**Website Papelaria ABC Requirements**

By: João Luís Palma Esteves, 21/05/2025

Content

[Objectives 2](#_Toc202279451)

[Phase 1 2](#_Toc202279452)

[Phase 2 2](#_Toc202279453)

[Main requirements 2](#_Toc202279454)

[Phase 1 2](#_Toc202279455)

[Front-end 2](#_Toc202279456)

[Back-end 3](#_Toc202279457)

[Phase 2 3](#_Toc202279458)

[General Architecture 4](#_Toc202279459)

[Topographical illustration 4](#_Toc202279460)

[Classes and methods 4](#_Toc202279461)

[Gateway 4](#_Toc202279462)

[Database Manager 4](#_Toc202279463)

[Admin Client 5](#_Toc202279464)

[Website Controller 5](#_Toc202279465)

[Miscellaneous notes 5](#_Toc202279466)

# Objectives

## Phase 1

This website’s main purpose is to allow clients to search and peruse the shop’s catalogue, without having to be physically present. This need arises out of two situations:

1 – Since the business is a one-man operation, it turns the whole sale process more efficient, since the clients can walk up to the counter and skip the whole selection process.

2 – A consequence of situation 1, is that during the months of August and September, where people overwhelm the operation with presential orders of schoolbooks for their children, the process of accelerating people’s choices of miscellaneous school materials would greatly aid in alleviating workload and freeing up more time.

## Phase 2

Another objective is the inclusion of a user account system, which would allow for notifications in real time related to schoolbook deliveries, as well as the creation of an ordering system, where clients could place their orders through the website and be notified when to pick them up and pay for them. This would require careful data handling and security specifications, thus being relegated to phase 2.

# Main requirements

## Phase 1

### Front-end

#### General

All front-end pages should present, in the top bar, a top right button to go to a side drawer with options for settings (font size, language, bright/dark modes), more information about the store (telephone, email, location) and a review submission, where users can leave their thoughts about the website. The store’s logo must be on the top left of the top bar and a button to go to the landing page next to it. In the middle of said top bar there should be “Papelaria ABC”.

In the page that shows results, there should be an option, at the top right, to choose product ordering (by name, by date?, by price, by manufacturer?, etc.).

#### Landing page

The landing page is used to show images, through a slider that the user can manipulate but also slides periodically, with news about new products, discounts and other information. On the top of the page, below the top bar, there should be an option to peruse the store’s inventory and services, which will lead to the catalogue page.

#### Catalogue page

In the catalogue page, at the top of the page under the top bar, should be a search bar to “search all” products and services by name.

Option 1: In the middle, several clickable images display illustrations of various types of products and services (services, office materials, arts and crafts materials, toys, apparel, tech, etc.). All these options lead to the same search page with different search filters.

Option 2: On the left of the page there should be a list of check boxes that allow you to rule out certain sub-types of services and products (services, office materials, arts and crafts materials, toys, apparel, tech, etc.) in order to create a faster and more narrowed down search experience, at the cost of intuitive interactivity.

#### Option 1 chosen – Search page

If option 1 of catalogue page was selected, there should be all products shown in that subcategory, with the search terms only serving as filters to these initial appearing results.

## Phase 2

[yet to be defined]

# General Architecture

## Topographical illustration

A diagram of a diagram

AI-generated content may be incorrect.

## 

## Classes and methods

### Gateway

The core of the entire system, connecting the website’s controller to the Database Manager and serving as the access point for Admin Client.

**searchItems (name = null; price\_min = 0; price\_max = -1; type = null, ID = -1)**

Calls upon **dbFetch** from the Database Manager with ‘action = search’and returns the search results to the client.

**manageDB(action; name = null; price = 0; quantity = 0; type = null; ID = -1)**

### Database Manager

Handles interactions between other programs and the database, whether it be lookups, updates, etc.

**dbFetch (name = null; price\_min = 0; price\_max = -1; type = null, ID = -1)**

Reads database and returns a list with all the elements that fit the requirements of the Fetch arguments (naming, price range, type of product/service). These arguments have default values.

**dbUpdate (action; name = null; price = null; quantity = null; type = null; ID)**

Depending on the ‘action’ argument, this method can:

* Update a product: Looks up items by ID and if it finds some correspondence, updates the data with its respective input value (uninitialized values mean no alteration to said data).
* Add new products: The method looks up items by ID and if it finds some correspondence returns an error, indicating that such item addition creates conflict with already existing version. If no matches exist, the program creates a new entry in the database, in which case all arguments must have a valid value.
* Remove a product: Looks up items by ID and if it finds some correspondence, eliminates such element.

### Admin Client

A simple, straightforward and user-friendly class that allows for testing, maintenance and other capabilities by admins. It is meant to launch the entire program from scratch, as well as shutting it down.

**websiteLaunch()**

Launches all classes and starts the website

**manageDBinterface()**

Presents a list of options to perform on the database, like adding new elements, altering existing ones and deleting them. Calls upon **ManageDB** from Gateway.

**shutDownWebsite()**

Gracefully shut down all website classes and programs.

### Website Controller

The controller that manages the interactions between the website’s model and its views.

**showLandingPage()**

Returns the landing page of the website.

**showCataloguePage()**

Returns the catalogue page with the product type options to search. Calls upon the **searchItems** method from the Gateway.

**showResearchPage()**

Returns the research page with the searched products. Calls upon the **searchItems** method from the Gateway.

## Database Architecture

The database will work as follows:

* Upon startup, the database manager will create a list of objects. Each instance of these objects holds the corresponding data from the excel sheet and a list with the names of the images corresponding to the product in question.
* This list of objects will then be consulted when information is accessed.
* Whenever data is altered in the database, the first step will be repeated, as if it were a startup.

### Phase 1

**Image folder**

This will house all pictures of the products, and the naming convention should be: [item ID]\_[photo num]

**Excel sheet**

On the Excel sheet will be all the other info, such as availability, price, item ID, name, etc.

### Phase 2

To be defined.

# Miscellaneous notes

- make the html search page use GET instead of POST, unlike our SD project