PC Configurator

First Laboratory Deliverable

Interfaces Persona-Computador

A computer store wants to provide its customers with an application that will enable them to build their own PC by choosing the components from the catalogue. The application is focused to customers with any previous experience building PCs; therefore it should provide help to novel users. The goal of this first deliverable is designing and implementing an application that allows users to prepare a budget for a PC built from its components.

The user will be able to select the components based on information about their price and availability provided by the application. The system will provide tools for searching components according to different criteria (name, category, price, availability.)

The result of the configuration will be shown in a summary window that will contain a line per component, the total amount before taxes, the VAT (21%) and the total amount of the budget. Each line of the document will contain the selected component, the unit price before taxes, the quantity and the total amount before taxes of the line.

A PC is composed, at least, of the following elements:

- Motherboard
- CPU
- RAM memory
- Graphics card
- Hard disk (rotating media or SSD)
- Case

The customer can purchase other optional components, such as:

- Keyboard
- Mouse
- Monitor
- Speakers
- Card multireader
- DVD/Bluray
- Fan
- Power supply

You will have to implement the following functionality:

- When building the final budget, the system will warn the user if the budget does not include at least one unit of each of the non-optional components

- The user will be able to start from a preconfigured PC (the application will offer several of them), or from scratch, selecting all the components.
- The current budget must be editable (adding components, removing components, changing quantities, etc.)
- The application must allow the user to load and save the current PC configuration.
- It will be possible to show the final budget that, besides the components of the computer and the prices as explained above, it will show a header with an image with the logo and the name of the store. The footer should show the date when the budget was created, and a warning that it is valid only for 7 days.

The `Database` class provides a number of methods for querying the database. The database is based on the `Product` class, defined as follows:

```
public class Product {
    public enum Category {
        SPEAKER, HDD, HDD_SSD, POWER_SUPPLY, DVD_WRITER, RAM, SCREEN,
        MULTIREADER, MOTHERBOARD, CPU, MOUSE, GPU,
        KEYBOARD, CASE, FAN
    }

    public Product(String description, double price, int stock, Category
category) {
        this.description = description;
        this.price = price;
        this.stock = stock;
        this.category = category;
    }

    public Category getCategory() {
        return category;
    }

    public String getDescription() {
        return description;
    }

    public double getPrice() {
        return price;
    }

    public int getStock() {
        return stock;
    }
    private final String description;
    private final double price;
    private final int stock;
    private final Category category;
}
```

The `Database` class contains the following static methods for accessing the database:

```
public class Database {
  /**
   * Returns all the products in the category
   *
```

```
   * @param cat the category
   * @return the list of Products in the category
   */
   public static List<Product> getProductByCategory(Product.Category cat)

 /**
   * Searches the products of a given category with prices within the given range
   *
   * @param cat the category
   * @param minPrice the minimum price (in euros)
   * @param maxPrice the maximum price (in euros)
   * @param available if true, only return available products in the store
   * @return the list of products
   */
   public static List<Product> getProductByCategoryAndPrice(Product.Category cat,
double minPrice, double maxPrice, boolean available)
  /**
   * Searches products in a category by name
   *
   * @param cat the category
   * @param substring the description must contain this substring
   * @param available if true, only return available products in the store
   * @return the list of products
   */
   public static List<Product> getProductByCategoryAndDescription(Product.Category
cat, String substring, boolean available)

  /**
   * Searches products in a category by name, within a price range and taking into
   * account availability
   *
   * @param cat the category
   * @param substring the description must contain this substring
   * @param minPrice the minimum price (in euros)
   * @param maxPrice the maximum price (in euros)
   * @param available if true, only return available products in the store
   * @return the list of products
   */
   public static List<Product>
getProductByCategoryDescriptionAndPrice(Product.Category cat, String substring,
double minPrice, double maxPrice, boolean available)
}
```

It is advisable to define the class PC, which represents the configuration of a computer in a given moment. It will contain an optional name (for example, “Deal of the month”, or “For supergamers” or “My ideal PC”), and a list of objects of the class Component. The Component class will contain a quantity and a Product (thus the user can buy four memory modules or two GPUs). The PC class can be used in different parts of your application. For example, the preconfigured PCs can be stored in a list of PC objects. The PC being configured by the user can also use this class. The user cannot change the preconfigured PCs, but she will be able to use one as a starting point for defining her PC.

The Database and Product classes are provided in a library called **`DatabasePCStore.jar`**, available in poliformaT. For using these classes in your project in NetBeans, right click on the

Libraries folder in the Projects explorer and select the option Add JAR/Folder and select the previous file.

Optional extensions:

The following optional extensions can be implemented for obtaining extra credit:

- Allow the user to work with several budgets.
- Allow the user to print the final budget.

Delivery rules:

- Export the NetBeans project in a ZIP (http://politube.upv.es/play.php?vid=68666) and upload it to the poliformaT task for this assignment. Each group will upload the project only once, including in the comments section the name of both students.

Evaluation:

- The main window must be resizable. The position and size of their controls should adjust to use the available space (use the containers studied in the lab).
- The projects that fail to compile or that fail to show the main window after launch will be graded with a zero.
- The application should include all the required confirmation dialogs, error dialogs, etc.
- The design of the interface will take into account the design guidelines studied in class.