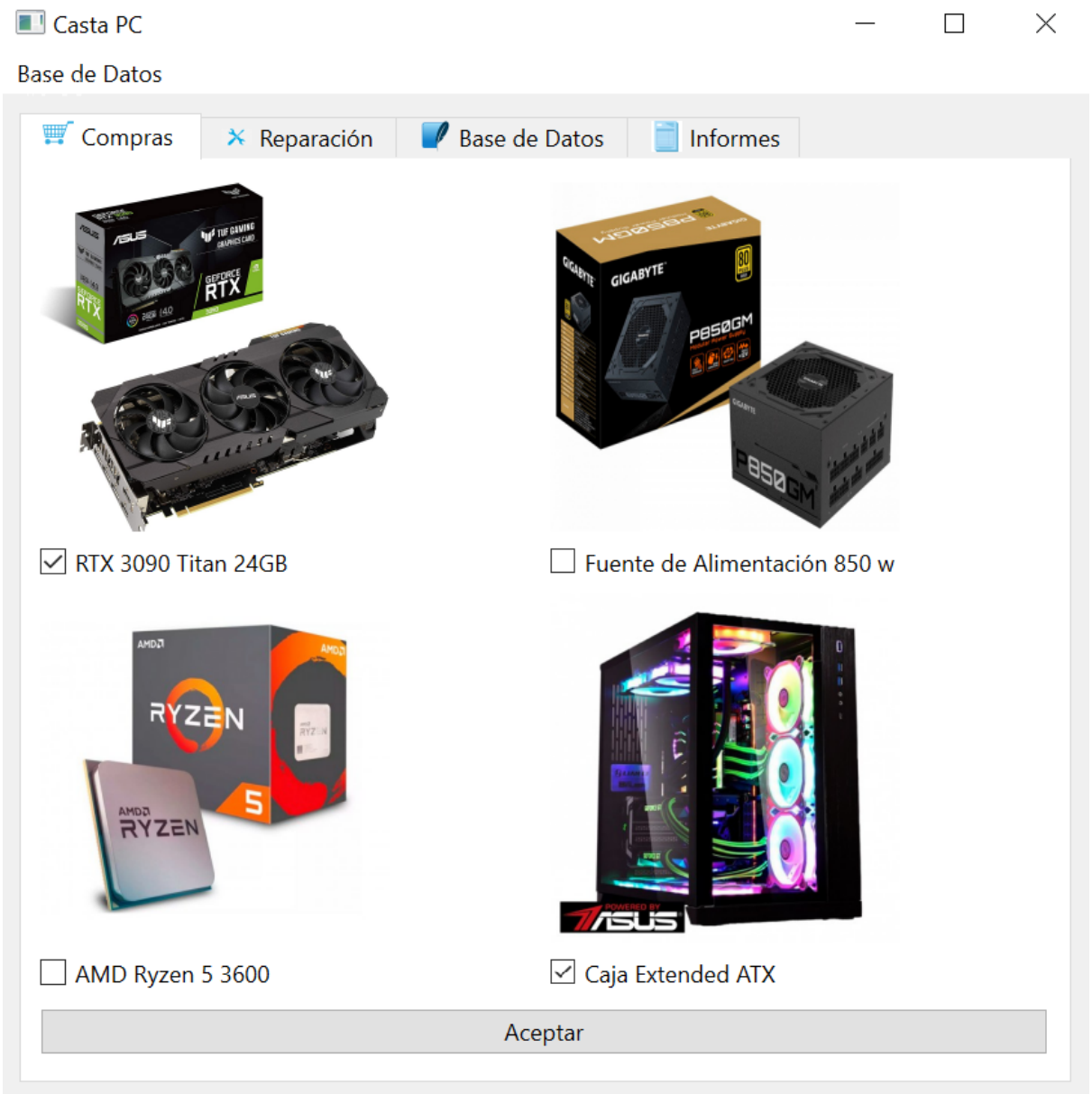
Guía de Referencia de Casta PC



ÍNDICE

Introducción

Aquí voy a hablar de la estructura interna de la aplicación, me centraré en las partes importantes.

Clase DB

Para empezar voy a hablar de la clase DB que se encuentra en DB.py, en esta clase tenemos su función más importante que es *lanzarDB.*

Def lanzarDB

Esta función se encarga de establecer conexión con la base de datos y la aplicación a través del fichero *BaseDatos.sqlite*, además también carga los datos almacenados de la base de datos en las checkBox de los productos y en la tabla de la pestaña de Base de Datos y junto a esta función hay otras 4 que funcionan sobre la base de datos y la aplicación.

Def seleccion

Esta función ayuda a gestionar la base de datos en la aplicación, lo que hace es que cuando seleccionamos un producto en la base de datos nos recoge los datos al seleccionarlo directamente desde la base de datos.

Def modificar

Esta función ayuda a gestionar la base de datos en la aplicación, su función es cambiar los datos introducidos por los que haya en la base de datos.

Def nueva

Esta función ayuda a gestionar la base de datos en la aplicación, esta en concreto es muy simple, sólo crea un nuevo producto con todos los campos vacíos aunque luego la base de datos le añade una id que se va incrementando automáticamente.

Def borrar

Esta función ayuda a gestionar la base de datos en la aplicación, nos borra el producto que seleccionemos.

Def generaImgGrafica

Crea una gráfica con los datos de la base de datos del stock (aunque actualmente sólo recoge los cuatro primeros valores), una vez crea la gráfica la exporta a imagen donde más adelante se implantará en el PDF.

Clase Main

Desde la clase Main es donde conecto con los componentes con los datos, incluso también se crean componentes dentro de esta clase, en su mayoría para los wizard.

Def contadorProductos

Esta función se encarga de calcular el precio de los productos seleccionados e incluso introduce las id_producto de los productos seleccionados.

Def actualizarDatosWizardReparacion

Esta función es para mostrar los datos introducidos en el wizard de reparación en la última página antes de que éste imprima el PDF.

Def actualizarDatosWizardCompra

Esta función es para mostrar los datos introducidos en el wizard de compra en la última página antes de que éste imprima el PDF.

Def iniciarWizardReparacion

Abre una nueva ventana y en ella inicia el wizard de reparación.

Def iniciarWizardCompra

Abre una nueva ventana y en ella inicia el wizard de compra.

Def generaPDFwizardReparacion

Obtiene los datos introducidos en el wizard de reparación para pasarlos a un objeto canva el cual colocará los datos donde se le indique a través de un PDF, haciendo que al colocar unas coordenadas exactas sobre una plantilla se coloquen los datos en un nuevo PDF correctamente.

Def generaPDFwizardCompra

Obtiene los datos introducidos en el wizard de compra para pasarlos a un objeto canva el cual colocará los datos donde se le indique a través de un PDF, haciendo que al colocar unas coordenadas exactas sobre una plantilla se coloquen los datos en un nuevo PDF correctamente, además obtiene la imagen de la gráfica generada por la función generaImgGrafica de la clase DB para añadirla al PDF.

Def mostrarCompraPDF

Con un objeto QWebEngine hacemos que al utilizarse este método se muestre el PDF de compra.

Def mostrarReparacionPDF

Con un objeto QWebEngine hacemos que al utilizarse este método se muestre el PDF de reparación.

Def __init__

Esta es la función principal, en ella podemos ver que se llama a las funciones de lanzarDB y generaImgGrafica de la clase de DB, también deja preparado el QWebEngine para que se vea el PDF de compra llamando a la función mostrarCompraPDF.

Se añaden las relaciones de botones y funciones.

```
self.botonCompraInforme.clicked.connect(self.mostrarCompraPDF)
self.botonReparacionInforme.clicked.connect(self.mostrarReparacionPDF)

self.botonAceptarCompras.clicked.connect(self.contadorProductos)
```

Se crea todos los componentes de ambos wizards y entre medias se utilizan los botones de next y finish del propio wizard para mostrar datos o para generar gráficas, por ejemplo:

```
page5.setFinalPage(True)

finish = self.wizard.button(QWizard.FinishButton)

finish.clicked.connect(self.generaPDFwizardReparacion)
```

Clase Ui_MainWindow

Esta clase está generada con la aplicación Qt Designer y transformada a través de un comando a .py

Todo lo de esta clase está generado de forma automática, desde aquí se puede ver todo el esqueleto de la aplicación, se puede ver todos los componentes creados en un primer momento y conectados entre sí en perfecta sinfonía junto a los recursos de imágenes, es mejor no tocar nada de aquí ya que cualquiera modificación es preferible hacerla desde otra clase.

Archivo Recursos.py

Aquí están todas las imágenes que he utilizado para la aplicación, este fichero también ha sido generado a través de la aplicación Qt Designer y modificado por un comando para generar este py.

Este archivo se sincroniza con la clase Ui_MainWindow para pasarle todas las imágenes.