Document for shoppingFT

Questions?

How to store img in database and how to retieve it?

A general practice is to store images in directories on the file system and store references to the images in the database. e.g. path to the image,the image name, etc.. Or alternatively, you may even store images on a content delivery network (CDN) or numerous hosts across some great expanse of physical territory, and store references to access those resources in the database.

Instead, consider updating your table to add an image\_path field. For example:

ALTER TABLE `your\_table`

ADD COLUMN `image\_path` varchar(1024)

Then store your images on disk, and update the table with the image path. When you need to use the images, retrieve them from disk using the path specified.

An advantageous side-effect of this approach is that the images do not necessarily be stored on disk; you could just as easily store a URL instead of an image path, and retrieve images from any internet-connected location.

How to update shopping cart?

1. Use form to get the input value;
2. Use foreach to check whether an element in the $\_SESSION[‘cart’]. Change the $value[‘quantity’] with the input value if the element is isset.

**Fatal error**: Uncaught PDOException: SQLSTATE[42000]: Syntax error or access violation: 1064 You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near ')' at line 13 in /opt/lampp/htdocs/sys11099/PHP/shoppingfruit/config/insert\_billing\_tbl.php:24 Stack trace: #0 /opt/lampp/htdocs/sys11099/PHP/shoppingfruit/config/insert\_billing\_tbl.php(24): PDOStatement->execute() #1 /opt/lampp/htdocs/sys11099/PHP/shoppingfruit/config/insert\_billing\_tbl.php(39): loadBilling(Array) #2 {main} thrown in **/opt/lampp/htdocs/sys11099/PHP/shoppingfruit/config/insert\_billing\_tbl.php** on line **24**

------ lastelement without , in the command

Absolute vs relative path in php

<https://code-boxx.com/php-absolute-relative-path/>

use dirname or

include "/opt/lampp/htdocs/sys11099/PHP/FurnitureStore"."/Model/product.php";

for the href inside the html

href="/sys11099/PHP/FurnitureStore/Controller/payment.php"

open new page in another window:

--window.open(“<http://192.168.64.2/sys11099/PHP/FurnitureStore/Controller/user/login.php>

”)



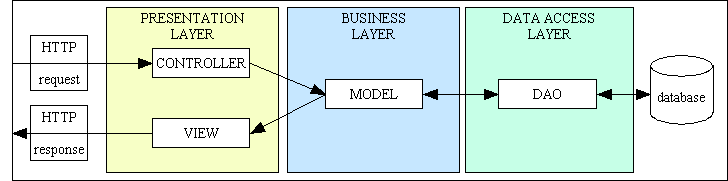
MVC design model is applied to Furniture Store application.

Model: The objects of products, order, billing, delivery, and customer hold their own properties with the appropriate methods, such as getting data from the database, inserting data to the database, validating user data against searching all of data in the database.

View: The header.php, index.php and main.css take the responsibilities for the the view.

Controller: for the logic part.

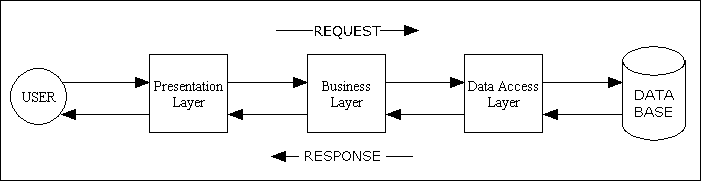
Figure 1 - MVC plus 3-Tier Architecture



DAO data access object

<https://www.tonymarston.net/php-mysql/3-tier-architecture.html>

Figure 6 - Requests and Responses in the 3 Tier Architecture



If you look carefully at those layers you should see that each one requires different sets of skills:

* The Presentation layer requires skills such as HTML, CSS and possibly JavaScript, plus UI design.
* The Business layer requires skills in a programming language so that business rules can be processed by a computer.
* The Data Access layer requires SQL skills in the form of Data Definition Language (DDL) and Data Manipulation Language (DML), plus database design.

In my largest application I have 2,000 components (user transactions) in the presentation layer, 250 in the business layer, and 1 in the data access layer. I have heard of some implementations which have a separate Data Access Object (DAO) for each individual table in the database, but more experienced developers can achieve the same functionality with just one. In my own implementation, for example, a single DAO can deal with every table in the database. However, I have a separate class file for each of the major DBMS engines - MySQL, PostgreSQL, Oracle and SQL Server - so I can easily switch from one to another by changing a single entry in my config file.

Note also that the connection to the database is not opened by any component within the Presentation layer. This should only be done within the Data Access layer when instructed to do so by the Business layer, and only the moment before an operation on the database is actually required. This is called the "Just In Time" (JIT) method as against the "Just In Case" (JIC) method. When the Business layer decides that it needs to talk to the database it follows these steps:

* Identify which DBMS is relevant for the database table.
* Instantiate an object from the relevant class file, which is usually a shared singleton.
* Pass a collection of variables to the DBMS object which will be used to construct the relevant query. This allows the query to be constructed according to the needs of that particular DBMS engine. The Oracle and SQL Server databases, for example, do not use OFFSET and LIMIT for pagination.
* Execute the query and wait for the response.
* Deal with the response before returning it as an array. Different DBMS engines, for example, have different methods of dealing with auto-increment columns or sequences, so the code to deal with these differences is contained within the DBMS object and totally invisible to the calling object.

$data = $database->fetchAssoc("SELECT \* FROM users WHERE name = ?", array("someone"));

<https://stackoverflow.com/questions/9525177/php-the-best-way-to-load-database-object-from-model-but-to-have-only-one-inst>

In a proper MVC the Model is a layer, and not a specific class. Model layer is composed from multitude of class/instances with two major responsibilities:

* domain business logic
* data access and storage

Figure 5 - Environment/Infrastructure Overview

