary.php

Report shows all orders for today.



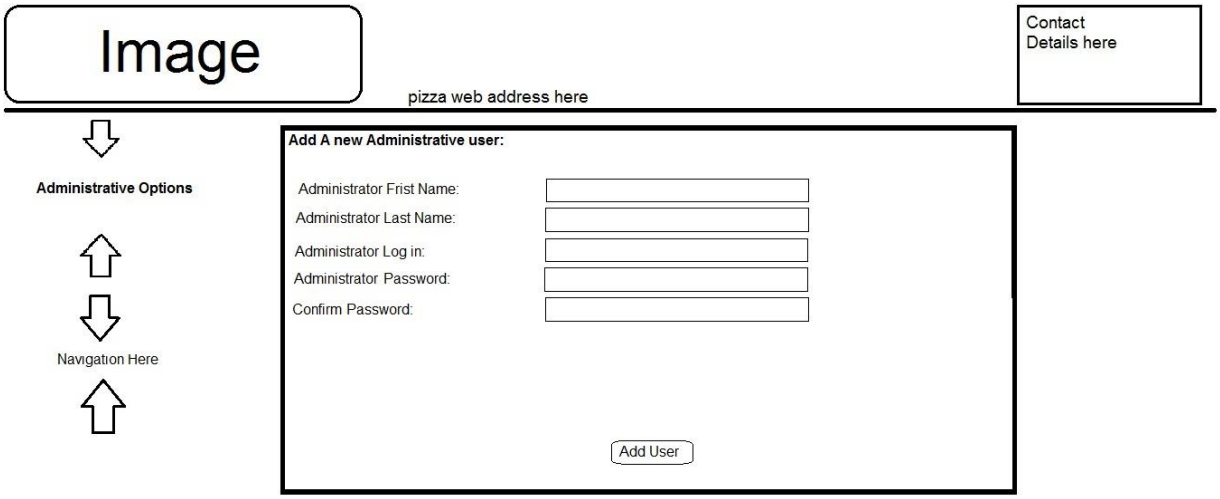
/admin/reportCustomerListing.php

Report shows customer info for mailings, emails, etc.



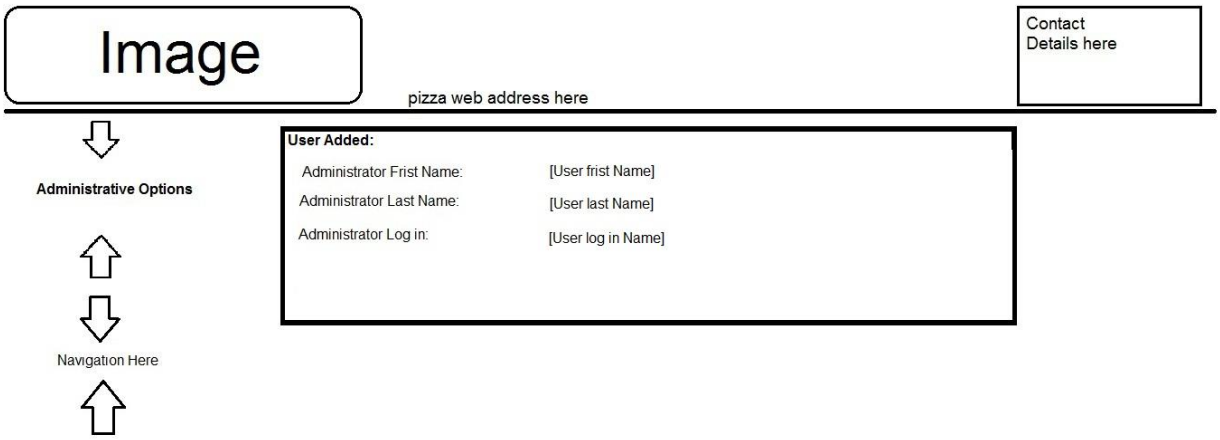
/admin/addAdmin.php

Admins may add other admins through this screen; click Add Admin to write to DB and see confirmation page.



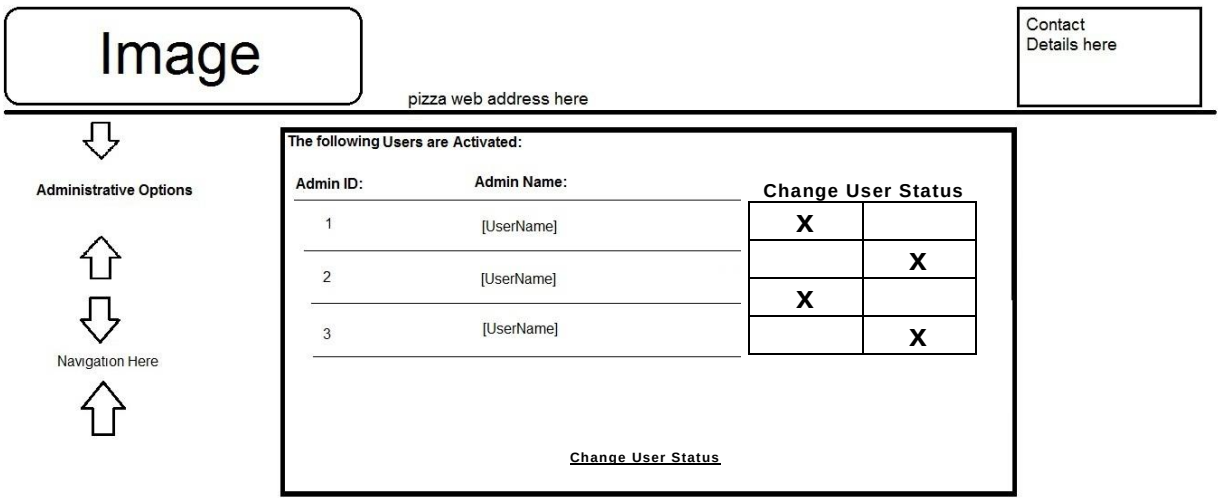
/admin/addAdminConfirmation.php

Summary of info entered on Add Admin page. Has link to Admin menu.



/admin/usersStatusAdmin.php

Screen shows Admin status (either active or inactive). Click check box and click **Change User Status** to set admins as active or inactive, depending on active checkbox.

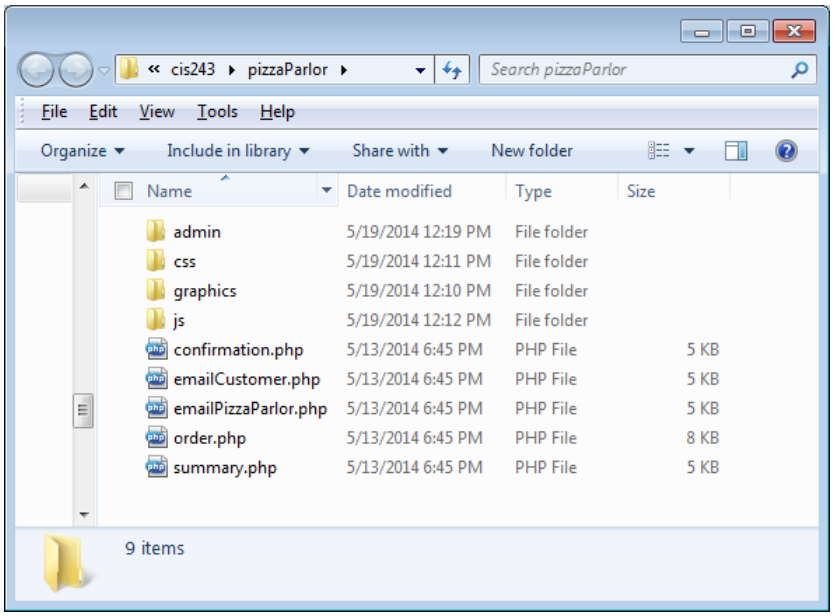


APPENDIX E – FILE STRUCTURE

The upgraded PizzaParlor site maintains its previous structure plus an added order confirmation.php page, and adds an Admin folder with added pages handling a variety of admin functionality.

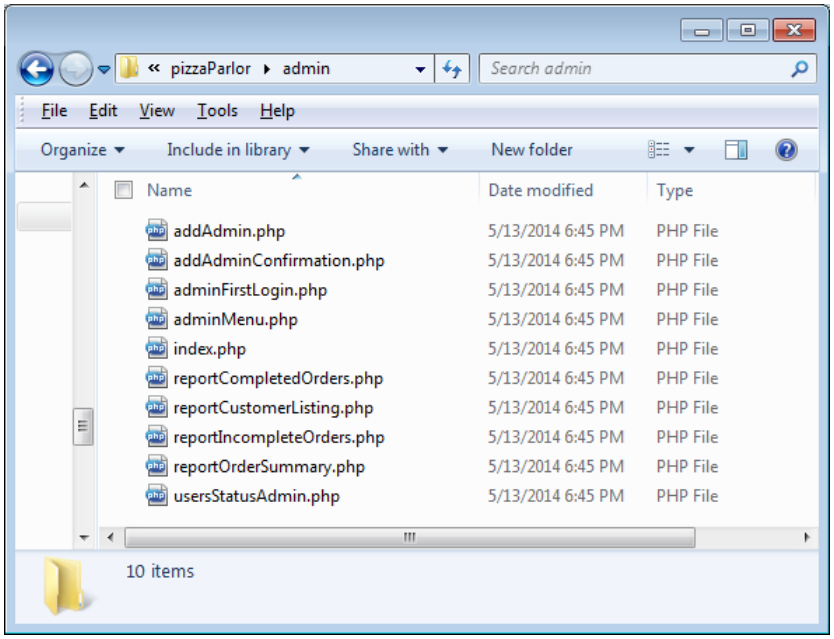
PizzaParlor Root Folder Contents (/*.*)

The project contains 5 folders (including the root PizzaParlor folder).



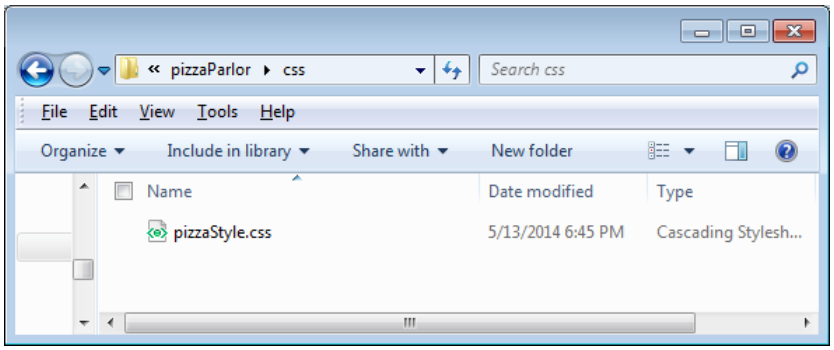
Admin Folder Contents (/admin/*.*)

The /admin folder contains 10 files, including index.php that servers as the Home Page.



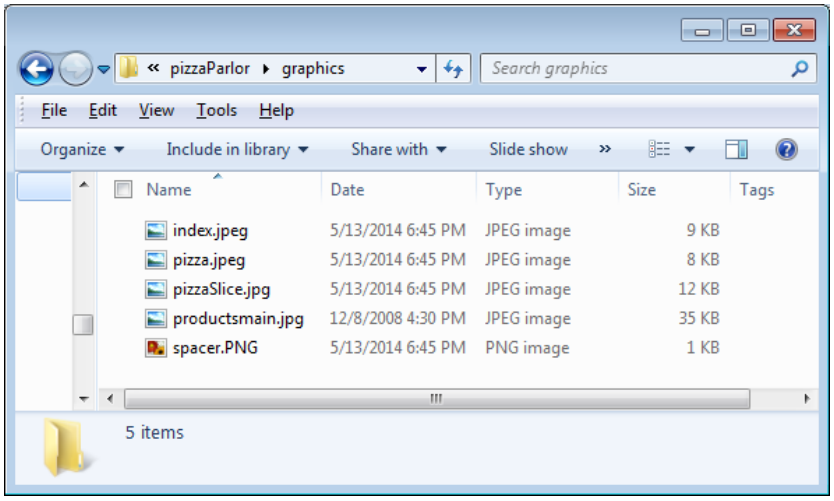
CSS Folder Contents

The /css folder contains 1 file that provides styling and layout info.



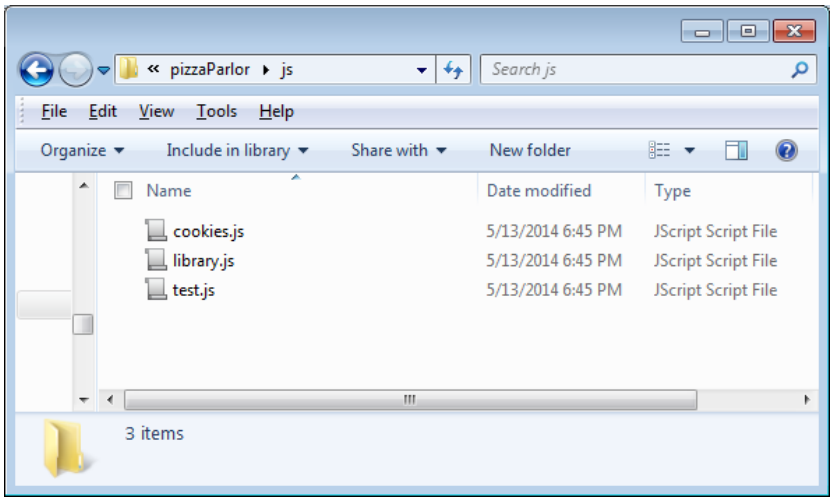
Graphics Folder Contents (/graphics/*.*)

The /graphics folder contains the 5 files used in Marti Baker's pizza parlor example.



JavaScript Folder Contents (/js/*.*)

The /js folder contains 3 files: cookies.js is the cookie handler supplied by Eric Collins; library.js contains miscellaneous js function; test.js was supplied with Marti Baker's pizza parlor example, however it is unknown if we'll have time to learn how to use it ☺.



APPENDIX F – BUG REPORT INFORMATION

Any bug written against the PizzaParlor will include the following information fields. See the document for recording bugs in the team's Canvas Bug folder.

- Bug ID (2014-DD-06 <BugNum>)
- Failed Test Case ID
- Tester Name
- Screenshot of bug
- Bug Status
    - Proposed
    - Active
    - Resolved
    - Closed
- Bug Priority Level
    - 1 (High)
    - 2 (Medium)
    - 3 (Low)
- Bug Severity Level
    - 1 (Critical)
    - 2 (High)
    - 3 (Medium)
    - 4 (Low)
- Expected Result
- Actual Result
- Additional Comments

See the following page for a sample bug report in Word format.

Sample Bug Report

See the Bug Report document in the team's Bugs folder in Canvas.

Two Bits and a Byte Bug Report		
Bug ID: 2014-DD-06	**Tester Name:**	**Failed Test Case ID:**
Bug Status:	**Bug Priority Level:**	**Bug Severity Level:**
Expected Result:		
Actual Result:		
Additional Comments:		
Screen Shot: Paste screen shot here…		