PHP Developer
2nd Round / Coding Hour

REQUIREMENTS

For this task we need to develop a web page to view and manage all inventories of an ecommerce shop. There are two components for this inventory manager: **a. Stock Management (Back-end)** and **b. Products View (Front-end)**

Stock Management (Back-end)
We need a multi-section administration web page (using tabs or side menu) that contains grids and trees for managing and doing CRUD on the following sections:

- Categories and Subcategories (Tree View)
  - Every product in a shop belongs to one category only. You can do a **GET** http://www.bamilo.com/mobapi/v2.3/catalog/categories to see list of possible categories
  - Each category itself can be a subcategory and belong to a parent category. Therefore, it's going to be a categories tree
- Products (Grid View)
  - The grid is fully functional supporting the fields mentioned in Table 1.1 - Products entity
- Attributes (Grid View)
  - The grid is fully functional where we can add attributes and assign it to a specific product category on Products grid view

The administration has authentication so a user should login in order to be able to manage these sections.

Database
We need to design a relational database for the stock management. The following are the required fields for each entity. **Feel free to add more tables and fields to solve for optimal database design**.

Entity	**Fields**
Categories	Title, desc
Products	title, category, model, attributes*, imgUrl, desc, price, status**, quantity
Attributes	title, category, desc

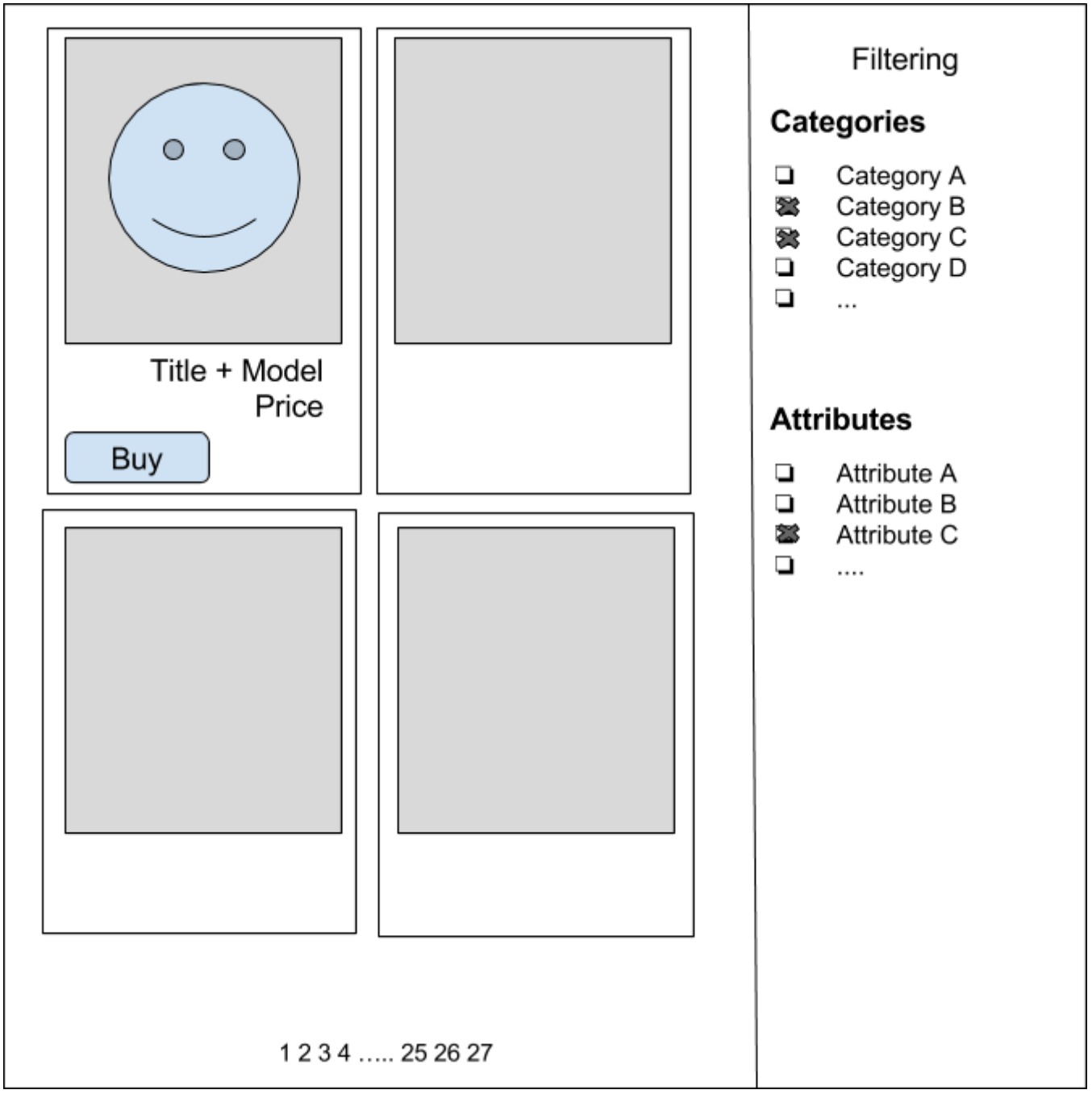
[Table 1.1]

* Attributes are properties of a product. Each product category may have different attributes. For instance, a shirt has size, color, etc. as its attributes. However, a teabag has flavor, pack-count, etc. Every attribute may have different options. For instance, a shirt size might be S, M, L, XL, and so on.

** Status are an enum with three options: *Not Available, Available and Coming Soon.* This is the status of that specific product.

Products View (Front-end)

We need to show the products to customers and they should be able to filter down based on Category and Attributes. The following is a wireframe of how the product view should look like:



The page kicks off under **/all-products** route where all products are initially loaded along with their attributes and categories. The user can select one or **multiple categories and/or attributes** for narrowing down their search. With each selection, there is an AJAX request sent to the backend and the relevant products are returned. The page is then rendered to show the new list of products. We need to use Memcached to store products for caching and conducting faster search.

All product cells have a main caption where title and the product model is shown. On top of that is the product image and below that is the price.

As shown in the wireframe, there is also a "**Buy**" button for every product. Each product has a Quantity field in database. We're going to use Cookies/Local Storage as a Shopping Cart so every time a user buys a product, we store it locally. This is to simulate the functionality of a shopping cart. In addition, we also would do an AJAX request to backend to update the product Quantity and reserve it as a potential sale.

TECHNOLOGIES

The following are the recommended technologies for this task. The Framework and Database of choice requirements are discrete. However, feel free to improvise on the rest of the technologies mentioned below:

Framework	Zend, Yii, Laravel
Database	MySQL (use ORM for data mapping)
Back-end Dependency Manager	Composer
Front-end Libraries	jQuery, Bootstrap, etc.
Front-end Package Manager	NPM, Bower, etc.
Bundlers	Gulp, Grunt, WebPack, Browserify etc.

[Table 2.1]