## Controllers

|  |  |  |  |
| --- | --- | --- | --- |
| Controller | Path | Written by | Allocated to |
| AbstractController.php | - | ML |  |
| AdminCategoriesController.php | /admin/categories | ML |  |
| AdminCategoryController.php | /admin/category | ML |  |
| AdminCustomerController.php | /admin/customer | Cainton |  |
| AdminCustomersController.php | /admin/customers | Cainton |  |
| AdminProductController.php | /admin/product | Not yet written |  |
| AdminProductsController.php | /admin/products | Cainton |  |
| CartController.php | /cart | Placeholder by ML | Glen |
| CheckoutController.php | /checkout | Placeholder by ML | Hayden |
| LoginController.php | /login | Example by ML |  |
| LogoutController.php | /logout | Example by ML |  |
| ProductsController.php | /products | Placeholder by ML | David |
| RegisterController.php | /register | Placeholder by ML | Jack |
| SearchController.php | /search | Placeholder by ML | Cainton |
| StaticController.php | - | ML |  |
| TestController.php | - | ML |  |

## Models

|  |  |  |
| --- | --- | --- |
|  | Written by | Status |
| AbstractEntityModel.php | ML | Complete, tested |
| AbstractModel.php | ML | Complete, tested |
| CategoryModel.php | ML | Complete, tested |
| CustomerModel.php | Cainton | Partial (needs validation, delete, tests) |
| CustomersModel.php | Cainton | Partial |
| ProductModel.php | Canton | Partial (needs validation, delete, tests) |
| ProductsModel.php | Glen/Cainton | Partial |
| ShoppingCartModel.php | ML | Tested, can be extended |
| StaticModel.php | ML | Complete, tested |
| TestModel.php | ML | Complete, tested |
| TestsModel.php | ML | Complete, tested |
| UserModel.php | ML | Complete, tested |

## Views

|  |  |  |  |
| --- | --- | --- | --- |
|  | Written by | Purpose | Status |
| AbstractView.php | ML | Library base class | Complete, tested |
| CustomersView.php | Cainton | Show list of customers (admin) | Partial |
| ~~CustomerView.php~~ | ~~Cainton~~ | ~~People View – to be deleted~~ | ~~Partial~~ |
| ProductsView.php | Cainton | Show list of products (admin) | partial |
| ListSelectionView.php | ML | A drop down list | Complete, tested |
| StaticView.php | ML | Render static html | Complete, tested |
| TableView.php | ML | Render as html table | Complete, tested |
| TestsView.php | ML | Display test results | Complete, tested |
| View.php | ML | Default view object | Complete, tested |

## Cart Controller

Note that there is an outline shopping cart model in the models folder. We need to add customer name and delivery address to this. Note that the shopping cart model does not use the database (and should not);’it is stored in the session object.

|  |  |
| --- | --- |
| Path | Function |
| /cart/view | Displays the shopping cart to the user. Embeds “remove this item” links in the display. Has a link to checkout. Handle the edge case of an empty cart by telling the user explicitly, “your shopping cart is empty”. |
| /cart/add/id | Asks user for quantity and confirmation, on post-back, validate and add specified quantity of item to cart. Finally, go to /cart/view |
| /cart/remove/id/ | Remove item from cart, then go to /cart/view |

## Checkout controller

We’ll need to add an order model and two supporting database tables: one for the order information (date, name, address, etc) and one for the list of products ordered.

|  |  |
| --- | --- |
| Path | Function |
| /checkout | If the cart is empty, tell the user, else ask for confirmation of purchase. IF confirmed, proceed as checkout/delivery |
| /checkout/delivery | Ask the user for delivery address information. On post-back validate and if OK, proceed as /checkout/payment (actually /checkout/final since we won’t be taking payments. |
| /checkout/payment | (WE’LL SKIP THIS) |
| /checkout/final | Create order and order details from the cart, empty the cart, tell the user “thank you for your order”. |

## Products controller

This should allow users of the site to browse products. Each product should have a link “add to cart” that should link to /cart/add/id, where id is the product id. It is particularly important to have a clear separation between the php code and html code here. The html designers should be able to edit an **html only** template for the product so they can create a responsive design for use across all devices.

## Register controller

Doing registration well is tricky. It’s often the weakest part of a system from a security perspective. Given the limited time left, we’ll take a simple, but safe approach; we’ll allow the user to enter registration but will not automatically activate the account. An administrator will have to review the data and then authorise the account. Due to time limitations, we won’t actually implement this second step.

The focus will be on careful validation of the user data, with particular regard to XSS risks. We won’t attempt to “saniitize” data but will simple ***reject*** data if sanitization changes the data in any way. So long as we do this, the standard mysqli escaping is safe.

The key process is to display a blank form on initial entry (/register with GET) and to validate on a post-back (/register with POST). If valid, add to the database else, redisplay the form with user data and error messages.

## Search Controller

|  |  |
| --- | --- |
| Path | Function |
| /search | Display a form asking the user for search terms, on post-back, process as one of the following |
| /search/n | Show products for category n |
| /search/n/term | Show products for category n that match the search term |
| /search/all/term | Show all products that match the search term, regardless of category |

The display should be the same as the products controller above.

Since URIs come from the client side, we have to take special care about security. Mostly we “white list” URIs, as per the front end controller, which is always a safe approach, but here we’re using the user’s search term in the URI. It is important that this is encoded by the php function ***rawurlencode***; don’t use the plain **urlencode**.