Une image contenant texte, affiche, habits, personne

Description générée automatiquement

SOMMAIRE

[1. Description de la base 4](#_Toc161762999)

[1.1 L’OBJECTIF DE NOTRE APPLICATION 4](#_Toc161763000)

[1.2 LE FONCTIONNEMENT DE LUDOCIEL 4](#_Toc161763001)

[1.3 L’UTILITE, L’ALIMENTATION ET L’EXTRACTION de notre base de données 5](#_Toc161763002)

[1.4 LES LIMITES 5](#_Toc161763003)

[2. Le dictionnaire des variables 7](#_Toc161763004)

[3. La construction de la base 8](#_Toc161763005)

[3.1 LES ENTITES : 8](#_Toc161763006)

[3.2 LES RELATIONS : 8](#_Toc161763007)

[3.3 LES CLES PRIMAIRES *de chacune des entités* : 8](#_Toc161763008)

[3.4 MODELE MCD : 9](#_Toc161763009)

[3.5 MODELE MCD : 9](#_Toc161763010)

[4. Le fonctionnement de l’application 10](#_Toc161763011)

[4.1 LES FORMULAIRES 10](#_Toc161763012)

[4.2 LA NAVIGATION ENTRE LES FORMULAIRES 15](#_Toc161763013)

[4.3 LES REQUETES ET LES ETATS PROPOSES 16](#_Toc161763014)

[4.4 LES PLUS : 17](#_Toc161763015)

[5. Les apports du projet 18](#_Toc161763016)

[6. Conclusion 19](#_Toc161763017)

DESCRITPTION DE LA BASE

1.

## Description de la base

# L’OBJECTIF DE NOTRE APPLICATION

Imaginez un concept novateur où une bibliothèque traditionnelle est transformée en une bibliothèque de jeux de société. Au lieu de parcourir des rayons de livres, vous pouvez désormais explorer une collection de jeux de société variés. De manière pratique, vous pouvez accéder à cette bibliothèque depuis votre ordinateur ou votre téléphone, et louer un jeu pour une durée déterminée d'un simple clic. Le jour-même, votre sélection de jeux est livrée directement à votre porte. Ludociel, notre bibliothèque spécialisée dans les jeux de société, offre ainsi à chacun la possibilité de découvrir de nouveaux jeux sans avoir à les acheter. Cette approche permet de réduire les coûts tout en évitant d'investir dans des jeux qui pourraient ne pas correspondre à vos préférences.

# LE FONCTIONNEMENT DE LUDOCIEL

L'essence même de cette bibliothèque réside dans la possibilité de louer des jeux de société selon vos préférences. Que vous recherchiez des jeux d'ambiance ou des jeux de plateau, notre collection offre une variété de catégories pour répondre à vos attentes, vous offrant ainsi un large éventail de choix. Une fois votre sélection effectuée, il vous suffit de les louer et de profiter d'agréables moments en compagnie de vos amis ou de votre famille. Si vous trouvez difficile de choisir parmi notre gamme de jeux, les commentaires et les avis laissés par d'autres utilisateurs peuvent vous guider dans votre décision en fonction de vos préférences personnelles. Nous fournissons également des informations essentielles telles que le nombre de joueurs requis, la durée approximative du jeu de société, ainsi que le niveau de complexité de compréhension de ses règles.

Le système de location repose sur des tarifs différents en fonction du nombre de jours de location définis par Ludociel. Voici le tableau des prix de location que nous appliquerons pour la location d'un jeu de société.

|  |  |
| --- | --- |
| 2 jours | 1,50€ |
| 5 jours | 2€ |
| 1 semaine | 2,50€ |
| 2 semaines | 3€ |

L'utilisation de notre application est simple et intuitive. Pour démarrer, vous devez créer un compte, ce qui constitue une adhésion à notre bibliothèque entièrement gratuite. Lors de cette étape, vous devrez fournir vos informations personnelles telles que votre nom, prénom, civilité, date de naissance, numéro de téléphone, adresse e-mail et adresse postale. Ces informations nous permettront de connaître votre âge si vous laissez des avis, et nous demandons votre adresse afin de pouvoir livrer les jeux de société. Enfin, votre numéro de téléphone et votre adresse e-mail nous seront utiles pour vous contacter plus facilement. Ensuite, vous choisirez un pseudonyme qui vous sera unique, ainsi qu'un mot de passe pour accéder à votre compte en toute confidentialité. Notre application permettra à tout utilisateur qui a oublié son mot de passe de le visionné pendant 5 secondes via un formulaire. Une fois votre compte créé, vous pourrez naviguer librement dans l'application, louer les jeux de société qui vous intéressent ou simplement consulter les avis et les commentaires.

Vous pouvez également filtrer les jeux qui correspondent à vos attentes en fonction des informations fournies sur ceux-ci. Enfin, les utilisateurs auront accès à leur historique de location, ce qui leur permettra de consulter les jeux de société qu’ils ont déjà loué et d'avoir un aperçu de leur historique. De plus, ils auront la possibilité d'imprimer leur historique de location, ce qui leur permettra de visualiser les prix de chaque location ainsi que le prix total de toutes leurs commandes. Ce document affichera des informations telles que le jeu de société loué, la date d'emprunt et la durée de la location.

# L’UTILITE, L’ALIMENTATION ET L’EXTRACTION de notre base de données

La base de données a pour **utilité** de stocker les informations des utilisateurs, leur permettant ainsi de se connecter à leur compte et de le gérer en conservant leurs données personnelles. Elle assure également le stockage des informations des jeux de société, enregistrant les détails de chaque jeu. De plus, elle permettra de conserver le suivi des locations en maintenant un historique des jeux empruntés par les utilisateurs.

Concernant l’**alimentation** de celle-ci, les informations des utilisateurs sont ajoutées lors de leur inscription à l'application, où ils fournissent leurs informations personnelles et choisissent un pseudonyme et un mot de passe. Les détails des jeux de société sont entrés manuellement dans la base de données par les administrateurs de l'application, qui ajoutent les nouveaux jeux disponibles dans la bibliothèque. Les données sur les locations sont enregistrées automatiquement chaque fois qu'un utilisateur effectue une réservation de jeu.

L’utilisateur a donc la possibilité d’**extraire** plusieurs informations, en consultant leur propre historique de locations, en lisant les commentaires et avis laissés par d'autres utilisateurs… Les administrateurs peuvent également utiliser les données de la base de données pour gérer les comptes des utilisateurs, suivre les tendances de popularité des jeux, optimiser l'inventaire de la bibliothèque et prendre des décisions stratégiques pour améliorer l'expérience des utilisateurs. Cependant, nous nous concentrerons uniquement sur l'expérience de l'utilisateur lorsqu'il utilise notre application Ludociel.

# LES LIMITES

Cependant, notre base de données comporte des limites, des aspects que nous n'avons pas couverts dans la version actuelle de notre application. Il aurait été intéressant d'explorer la possibilité de proposer la réservation de salles de jeux au sein de notre bibliothèque. Les utilisateurs auraient ainsi accès à tous les jeux de société disponibles sur place, leur permettant de réserver des espaces pour jouer entre amis, en famille ou même avec des inconnus De plus, nous avons choisi de ne pas intégrer la gestion des stocks de jeux disponibles, supposant ainsi une disponibilité illimitée des jeux. De même, nous n'avons pas pris en compte les délais de livraison et les problèmes qui peuvent en découler, tels que les retards de livraison ou d'autres incidents liés à la logistique. En conclusion, nous avons limité le nombre de jeux de société qu'un utilisateur peut commander lors d'une seule location, il ne peut commander qu'un seul jeu de société à la fois.

Nous aurions également souhaité intégrer la possibilité de prendre en compte la date de retour réelle des jeux de société rendus par l'utilisateur. Cette fonctionnalité aurait permis d'appliquer notre politique axée sur le respect des règles, notamment en cas de retard dans le retour d'un jeu, où des frais de retard de 50 centimes par jour auraient été facturés. Cette mesure était conçue pour encourager le respect des délais de retour, assurant ainsi le bon fonctionnement de notre bibliothèque. Nous avions prévu la mise en place d'un système de suivi des commandes/locations avec différents statuts tels que "en cours", "terminé" ou "en retard". Cela aurait permis aux utilisateurs de suivre l'évolution de leurs commandes et de planifier leur retour en conséquence. En outre, nous n'avons pas non plus pris en considération les dépôts de garantie que les utilisateurs doivent fournir avant chaque location de jeu, afin de garantir son bon état à la réception.

En conclusion, nous reconnaissons qu'une importante lacune existe dans notre application, à savoir la possibilité pour chaque utilisateur d'utiliser le compte d'un autre pour passer des commandes ou même visualiser les mots de passe des autres comptes. En d'autres termes, notre application n'est pas sécurisée, mais nous comptons sur la vigilance de nos utilisateurs et sur leur conscience morale pour respecter la confidentialité des autres utilisateurs.

LE DICTIONNAIRE DES VARIABLES

2.

## Le dictionnaire des variables

|  |  |  |  |
| --- | --- | --- | --- |
| Nom de la variable | Type | Taille | Remarques |
| Pseudo | Texte | 30 | Pseudo unique |
| Nom | Texte | 30 | Format majuscule |
| Prénom | Texte | 30 |  |
| Civilité | Texte | 15 | Liste choix : Monsieur, Madame, Autre |
| Mot de passe | Texte | 15 | Masque de saisie : mot de passe |
| Date de naissance | Date |  | Masque de saisie : date |
| Numéro de téléphone | Texte | 10 | Masque de saisie : numéro de téléphone |
| Mail | Texte | 40 | @ obligatoire (valide si…) |
| Rue | Texte | 100 |  |
| Ville | Texte | 30 | Format majuscule |
| Code postal | Texte | 5 | Masque de saisie : code postal |
| Date d’emprunt | Date |  | Masque de saisie : date |
| Durée du prêt | Texte |  | Liste déroulante : 2 jours, 5 jours, 1 semaine, 2 semaines |
| Nombre de joueurs | Numérique |  | Entre 1 et 15 joueurs (nombre entier) |
| Temps approximatif | Numérique |  | Liste déroulante : Variable, 5 à 10 minutes, 15 à 30 minutes, 30 à 45 minutes, 45 minutes à 1 heure, Plus de 1 heure |
| Complexité | Texte | 15 | Liste déroulante : Easy, medium, hard |
| Note sur 5 | Numérique |  | Entre 1 et 5 (nombre entier) |
| Commentaire | Texte | 200 | 200 caractères maximum |
| Catégorie | Texte | 20 | Liste déroulante : Plateau, coopération, culture, énigme, rôle, stratégie, ambiance **(choix multiple)** |
| Date avis | Date |  |  |
| Nom du jeu | Texte | 100 |  |

LA CONSTRUCTION DE LA BASE

3.

## La construction de la base

# LES ENTITES :

Pour concevoir la base de données de Ludociel, notre bibliothèque de jeux de société, nous identifions plusieurs entités principales et déterminons leurs attributs.

Pour l'entité **« Utilisateur »**, nous spécifions le pseudo, nom, prénom, civilité, mot de passe, date de naissance, numéro de téléphone, adresse e-mail.

L'entité **« Jeu de société »** inclut le nom du jeu, le nombre de joueurs, la durée approximative, la complexité.

L’entité **« Adresse »** comprend la rue, la ville ainsi que le code postal de la résidence de l’utilisateur.

Enfin, en ce qui concerne l’entité **« Avis »**, elle intègre les commentaires laissés par les utilisateurs, ainsi que les notes attribuées sur une échelle de 1 à 5.

Enfin, l'entité **« Location »** (qui est une table de jointure) est définie par la date d'emprunt, la durée du prêt.

# LES RELATIONS :

*Les relations principales*

* **Un** utilisateur peut avoir **une** seule adresse, et **une** adresse peut être utilisé pour **plusieurs** comptes (car on prend en compte une famille vivant sous le même toit)
* **Un** jeu de société peut inclure **plusieurs** avis, et **un** avis peut être inclus par **un** seul jeu de société
* **Un** utilisateur peut laisser **plusieurs** avis, et **un** avis peut être laissé que par **un** utilisateur
* **Un** jeu de société peut être louer par **plusieurs** utilisateurs, et **un** utilisateur peut louer **plusieurs** jeux de société

*Les relations non principales*

* **Un** utilisateur peut effectuer **plusieurs** locations, et **une** location peut être effectuer par **un** seul utilisateur
* **Une** location peut contenir **un** jeu de société, et **un** jeu de société peut être contenu dans **plusieurs** location

# LES CLES PRIMAIRES *de chacune des entités* :

* **Utilisateur :** Pseudo (unique)
* **Jeu de société :** ID\_Jeu (numéro automatique)
* **Adresse :** ID\_Adresse (numéro automatique)
* **Avis :** ID\_Avis (numéro automatique)
* **Location:** ID\_Location (numéro automatique)

# Une image contenant texte, diagramme, capture d’écran, Parallèle Description générée automatiquementMODELE MCL :

*Les cardinalités entourées sont celles que nous retenons, correspondant aux valeurs les plus élevées à chaque extrémité.*

# MODELE MCD :

Une image contenant texte, diagramme, capture d’écran, ligne

Description générée automatiquement

*La table location est donc une table de jointure permettant de lié la table Utilisateur ainis que Jeu de Société.*

LE FONCTIONNEMENT DE L’APPLICATION

4.

## Le fonctionnement de l’application

# LES FORMULAIRES

**F\_Inscription :**

Ce formulaire facilite l'inscription de l'utilisateur en collectant ses informations personnelles. Au démarrage, l'utilisateur est invité à saisir en premier lieu ses coordonnées postales, essentielles pour la livraison des jeux de société. Cette approche nous permet de commencer à organiser la livraison dès que possible. Une fois les informations relatives à l'adresse entrées, l'utilisateur peut cliquer sur le bouton "Suivant" pour enregistrer ces données dans la table Adresse, ce qui rend visibles les autres champs tels que le pseudo, le nom, etc. (code VBA associé)

Une image contenant texte, capture d’écran, document

Description générée automatiquementLorsque le bouton "Suivant" est sélectionné, un code VBA est déclenché pour rechercher l'ID\_Adresse correspondant aux informations nouvellement saisies. Ainsi, la première étape consiste à ouvrir la requête d'ajout, ce qui permet de l'activer.

Ce processus garantit la cohérence des données dans la base de données. Tout d'abord, une vérification est effectuée pour s'assurer que les champs de texte ne sont pas vides, bien que cela soit peu probable puisque l'utilisateur vient de les remplir (c’est une simple précaution). Ensuite, une requête est construite pour sélectionner l'ID\_Adresse à partir de la table "T\_Adresse" où la rue et la ville correspondent aux valeurs saisies dans les champs de texte respectifs. Cette requête recherche une correspondance exacte dans la table "T\_Adresse" en fonction des valeurs saisies par l'utilisateur.

La base de données est ouverte et un objet Recordset est créé à partir de cette requête SQL, permettant l'exécution de la requête et la récupération des résultats dans un format manipulable. Si une correspondance est trouvée, l'ID de l'adresse est affiché dans le champ de texte "txtIDAdresse". En cas d'absence de correspondance, un message d'erreur est affiché pour informer l'utilisateur. Une fois que les opérations sur le Recordset et la base de données sont terminées, les objets sont libérés de la mémoire en attribuant la valeur "Nothing" à ces objets, assurant ainsi une gestion efficace des ressources système. Après l'exécution de cette section de code, une autre partie du processus est mise en place pour gérer la visibilité des différents champs.

Enfin, l'utilisateur remplit ensuite toutes les informations restant et requises, puis il clique sur ce bouton pour finaliser son inscription. Ce bouton est lié à un code VBA qui ouvre simplement la requête "R\_AjoutUtilisateur". De plus, il ferme également le formulaire actuel et ouvre le formulaire central "F\_Menu", permettant ainsi à l'utilisateur d'accéder à l'ensemble des fonctionnalités de l'application.

Une image contenant texte, capture d’écran, Police, ligne

Description générée automatiquement

**F\_Connexion :**

Ce formulaire permet à ceux qui ont déjà un compte Ludociel de se connecter simplement en renseignant leur pseudo puis en inscrivant leur mot de passe. Ce formulaire contient deux zones de texte indépendantes (une pour le pseudo et une autre pour le mot de passe). De plus, il y a un bouton « Se connecter » et un autre « Quitter » ; le bouton Quitter permet simplement de quitter la fenêtre, tandis que le bouton Se connecter contient une macro bien spécifique pour vérifier si le pseudo renseigné ainsi que le mot de passe correspondent bien à un compte.

La macro contient une fonction SI étant :

**EstNull(RechDom("Pseudo";"T\_Utilisateur";"Pseudo='" & [txtPseudo] & "' and MDP\_Utilisateur='" & [txtMDP] & "'"))**

La fonction EstNull permet de vérifier si une expression donnée est nulle. Si l'expression est nulle, la fonction retourne Vrai ; sinon, elle retourne Faux. Ensuite, la fonction RechDom effectue une recherche dans la table T\_Utilisateur en fonction de certains critères. Dans un premier temps, « Pseudo » est le nom du champ dans lequel la recherche est effectuée, suivi de la table dans laquelle se trouve ce pseudo. Ensuite, « Pseudo='" & [txtPseudo] & "' and MDP\_Utilisateur='" & [txtMDP] & "'" est le critère de recherche. Il recherche un enregistrement où les champs indépendants du formulaire sont égaux à une ligne de la table. Enfin, pour récapituler, EstNull permet de vérifier si le résultat de la fonction RechDom est nul ou non. Si le résultat est nul, cela signifie que la recherche n'a pas renvoyé de correspondance, et donc l'expression dans son ensemble évaluera Vrai, indiquant que l'utilisateur ou le mot de passe n'existe pas dans la base de données. Dans ce cas un message d’erreur s’affiche, sinon, la fenêtre F\_Connexion se ferme, et l’utilisateur accède directement au F\_Menu.

*Remarque : Cette macro fonctionne correctement lorsque le mot de passe et le pseudo sont corrects. Cependant, si l'un des deux est incorrect, un message d'erreur d’Access s'affiche.*

**F\_MDP\_Oublie :**

Nous laissons la possibilité à l’utilisateur s’il a oublié son mot de passe de le visionner pendant 5 secondes seulement en sélectionnant son pseudo, puis le mot de passe s’affichera. Dans le formulaire, il y a 2 champs qui sont des champs indépendants, ainsi qu’un champs MDP\_Utilisateur. Nous avons donc lié le champs indépendant correspondant à une liste de Pseudo (étant alimenter par une requête permettant de rechercher le pseudo dans la table pseudo) et le champs MDP\_utilisateur. Lorsqu'un pseudonyme est sélectionné dans la liste déroulante, le mot de passe correspondant s'affiche dans la zone prévue à cet effet (MDP\_Utilisateur). Ce mot de passe peut ainsi être copié dans un champ où il sera visible contrairement au MDP\_Utilisateur. De plus, à chaque changement de pseudonyme, le mot de passe associé est également mis à jour.

Nous avons également utilisé VBA pour permettre l’affichage temporaire du mot de passe. Lorsque l’on clique sur afficher le mot de passe le code ci-dessous se lance.

Une image contenant texte, capture d’écran, Police, nombre

Description générée automatiquement

« Start = Timer » enregistre le temps actuel dans la variable « Start ». Ensuite, une boucle s'exécute jusqu'à ce que le temps écoulé soit supérieur ou égal à 5 secondes de plus que le temps enregistré au début de la boucle. À chaque itération de la boucle, la ligne « Check = Timer » met à jour la variable Check avec le temps actuel. Enfin, l'instruction « DoEvents » permet de traiter les événements en attente pendant que la boucle est en cours d'exécution, ce qui peut empêcher l'application de sembler gelée. Une fois cette boucle terminée, on vide les champs MDP\_Affiche, ainsi que lstPseudo.

**F\_EnregistrerLocation :**

Ce formulaire permet à l'utilisateur d'effectuer une location. Pour ce faire, il doit simplement sélectionner son pseudo dans une liste déroulante (obtenue grâce à une requête permettant d'extraire tous les pseudos de la table), puis choisir la durée de sa location (également dans une liste déroulante) et le nom du jeu qu'il souhaite louer.

En fonction de la durée qu'il aura choisie, le formulaire affiche le prix qu'il devra payer (à titre informatif, cela ne sera pas enregistré dans une table), ainsi que dans une autre zone de texte la date de retour prévue (également à titre informatif). Pour ces deux cas, nous avons utilisé une fonction SI dans la zone de contrôle de la zone de texte indépendante.

* Pour le prix qu'il devra payer :

**=SI([lstDuree]="2 jours";1,5;SI([lstDuree]="5 jours";2;SI([lstDuree]="1 semaine";2,5;SI([lstDuree]="2 semaines";3;0))))**

* Pour la date de retour prévue :

**=SI([lstDuree]="2 jours";[txtDate]+2;SI([lstDuree]="5 jours";[txtDate]+5;SI([lstDuree]="1 semaine";[txtDate]+7;SI([lstDuree]="2 semaines";[txtDate]+14;0))))**

De plus, lorsqu'il choisit le nom du jeu qu'il souhaite louer dans une liste déroulante (dont le contenu provient d'une requête qui extrait les noms de jeu), une autre zone de texte indépendante se met à jour à chaque fois que la liste déroulante du nom du jeu est modifiée, affichant ainsi l'identifiant du jeu. Cela permet donc d'enregistrer dans la table l'ID du jeu.

Enfin, une zone de texte indépendante permet d'afficher la date d'emprunt dans ce formulaire, où sa source de contrôle est la date d'aujourd'hui. En effet, nous considérons que lorsque l'utilisateur effectue sa commande pour sa location, sa date d'emprunt est la date à laquelle il remplit ce formulaire.

Avant de rattacher ce formulaire à une requête d'action d'ajout, nous avons bien fait attention de nommer correctement tous nos objets indépendants dans ce formulaire pour faciliter la requête d'action. Ensuite, nous avons ajouté un bouton "Valider ma location". Lorsque l'utilisateur clique dessus, la requête d'action [R\_AjoutLocation](#_LES_REQUETES_ET) *(cf l’explication de la requête)* , qui est créée spécifiquement pour ce formulaire, s'exécute. Ce formulaire alimente donc la table Location avec les informations qui y sont contenues.

**F\_Filtres:**

Ce formulaire offre aux utilisateurs la possibilité de consulter les jeux de société afin de leur permettre de choisir des jeux qui leur plaisent potentiellement pour une location. En effet, l'utilisateur peut consulter les jeux de société avec toutes leurs caractéristiques correspondantes telles que la catégorie, le nombre de joueurs, le temps approximatif, la complexité de compréhension et enfin la note moyenne émise par certains utilisateurs sur un jeu de société. Toutes ces informations lui permettront de mieux choisir le jeu de société correspondant à ses préférences en filtrant simplement ces informations en fonction de ses souhaits. Lorsqu'il filtre grâce à une liste déroulante ou une zone de texte, cela modifie le sous-formulaire qui est en mode feuille de données. Pour ce faire, une requête est attachée au sous-formulaire, la requête étant [R\_Filtres](#_LES_REQUETES_ET) (voir l'explication de la requête). Chaque caractéristique est représentée soit par une liste déroulante, soit par une zone de texte (comme pour le nombre de joueurs). À chacun de ces quatre filtres, nous avons attribué un code VBA qui actualise automatiquement les résultats dès que l'utilisateur met à jour l'un de ces objets indépendants. Cela est réalisé en utilisant la commande DoCmd.Requery. De plus, pour le champ txtJoueur, nous avons configuré une valeur par défaut à l'aide d'un code VBA. Cela permet un filtrage continu en fonction du nombre maximal de joueurs autorisé pour un jeu donné. Cette valeur par défaut reste active à moins d'être modifiée par l'utilisateur. En outre, nous avons inclus un bouton qui permet de réinitialiser tous les filtres à leur état initial grâce à un code VBA. Cela signifie que tous les objets indépendants redeviennent vides, à l'exception du champ txtJoueur qui reprend sa valeur par défaut.

**F\_Commentaire :**

Ce formulaire permet de filtrer tous les commentaires ainsi que chacune des notes émises pour chaque nom de jeu. L'utilisateur sélectionne simplement dans la liste déroulante le nom du jeu pour lequel il souhaite visualiser tous les avis. Une zone de texte indépendante est liée à cette liste, affichant ainsi l'IDJeu. Cet identifiant est utilisé pour filtrer les données dans le sous-formulaire provenant d'une requête, celle-ci contient les informations tel que le pseudo, la date à laquelle l’utilisateur à déposer son avis ainsi que les commentaires, et les notes qu’il a attribué au jeu de société. La requête attachée à ce formulaire est [R\_Commentaire](#_LES_REQUETES_ET).

De plus, nous avons voulu dans une zone de texte calculé la note moyenne qu’obtient un jeu de société, pour ce faire nous avons utilisé la fonction « MoyDom » :

=MoyDom("[T\_Avis]Note\_Avis]";"[T\_Avis]";"[T\_Avis]![ID\_Jeu#] =[txtIDJeu]")

Cette zone de texte s’actualise donc en fonction du nom de jeu que l’on a choisi.

**F\_Avis :**

Ce formulaire permet à l'utilisateur d'ajouter un avis sur un jeu de société spécifique. Pour ce faire, il sélectionne son pseudo dans une liste déroulante (dont la source provient d'une requête permettant d'extraire tous les pseudos). Ensuite, il choisit le nom du jeu de société sur lequel il souhaite laisser un avis. Il sélectionne ensuite une note qu'il souhaite attribuer grâce à une liste déroulante, offrant les choix de 1 à 5. Enfin, il a la possibilité d'écrire son commentaire dans une zone de texte indépendante. Nous avons également ajouté une zone de texte indépendante, dont la source est la date d'aujourd'hui (la date à laquelle l'avis est laissé). Nous avons pris soin de nommer tous nos objets pour faciliter la démarche de la requête d'ajout. Pour terminer, le bouton "Ajouter mon avis" permet, lorsqu'il est cliqué, de lancer la requête R\_AjoutAvis, alimentant ainsi la table des avis.

*Remarque : Nous aurions préféré que seuls les utilisateurs ayant déjà loué ce jeu laissent leurs avis sur le jeu en question.*

**F\_Menu :**

Ce formulaire centralise et permet l'accès à la plupart des autres formulaires. Il se compose uniquement de boutons permettant d'accéder à différents formulaires spécifiques.

**F\_Historique :**

Ce formulaire se compose principalement d'une liste déroulante affichant les pseudos des utilisateurs. Lorsque l'utilisateur sélectionne un pseudo dans cette liste, le sous-formulaire (R\_Historique) est filtré pour afficher uniquement les informations relatives aux locations effectuées par cet utilisateur. De plus, un bouton "Imprimer mon historique" est disponible, permettant à l'utilisateur de générer un état (E\_HistoriqueLoc) récapitulant son historique de location.

***Remarque :***

*Nous avons omis d'imposer dans nos formulaires d'ajout la nécessité de remplir certains champs obligatoires. Par exemple, un utilisateur peut laisser un avis sans inclure de note ou de commentaire. Ce problème aurait pu être résolu facilement par l'ajout d'un code VBA pour la validation des champs requis.*

# LA NAVIGATION ENTRE LES FORMULAIRES

Au démarrage de notre application, l'utilisateur sera dirigé vers une page d'accueil (F\_Inscription), qui servira de point de départ. Cette interface inclura également un bouton pour se connecter, lequel ouvrira le formulaire F\_Connexion. Depuis ce formulaire, l'utilisateur aura la possibilité de cliquer sur "mot de passe oublié", lui permettant ainsi d'accéder à F\_MDP\_Oublie pour réinitialiser son mot de passe. Une fois le compte créé ou la connexion établie, l'utilisateur sera dirigé directement vers le formulaire centralisé, F\_Menu. À partir de là, quatre choix s'offriront à lui : il pourra commander un jeu (F\_EnregistrerLocation), parcourir la liste complète des jeux disponibles (F\_Filtres), consulter les avis des utilisateurs et en ajouter sur les jeux (F\_Avis), ainsi que visualiser son historique de location, lui permettant même de générer un état récapitulatif (E\_HistoriqueLoc).

![Une image contenant texte, diagramme, Police, capture d’écran

Description générée automatiquement

# LES REQUETES ET LES ETATS PROPOSES

**R\_AjoutAdresse :**

Cette requête permet d'ajouter des informations complémentaires dans le formulaire F\_Inscription. Nous avons spécifié dans quelle table et dans quels champs nous voulions enregistrer ces informations. En l'occurrence, dans cette requête, nous souhaitons enregistrer la rue, l'adresse et le code postal dans la table T\_Adresse. Ensuite, nous avons complété les champs à l'aide du générateur d'expression en sélectionnant l'objet correspondant à l'emplacement où nous voulions ajouter cette information. Pour illustrer cela, prenons un exemple : si nous voulons ajouter l'information de la rue dans la table Adresse, nous avons renseigné le champ comme suit : [**Formulaires]![F\_Inscription]![txtRue]**

**R\_AjoutUtilisateur/ R\_AjoutAvis / R\_AjoutLocation :**

Nous avons également répété ce procédé pour ces autres requêtes d’ajout.

**R\_Commentaire :**

Cette requête permet de créer un sous-formulaire dans le formulaire F\_Commentaire, qui peut être filtré par l'identifiant du jeu de société, non affiché dans la feuille de données. Pour filtrer par l'ID\_Jeu#, nous avons utilisé le critère **= [Forms]![F\_Commentaire]![txtIDJeu]** dans la zone critère de la colonne ID\_Jeu#. Ensuite, nous avons intégré cette requête dans le formulaire F\_Commentaire pour créer un sous-formulaire. Lorsque l'utilisateur choisit le numéro du jeu, le filtrage se met automatiquement à jour à chaque actualisation de la liste déroulante, grâce à un code VBA simple, **DoCmd.Requery**, pour rafraîchir les données. Nous avons délibérément omis d'inclure un bouton de réinitialisation du filtre, car le champ du nom du jeu n'étant pas inclus dans le sous-formulaire, il serait peu pertinent de laisser une liste d'avis sans savoir à quel jeu ils appartiennent.

**R\_ComplexiteListe :**

Cette requête est utilisée uniquement pour ajouter des options à une liste déroulante d'un élément indépendant dans un formulaire.

**R\_Filtres :**

Ce sous-formulaire fonctionne de manière similaire à R\_Commentaire, mais il est conçu pour être utilisé sans F\_Filtres. Au lieu de cela, il peut être filtré en fonction des critères définis dans les champs souhaités pour le filtrage. Nous utilisons la même méthode que pour R\_Commentaire, mais avec un critère numérique différent pour filtrer le nombre de joueurs. Pour cela, nous utilisons la syntaxe suivante **: <= [Formulaire]![F\_Filtres]![txtJoueur]**. Cela permet à l'utilisateur de saisir un nombre entre 1 et 15, filtrant ainsi tous les jeux de société comportant moins ou le même nombre de joueurs que celui saisi. Une macro, nommée **AfficherTousEnreg**, est associée à chaque mise à jour de ces champs de filtrage, ce qui permet d'actualiser le filtre chaque fois qu'il est modifié.

**R\_Historique :**

Nous avons mis en place une requête d'historique conçue pour filtrer les informations relatives à la location d'un jeu de société en utilisant le pseudo de l'utilisateur (le fonctionnement est similaire à celui de R\_Commentaire). De plus, nous avons ajouté un champ que nous avons créé à l'aide du générateur de requêtes, permettant d'afficher le prix payé en fonction de la durée de l'emprunt. La formule pour calculer ce prix est la suivante : **Prix: VraiFaux([Duree\_Emprunt]="2 jours";1,5;VraiFaux([Duree\_Emprunt]="5 jours";2;VraiFaux([Duree\_Emprunt]="1 semaine";2,5;VraiFaux([Duree\_Emprunt]="2 semaines";3;0))))**

Cette formule calcule le prix en fonction de la durée de l'emprunt :

- Si la durée est de 2 jours, le prix est de 1.5€.

- Si la durée est de 5 jours, le prix est de 2€.

- Si la durée est de 7 jours, le prix est de 2.5€.

- Sinon le prix est de 3€.

**E\_HistoriqueLoc :**

Nous avons mis en place un état lié à un formulaire grâce au bouton "Imprimer" situé dans le formulaire F\_Historique. Pour créer cet état, nous avons sélectionné notre requête R\_Historique (la même que celle utilisée pour le formulaire, ce qui signifie que si elle est filtrée dans le formulaire, elle le sera également pour cet état). Nous avons choisi d'afficher les données par la table T\_Location. De plus, nous avons décidé d'ajouter un niveau de regroupement par le pseudo. Ensuite, nous avons décidé de trier nos enregistrements dans l'ordre décroissant des dates d'emprunt.

Une fois l'état créé, nous avons simplement ajouté un pied de groupe "Pseudo" où nous calculons la somme totale des locations de jeux pour chaque utilisateur. Pour ce faire, nous avons utilisé un objet indépendant dont la source de valeur est définie par une fonction : **= Somme([Prix]).**

# LES PLUS :

Cette section rappelle seulement des éléments déjà expliquer mais met en lumière les éléments que nous considérons comme des « plus ».

* Dans un premier temps, nous avons choisi d’effectuer un code VBA pour permettre à l’utilisateur de visionner le **mot de passe** attaché à son pseudo s’il l’a **oublié.**
* Nous avons également choisi d’effectuer une **connexion** grâce au pseudo et au mot de passe que l’utilisateur renseigne.
* Lorsqu'une **requête d'action** est effectuée, nous avons décidé de **supprimer les messages d'Access pompeux** afin de faciliter la lisibilité de notre application et de moins perturber ses utilisateurs. Pour se faire, nous avons désélectionner les avertissements actifs dans la macro du bouton.
* Une macro a été créée pour afficher un **message-box** lors de **l'ouverture de l'application.**

LES APPORTS DU PROJET

5.

## Les apports du projet

Ce projet a été incroyablement enrichissant à plusieurs niveaux. Tout d'abord, il nous a permis de saisir la puissance et l'importance d'une base de données, allant bien au-delà de la simple manipulation d’un jeu de données que nous avions l'habitude de faire. En explorant les fonctionnalités complexes de la gestion des données, nous avons réalisé toute la richesse cachée derrière les informations que nous côtoyons quotidiennement. Ce projet nous a également sensibilisés à l'importance critique des données dans divers contextes. En effet, nous avons compris leur rôle central dans la prise de décisions éclairées, la gestion efficace des ressources et la création de solutions innovantes. De plus, en travaillant sur un projet concret avec Microsoft Access, nous avons acquis des compétences pratiques précieuses en matière de conception de bases de données, de requêtes SQL et de gestion des relations entre les données. En fin de compte, ce projet nous a offert une perspective nouvelle et approfondie sur le monde des données et nous a dotés d'outils essentiels pour naviguer dans un environnement de plus en plus axé sur l'information et la technologie. Enfin, nous avons eu l'opportunité de mettre en pratique nos connaissances acquises en VBA lors du semestre précédent.

Deuxièmement, comme de nombreux travaux que nous avons effectués cette année, ce projet nous a offert l'occasion de coopérer et de réfléchir collectivement autour d'un objectif commun. Nous sommes tous deux conscients que cette compétence est primordiale pour nos expériences futures, en particulier sur le plan professionnel.

Enfin, nous avons dû faire preuve de rigueur et de vigilance à chaque étape de notre travail, afin d'éviter des erreurs qui pourraient nous bloquer dans l'avancement du projet. Ce dernier s'est rapproché de notre projet du semestre précédent, car nous avons ressenti les mêmes satisfactions lorsque quelque chose fonctionnait. De plus, l'idée que nous pouvons toujours améliorer notre application et trouver de nouvelles utilisations pour notre base de données est très stimulante, bien que parfois frustrante lorsque nous devons nous arrêter. La découverte d'Access ainsi que la création d'une base de données ont été très positives pour nous, car cela nous apporte des compétences dont nous aurons besoin dans la suite de nos études.

CONCLUSION

6.

## Conclusion

En conclusion, Ludociel offre une solution pratique et novatrice pour les amateurs de jeux de société. En transformant le concept traditionnel de bibliothèque en une plateforme en ligne, Ludociel permet aux utilisateurs de découvrir, louer et partager une large gamme de jeux facilement et à moindre coût.

Cette application simplifie l'accès aux jeux de société, offrant une alternative économique à l'achat de jeux individuels. En permettant aux utilisateurs de consulter les avis et les commentaires, Ludociel favorise également une communauté d'échange et de partage autour des jeux, enrichissant ainsi l'expérience de jeu de chacun.

Les formulaires de Ludociel ont été conçus dans un but de simplicité et d'efficacité. Leur objectif est de faciliter la navigation et l'interaction des utilisateurs avec l'application. Que ce soit pour s'inscrire, se connecter, effectuer une location ou filtrer les jeux selon leurs préférences, chaque formulaire a été pensé pour offrir une expérience fluide et intuitive. Ces interfaces permettent aux utilisateurs de naviguer aisément dans l'application, de trouver rapidement les informations recherchées et de profiter pleinement de leur expérience.

Ludociel représente donc bien plus qu'une simple bibliothèque de jeux. C'est un espace numérique où la passion pour les jeux de société se rencontre avec la technologie moderne, créant ainsi une plateforme inclusive et accessible à tous les amateurs de jeux, qu'ils soient novices ou experts. En somme, Ludociel rend les jeux de société plus accessibles, interactifs et enrichissants pour tous.

In conclusion, Ludociel offers a practical and innovative solution for board game enthusiasts. By transforming the traditional concept of a library into an online platform, Ludociel allows users to discover, rent, and share a wide range of games easily and affordably.

This application simplifies access to board games, providing an economical alternative to purchasing individual games. By enabling users to browse reviews and comments, Ludociel also fosters a community of exchange and sharing around games, enriching each player's gaming experience.

Ludociel's forms have been designed for simplicity and efficiency. Their aim is to facilitate user navigation and interaction with the application. Whether it's for signing up, logging in, renting a game, or filtering games according to preferences, each form has been designed to offer a smooth and intuitive experience. These interfaces allow users to navigate the application easily, quickly find the information they're looking for, and fully enjoy their experience.

Ludociel represents more than just a game library. It's a digital space where the passion for board games meets modern technology, creating an inclusive platform accessible to all game enthusiasts, whether they're beginners or experts. In summary, Ludociel makes board games more accessible, interactive, and enriching for everyone.