Technical Document

Database Setup

Introduction

This document outlines the process of setting up the RetroArc database using phpMyAdmin. The RetroArc project involves a database system to manage various aspects related to retro gaming consoles and products.

Creating the Database

1. **Accessing phpMyAdmin:** Logged into phpMyAdmin using the required credentials.
2. **Database Creation:** Created a new database named retroarc through the phpMyAdmin interface.

Overview of Tables

The database consists of the following tables:

- cart
- consoles
- orderproducts
- orders
- product
- producttype
- review
- status
- town
- user
- wishlist

Table Creation and Column Definition

1. **Cart Table:**
   - Created columns for cart items, quantities, and associated user details.
   - Set appropriate data types for each column (e.g., INTEGER for quantities).
   - Defined primary keys and foreign keys where necessary.
2. **Consoles Table:**
   - Included column for console names.
   - Ensured proper data types (e.g., VARCHAR for console names).
3. **OrderProducts Table:**
   - Columns for orderID, product ID, and quantity.
   - Applied relevant data types and constraints for order and product identification.
4. **Orders Table:**
   - Created to manage order information including order dates, status, and userIDs.
   - Configured columns with suitable data types like DATE for order dates.
5. **Product Table:**
   - Established to store information about products including names, descriptions, and prices.
   - Chose appropriate data types such as TEXT for product descriptions.
6. **ProductType Table:**
   - Formed to classify products into various types.
   - Included column for type with VARCHAR data type.

7. **Review Table:**
   - Designed for customer reviews, containing columns for ratings, comments, and user IDs.
   - Used INTEGER for ratings and TEXT for comments.
8. **Status Table:**
   - Set up to track the status of orders.
   - Included columns for status types and descriptions.
9. **Town Table:**
   - Created for storing towns related to user addresses.
   - Utilized VARCHAR for town names.
10. **User Table:**
    - Formed to manage user information including name, email address, and password.
    - Ensured secure data types like VARCHAR for emails and encrypted formats for passwords.
11. **Wishlist Table:**
    - Designed to keep track of user wishlists.
    - Included product IDs and user IDs with appropriate keys and data types.

Establishing Relationships and Keys

- **Foreign Keys:**
  - Established foreign keys to maintain relationships between tables. For example, linking user IDs in the cart table with the user table.
- **Primary Keys:**
  - Set primary keys for each table to uniquely identify records. For instance, product ID as a primary key in the product table.

Data Manipulation Techniques

Introduction

This section details the methods implemented for manipulating data in the RetroArc database through the website interface. It covers the creation of specialized functions and handler files designed to interact with the database.

Overview

Custom functions were developed to facilitate specific data retrieval and manipulation tasks. These functions are integral to the website's backend, enabling dynamic content display and database interaction based on user actions.

Key Functions

1. **getProductByTypeAndBrand:**
    - **Purpose:** To fetch products based on their type and brand.
    - **Operation:** When a user selects a product type and brand on the website, this function queries the product table to retrieve matching items.
    - **Implementation:** Utilizes SQL queries to filter products by the specified type and brand.

Handler Files for Database Operations

Overview

Handler files were created to manage various database interactions, including data retrieval for editing, validation, and alert display.

Key Handlers

1. **UpdateProductHandler.php:**
    - **Functionality:** Facilitates the editing of product details.
    - **Process:**
        - When a user opts to edit a product, updateProductHandler.php is called.

        - This file retrieves the current data of the specified product from the database.
        - The user is presented with a form pre-filled with the existing product information.
        - Upon submission of the form, the handler updates the product details in the database with the new information provided.

2. **Validation Handlers:**
    - **Purpose:** To ensure data integrity and user input validation.
    - **Implementation:** These handlers check the validity of user inputs (like form submissions) and ensure that they meet the required criteria before being processed or stored in the database.
3. **Alert Display Handlers:**
    - **Purpose:** To provide feedback to users based on their actions or system events.
    - **Function:** These files generate appropriate alerts or messages in response to user interactions, such as successful operations, errors, or validation failures.

Integrating Handlers with Website Functionality

- Handlers are integrated into the website's backend.
- They are triggered by specific user actions, such as clicking a 'Submit' button to update product information.
- The handlers interact with the database using SQL queries, performing actions like SELECT, UPDATE, or INSERT based on the operation.

<u>Setting Up a Local Virtual Server Using XAMPP</u>

Introduction

For the development and testing of the RetroArc website, a virtual server environment was essential. XAMPP, a popular PHP development environment, was utilized to create this local server setup. This section outlines the process and purpose of using XAMPP for the project.

XAMPP Installation and Configuration

Installation

1. **Downloading XAMPP:** The XAMPP software was downloaded from the official website.
2. **Installing XAMPP:** Followed the installation prompts, selecting components necessary for the project (Apache, MySQL, PHP, phpMyAdmin).

Configuration

- **Apache and MySQL Services:** Initiated Apache (web server) and MySQL (database server) services through the XAMPP Control Panel.
- **Port Settings:** Default port settings were used, ensuring compatibility with standard web server and database configurations.

Purpose of the Local Virtual Server

- **Development and Testing:** The local server environment provided by XAMPP allowed for the development and testing of the RetroArc website in a controlled, offline environment.
- **Database Interaction:** XAMPP facilitated the connection to the local instance of phpMyAdmin, enabling database setup and manipulation as would be performed on a live server.
- **Mimicking Live Environment:** By using XAMPP, the project could simulate a live server environment, ensuring that the website and its database interactions functioned correctly before deployment to a real server.

Techniques Used to Build a Dynamic Website

Introduction

The RetroArc website is designed to be dynamic, meaning it changes its content based on user interactions and inputs. This dynamic nature enhances user experience by providing relevant and customized content. Two key techniques are employed: dynamically displaying products based on category selection and showing product details based on user selection.

Dynamic Display of Products Based on Category

Overview

The website dynamically alters the product display based on the category and brand selected by the user from the navigation bar.

Technique

- **Category and Brand Selection:**
    - When a user selects a category and brand from the navigation bar, this action triggers a query to the database.
    - The website header is used to pass the selected category and brand information.
- **Database Querying:**
    - Based on the information received from the header, a query is executed to fetch products matching the selected category and brand from the product table.
- **Dynamic Content Generation:**
    - The retrieved data is then dynamically loaded onto the products page.
    - The page layout is updated to display the products pertinent to the chosen category and brand, enhancing the relevance of content for the user.

Displaying Product Details Based on Selection

Overview

When a product is clicked, the website dynamically loads and displays the details of that specific product.

Technique

- **Product Selection:**
    - Each product displayed has a unique identifier, typically the product ID.
    - When a product is clicked, this ID is captured and passed in the website header.
- **Fetching Product Details:**
    - The server uses the product ID from the header to query the database for detailed information about the selected product.
    - This query fetches data such as product description, price, reviews, and images.
- **Dynamic Detail Page:**
    - The product detail page is dynamically generated based on the retrieved information.
    - It displays all relevant details of the clicked product, providing a tailored experience for the user.

Test Cases

Test Case ID	Description	Input	Expected Process	Expected Output	Actual Output	Pass/Fail
TC01	Sign-In Page Access	Click Sign-In button	When button is clicked, sign in page is shown.	Takes user to the sign in page.	Sign in page displayed as expected	Pass
TC02	Empty Sign-In Fields Validation	Click Sign-In button (Not filled)	When button is clicked, a pop up appears stating that the fields need to be filled.	Shows a pop-up stating to fill all fields.	Pop-up displayed correctly	Pass
TC03	Invalid Sign-In Details Validation	Click Sign-In button (Filled with invalid details)	When button is clicked, a pop up appears stating that the data inputted is invalid.	Shows a pop-up stating that the data inputted is invalid.	Incorrect error message displayed	Fail
TC04	Incorrect Password Validation	Click Sign-In button (Wrong password)	When button is clicked, a pop up appears stating that the password is incorrect.	Shows a pop-up stating that the password is incorrect.	No pop-up, page refreshes	Fail
TC05	Unregistered Email Validation	Click Sign-In button (Email not registered)	When button is clicked, a pop up appears stating that the email is not registered.	Shows a pop-up stating that the email is not registered.	Correct pop-up displayed	Pass
TC06	Successful Sign-In	Click Sign-In button	When button is	Shows a pop-up	User signed in	Pass

		(Filled with valid details)	clicked, a pop up appears stating that user has signed-in.	stating that the user has been signed-in.	successfully	
TC07	Empty Register Fields Validation	Click Register button (Not filled)	When button is clicked, a pop up appears stating that the fields need to be filled.	Shows a pop-up stating to fill all fields.	No pop-up, button unresponsive	Fail
TC08	Invalid Registration Details Validation	Click Register button (Filled with invalid details)	When button is clicked, a pop up appears stating that the data inputted is invalid.	Shows a pop-up stating that the data inputted is invalid.	Correct validation message displayed	Pass
TC09	Successful Registration	Click Register button (Filled with valid details)	When button is clicked, a pop up appears stating that user has been registered.	Shows a pop-up stating that the user has been registered.	Registration successful, but no pop-up	Fail
TC10	Home Page Access via Logo	Click Logo	When logo is clicked, home page is loaded.	Takes user back to Home.	Home page accessed successfully	Pass
TC11	Navigation Link Functionality	Click Nav link	When a nav link is clicked, the dedicated page is loaded.	Takes user to the page that was clicked.	Correct page loaded	Pass
TC12	Access Nintendo Page	Click Nintendo sub nav link	When Nintendo sub nav link is clicked,	Takes the user to the Nintendo Page of the	Nintendo page accessed	Pass

			the dedicated page is loaded.	chosen category.	successfully	
TC13	Access PlayStation Page	Click PlayStation sub nav link	When PlayStation sub nav link is clicked, the dedicated page is loaded.	Takes the user to the PlayStation Page of the chosen category.	Incorrect page loaded	Fail
TC14	Access Sega Page	Click Sega sub nav link	When Sega sub nav link is clicked, the dedicated page is loaded.	Takes the user to the Sega Page of the chosen category	Sega page accessed successfully	Pass
TC15	Create New Product Page Access	Click create new product button	When button is clicked, user is redirected to the Create new product page.	Takes the user to the Create new product page.	Redirected to the correct page	Pass
TC16	New Product Creation	Click done button (New product)	When button is clicked, product is created.	Product is created.	Product is created	Pass
TC17	Edit Product Functionality	Click edit product button	When button is clicked, user can edit the details of the product.	Enables the editing functionality.	Editing functionality enabled	Pass
TC18	Product Edit Confirmation	Click done button (Edit product)	When button is clicked, product is edited.	Product is edited.	Product is edited	Pass

TC19	Product Deletion	Click delete button	When button is clicked, product is deleted.	Product is deleted.	Deletion confirmed	Pass
TC20	Product Page Access	Click Product Card	When a product is clicked, the product page is loaded.	Takes user to the clicked product's page.	Correct product page displayed	Pass
TC21	Add Product to Cart	Click Buy button	When buy button is clicked, the product is added to cart.	Product is added to the user's cart.	Product added successfully	Pass
TC22	Add to Wishlist (Not in Wishlist)	Click Wishlist button (Not in Wishlist)	If product is not in Wishlist and button is clicked, the product is added to the user's Wishlist.	Product is added to the user's Wishlist.	Wishlist updated correctly	Pass
TC23	Remove from Wishlist (Is in Wishlist)	Click Wishlist button (Is in Wishlist)	If product is in Wishlist and button is clicked, the product is removed from the user's Wishlist.	Product is removed from the user's Wishlist.	Removal from wishlist successful	Pass
TC25	Edit Review Option	Click Update review button	When the button is clicked, the review can be edited.	Enables the option to edit review.	Enables the option to edit review	Pass
TC26	Review Deletion	Click Delete review button	When the button is clicked, the user can delete their review.	Deletes review.	Review deleted successfully	Pass

TC27	Checkout Process	Click Proceed to Checkout button	When the button is clicked, the user checks out their cart and the order would be accepted.	User's order is accepted, and cart is emptied.	Checkout process incomplete, error in cart update	Fail
TC28	Remove Product from Cart	Click Remove button	When the button is clicked, the product chosen is removed from the cart.	Product chosen is removed from user's cart.	Product removed successfully from cart	Pass