Texte :

Classes

Premièrement, je vais vous parler des classes que l’on a créé pour pouvoir enregistrer les labels avec les données qu’il y a dedans pour avoir la capacité de les .destroy() plus tard. Ces classes ne font que classer dans les listes all\_texte et all\_find les données qui y sont entrées. La classe all\_texte est là pour les informations affichées à partir des listes tirées des bases de données et all\_find est là pour les autres labels créés. La différenciation est nécessaire.

Fonctionnement des recherches par catégories.

Lorsque l’on choisit la fonction recherche, des boutons s’affiche par catégories. Lors du choix d’une de ces catégories, il est possible de trier les informations par marque.

La question est : Comment récupérer ces informations ?

Pour cela on utilise sqlite3 qui récupère les informations depuis une bdd dans laquelle se trouvent 2 tables.

Tables que l’on met n lien via la méthode INNER JOIN qui permet de joinder les deux tables.

Pour l’affichage, on utilise simplement une boucle ( traversant les colonnes) et une boucle à l’intérieur permettant de récupérer les informations pour les lignes.

Les données sont mises dans une classes pour être supprimées plus tard afin d’éviter d’avoir des textes qui se superposent. ( Par exemple, lors d’un changement de catégorie)

* Compositions de configuration

Décrire le tableau.

Dans la fonction search for choice, on obtiens les compositions compatibles à partir du prix et en fonction du choix grâce à des fonctions (Montrer au tableau)

Dans le cas où la liste obtenue est vide, on exprime simplement à l’utilisateur qu’il n’y a ps de compositions compatibles au budget rentré.

Explications de chacune des fonctions.

Pour bureautique : On affiche simplement les compositions compatibles pour le prix car dans compositions compatibles, il n’y a pas de barres de ram. Pour récupérer les information, on utilise une requête sql.(Montrer la requette) . La fonction renvoie à la fin les compositions possibles avec le budget entré et pour le choix fait par l’utilisateur. Seulement, pour réduire le nombre de résultat, on utilise une fourchette de prix. Ici, c’est entre le prix et le prix – 1 euro car dans le cas contraire des milliers de compositions pas spécifiquement nécessaires vont s’afficher.

Pour Graphisme : On récupère aussi les informations depuis la bdd composition compatibles via une requête sql mais en plus de cela, on récupère les informations de la bdd ram pour ensuite chercher toutes les compositions possibles avec dans une fourchette de prix de 0.5 en dessous de budget.

Regarder tableau. De plus, on fait un autre tri en fonction du nombre de Go de la carte. Une carte graphique possédant au moins 6 GIGA est considérée comme puissante or c’est ce qui est nécessité pour avoir de beaux graphismes.

De même pour vitesse mais ou les compositions choisies se font plus en fonctions du nombre de cœurs. En effet, on considèrera un processeur puissant lorsque celui-ci possède au moins plus de 8 cœurs

De même pour le stockage. La capacité de stockage d’un pc dépend du disque dur interne et c’est pourquoi on vérifie si la capacité du disque dur est suffisamment importante.

Ce système est sur des fonctions comptant le nombre de pages et détruisant les labels inutiles lors des changements de page. Ce système permet d rester à seulement 16 lignes d’informations affichées par page afin de pouvoir afficher toute les informations et don ici toute les compositions possibles.