Rapport Projet BDD (SkinVault)

INTRODUCTION

Dans le cadre du module de **Bases de Données (BDD)**, ce projet a pour objectif la conception et la réalisation d'une application exploitant une base de données relationnelle dans un contexte concret. Afin de répondre à cette problématique, il a été choisi de développer une **application web** permettant de présenter des **skins du jeu Counter- Strike: Global Offensive (CS:GO)** ainsi que leurs **prix actuels sur le marché**, en s'inspirant du site de référence *csgodatabase.com*.

L'application développée prend la forme d'un **site web vitrine**, dont le rôle principal est l'affichage structuré des données issues de la base, tout en proposant des fonctionnalités simples de consultation et de gestion des préférences utilisateur. Ce choix permet de mettre en pratique les notions fondamentales du cours de BDD, telles que la **modélisation relationnelle**, la **création de tables**, la **gestion des relations** et l'**exploitation des données** au sein d'une application réelle.
Pour la gestion des données, le système de gestion de base de données **PostgreSQL** a été utilisé. Les données exploitées proviennent de deux sources distinctes : une base issue de la plateforme **Kaggle** et une autre récupérée depuis un **dépôt Git**. Ces deux bases ont été analysées, nettoyées et fusionnées afin de constituer une base de données unique et cohérente, adaptée aux besoins de l'application. La base finale repose sur une architecture

composée de **cinq tables principales** : user,

`user_favorite`, `weapon`, `weapon_category` et `skins`, auxquelles s'ajoute la table `auth_user` fournie par le framework Django pour la gestion de l'authentification. Cette organisation permet de structurer efficacement les données tout en assurant la cohérence et l'intégrité des relations entre les différentes entités.

Le développement de l'application a été réalisé à l'aide du langage **Python**, du framework web **Django** pour la partie serveur, ainsi que des technologies **HTML** et **CSS** pour l'interface utilisateur. Des réglages spécifiques de Django ont été appliqués afin d'obtenir un site volontairement **minimaliste**, mettant l'accent sur la lisibilité des informations et l'exploitation de la base de données plutôt que sur l'aspect graphique.

Ce rapport présente les différentes étapes de réalisation du projet, depuis la conception de la base de données jusqu'à l'implémentation de l'application web, en mettant en évidence les choix techniques effectués et les solutions apportées aux problématiques rencontrées.

Conception et modélisation de la base de données

Laconception de la basededonnées constitueune étape centrale du projet, carelle conditionne à la fois la cohérence des données, les performances de l'application et la simplicité de son exploitation. Avant toute implémentation, une réflexion a été menée afin d'identifier les entités principales du domaine étudié ainsi que les relations existant entre elles.

Le domaine de l'application repose principalement sur les concepts suivants :

- les armes présentes dans le jeu CS:GO,
- les catégories d'armes,
- les skins associés à chaque arme,
- les utilisateurs,
- les skins favoris des utilisateurs.

L'objectif principal de la modélisation était d'éviter toute redondance inutile, de respecter les règles de normalisation et de garantir l'intégrité référentielle entre les différentes tables.

Schéma relationnel et description détaillée des tables

Table `weapon_category`

La table `weapon_category` permet de regrouper les armes par type. Chaque catégorie est unique et identifiée par une clé primaire.

Champs principaux :

- `id` : identifiant unique de la catégorie (clé primaire, auto-incrémentée)
- `name` : nom de la catégorie (unique et non nul)

Cette table facilite l'organisation des armes et permet un affichage structuré sur le site web.

Table `weapon`

La table `weapon` contient les informations relatives aux armes disponibles dans le jeu. Chaque arme appartient obligatoirement à une catégorie.

Champs principaux :

- `id` : identifiant unique de l'arme `name` : nom de l'arme (unique)
- `category_id` : clé étrangère faisant référence à `weapon_category(id)`
-

Table `skin`

Latable `skin` est l'une des tables centrales du projet. Elle stocke l'ensemble des informations relatives aux skins associés aux armes.

Champs principaux :

- `id` : identifiant unique du skin
- `name` : nom du skin
- `weapon_id` : clé étrangère vers la table `weapon`
- `min_price` : prix minimum observé sur le marché
- `max_price` : prix maximum observé sur le marché
- `image_url` : lien vers l'image du skin

Une contrainte d'unicité est appliquée sur le couple `(weapon_id, name)` afin d'éviter les doublons logiques tout en permettant à plusieurs armes de posséder des skins portant le même nom.

Table `Auth_user`

La table`auth_user` estfournie automatiquement par Django. Elle gère l'authentification des utilisateurs et contient notamment :

- le nom d'utilisateur,
- le mot de passe chiffré,
- les permissions,
- les informations liées aux sessions.

Cette table garantit un système d'authentification sécurisé et conforme aux standards du framework.

Table `user_favorite`

La table `user_favorite` permet de gérer les skins favoris des utilisateurs. Elle implémente une relation entre les utilisateurs et les skins.

Champs principaux :

- `user_id` : identifiant de l'utilisateur
- `skin_id` : identifiant du skin

La clé primaire est composée du couple `(user_id, skin_id)`, ce qui empêche un utilisateur d'ajouter plusieurs fois le même skin en favori.

Collecte, fusion et enrichissement des données

Les données initiales utilisées pour ce projet provenaient de deux sources distinctes : une base de données issue de la plateforme Kaggle et une autre récupérée depuis un dépôt Git. Bien que ces bases contenaient des informations pertinentes sur les skins et les armes, elles présentaient plusieurs limitations importantes.

Tout d'abord, les bases ne contenaient pas les images des skins. De plus, les prix disponibles étaient souvent exprimés sous forme d'intervalles approximatifs, sans valeur actuelle précise. Ces limitations rendaient les données insuffisantes pour une application web vitrine destinée à présenter des informations claires et visuelles.
Afin de pallier ces manques, un outil de scraping nommé **Thunderbit** a été utilisé. Cet outil permet d'extraire automatiquement des données depuis des pages web et de les exporter

sous forme de fichiers `.csv`. Grâce à Thunderbit, il a été possible de récupérer :

- les images des skins,
- les prix actuels observés sur le marché.

Les fichiers obtenus ont ensuite été traités à l'aide de scripts Python afin d'être nettoyés, structurés et intégrés à la base de données PostgreSQL.

Architecture de l'application web

L'application a été développée à l'aide du framework **Django**, qui repose sur une architecture **MTV (Model – Template – View)**. Cette architecture permet une séparation claire des responsabilités :

- Les **Models** correspondent aux tables de la base de données.
- Les **Views** contiennent la logique applicative et les interactions avec la base.
- Les **Templates** définissent l'interface utilisateur à l'aide de HTML et CSS.

L'utilisation de l'ORM de Django a permis de manipuler les données sans écrire directement de requêtes SQL complexes, tout en conservant de bonnes performances et une sécurité accrue.

Interface utilisateur et choix graphiques

L’interface utilisateur a été développée en **HTML** et **CSS**, avec un design volontairement minimaliste. Ce choix a été fait afin de mettre l’accent sur les données plutôt que sur l’aspect graphique.

Le site permet notamment :

- la navigation par catégorie d’armes,
- la consultation des armes disponibles,
- l’affichage des skins avec leur image et leur prix,
- l’ajout et la suppression de skins en favoris pour les utilisateurs connectés.

Problèmes rencontrés et solutions apportées

Absence d’images et de prix précis dans les bases initiales

L’un des principaux problèmes rencontrés concerne la qualité des données initiales. Les bases récupérées depuis Kaggle et Git ne contenaient pas les images des skins et proposaient uniquement des intervalles de prix, ce qui n’était pas suffisant pour une application vitrine.

Solution apportée :
L’utilisation de Thunderbit comme outil de scraping a permis d’extraire directement les images et les prix actuels depuis des pages web spécialisées. Les données ont ensuite été intégrées sous forme de fichiers CSV puis importées dans PostgreSQL après nettoyage.

Conflit lors de l’ajout de skins en favoris

Un problème important est apparu lors de l’implémentation de la fonctionnalité “ajouter aux favoris”. Certains skins possèdent le même nom mais sont associés à des armes différentes (par exemple *Black Laminate* pour un couteau et pour une AK-47). Lors de l’ajout en favori, le système sélectionnait systématiquement le premier skin correspondant au nom, ce qui provoquait des incohérences.

Solution apportée :
La logique a été corrigée en utilisant exclusivement l’**identifiant unique (id) du skin** plutôt que son nom. Cette approche garantit que le bon skin est ajouté aux favoris, indépendamment des doublons de noms.

Problème d’interface avec le bouton “Add favorite”

Un problème d’ergonomie est apparu au niveau de l’interface utilisateur. Une animation CSS de type hover appliquée aux skins provoquait des comportements inattendus lors du clic sur le bouton “Add favorite”, rendant l’interaction peu fiable.

Solution apportée :
L’animation hover a été supprimée afin de garantir une interaction stable et prévisible. Cette décision a permis d’améliorer la fiabilité de l’interface tout en conservant une présentation claire.

Conclusion générale

Ce projet a permis de concevoir et de développer une application web complète reposant sur une base de données relationnelle. Il a mobilisé des compétences en modélisation de bases de données, en traitement de données, en développement back-end avec Django et en création d’interfaces web simples et fonctionnelles.

Les différentes difficultés rencontrées ont permis de mieux comprendre les enjeux liés à la qualité des données, à l’identification unique des entités et à l’ergonomie d’une application web. Ce projet constitue ainsi une application concrète et formatrice des notions abordées dans le module de Bases de Données.

Table weapon_category

```
bakimidb=# select * from weapon_category;
 id |    name
----+------------
  1 | Knife
  2 | Rifle
  3 | SMG
  4 | Sniper
  5 | Pistol
  6 | Shotgun
  7 | Machine Gun
  8 | Agent
  9 | Sticker
(9 lignes)
```

```
CREATE TABLE weapon_category (
    id SERIAL PRIMARY KEY,
    name TEXT UNIQUE NOT NULL
);
```

Table weapon

```
bakimidb=# select * from weapon;
 id |     name     | category_id
----+--------------+-------------
  1 | Karambit     |           1
  2 | Butterfly    |           1
  3 | M9 Bayonet   |           1
  4 | Gut          |           1
  5 | Flip         |           1
  6 | Huntsman     |           1
  7 | AK-47        |           2
  8 | M4A4         |           2
  9 | M4A1-S       |           2
 10 | AUG          |           2
 11 | SG 553       |           2
 12 | FAMAS        |           2
 13 | MP9          |           3
 14 | UMP-45       |           3
 15 | MP7          |           3
 16 | P90          |           3
 17 | MAC-10       |           3
 18 | AWP          |           4
 19 | SSG 08       |           4
 20 | SCAR-20      |           4
 21 | G3SG1        |           4
 22 | Glock        |           5
 23 | USP-S        |           5
 24 | Desert Eagle |           5
 25 | P250         |           5
 26 | Five-SeveN   |           5
 27 | CZ75         |           5
 28 | Nova         |           6
 29 | XM1014       |           6
 30 | MAG-7        |           6
 31 | Sawed-Off    |           6
 32 | M249         |           7
 33 | Negev        |           7
 34 | Agents       |           8
 35 | All Stickers |           9
(35 lignes)
```

```
CREATE TABLE weapon (
    id SERIAL PRIMARY KEY,
    name TEXT UNIQUE NOT NULL,
    category_id INTEGER NOT NULL REFERENCES weapon_category(id)
);
```

Table skin

```
 id  | weapon_id |                    name                    | max_price |
e_url
-----+-----------+--------------------------------------------+-----------+----------------------------------------------------------------------------

   1 |         7 | Nightwish                                  |     145.6 | https://cdn.csgoskins.gg/public/uih/items/aHR0cHM6Ly9jZG4uY3Nnb3NraW5zLm
mlnaHR3aXNoX2xpZ2h0LjA4YTY1NjE1ZDdjZGYyZjhkNGQ2NTBmOTk5OWZjOTQxNjlmZjg3MjgucG5n/auto/auto/85/notrim/5c08a9a697a566343389a078253f7fe3.webp
   2 |         7 | Legion of Anubis                           |     57.41 | https://cdn.csgoskins.gg/public/uih/items/aHR0cHM6Ly9jZG4uY3Nnb3NraW5zLm
W51YmlzX2xpZ2h0LjljYmM4OGFlOGQwYWI2NjY2YzA4NTY4YTc3NmE1YzFjODQ4YjdlYzYucG5n/auto/auto/85/notrim/725301d64d2a92b3f0f82c94036b8e50.webp
   3 |         7 | Inheritance                                |    176.48 | https://cdn.csgoskins.gg/public/uih/items/aHR0cHM6Ly9jZG4uY3Nnb3NraW5zLm
GFpbl9saWdodC5kYzRmNzZmYzM2NTE3NWU0YWM0MmQxNGVhODQ3MmQ3NmNmM2ZhNjYzLnBuZw--/auto/auto/85/notrim/66286698568f34224b86256f3790dbad.webp
   4 |         7 | Head Shot                                  |    134.92 | https://cdn.csgoskins.gg/public/uih/items/aHR0cHM6Ly9jZG4uY3Nnb3NraW5zLm
WRfc2hvdF9ob2xvX2xpZ2h0LmEzYTY4YWRiMjAxNjcxNGM2MjZiNjUzMzM1NGM0ZDg1YTgyMGNkYmEucG5n/auto/auto/85/notrim/37b8db9354aed5cd03a41c016dc28249.webp
   5 |         7 | Ice Coaled                                 |     17.02 | https://cdn.csgoskins.gg/public/uih/items/aHR0cHM6Ly9jZG4uY3Nnb3NraW5zLm
29ndGhpbmdzX2xpZ2h0LmVjMGVkODkzYzJkMGFlMmY5MTdlZjc1ZTBjYjc3MmM1MGMxZmQwMzIucG5n/auto/auto/85/notrim/9a238d8558503981eec09d473aa87da1.webp
   6 |         7 | Slate                                      |     16.33 | https://cdn.csgoskins.gg/public/uih/items/aHR0cHM6Ly9jZG4uY3Nnb3NraW5zLm
HJvZmVzc2lvbmFsX2xpZ2h0LjI5NmJhOGIxNTEwZGE0OWFlZjU1MWY2ZTNlOGVkODBlMjE3ZGE3NDYucG5n/auto/auto/85/notrim/187f28c57969731ba970cf4105fafe5b.webp
   7 |         7 | Asiimov                                    |    641.89 | https://cdn.csgoskins.gg/public/uih/items/aHR0cHM6Ly9jZG4uY3Nnb3NraW5zLm
XNpaW1vdl9saWdodC4xZWMwMjRhOWVkMjE0NzE4MjUzOTA3NjM4MzFmZGNkNDE3NTA1Y2MwLnBuZw--/auto/auto/85/notrim/500cc78c57df4c739bccf978b38d8d83.webp
   8 |         7 | Neon Revolution                            |    446.73 | https://cdn.csgoskins.gg/public/uih/items/aHR0cHM6Ly9jZG4uY3Nnb3NraW5zLm
W5hcmNoeV9saWdodC5kMjFhNGYxNDQ3YTBjMGM3ODQ4YmY4ZTM5OGQ1MGUwOTcwYzBkZTdmLnBuZw--/auto/auto/85/notrim/999cc0ccae568cc864a8a0eed0fcbedf.webp
   9 |         7 | Searing Rage                               |     42.47 | https://cdn.csgoskins.gg/public/uih/items/aHR0cHM6Ly9jZG4uY3Nnb3NraW5zLm
G9zaXZlX2xpZ2h0LjY0MWUyYzA2ZThlOGY1YmEzZDE5OTg2YTczN2FjODcwMGI3NjYxMTgucG5n/auto/auto/85/notrim/a4c0031808d3fc0b445b309c26ae96bb.webp
  10 |         7 | The Empress                                |    333.46 | https://cdn.csgoskins.gg/public/uih/items/aHR0cHM6Ly9jZG4uY3Nnb3NraW5zLm
W1wcmVzc19saWdodC5kOWVjNzE5M2Y2MmM3ZjIxZTI3MmViMWJhYjFiYjE5ODdmOTlhYmM0LnBuZw--/auto/auto/85/notrim/0e33eda6786bc1b4cfdaf3ad9608c792.webp
  11 |         7 | Frontside Misty                            |    187.54 | https://cdn.csgoskins.gg/public/uih/items/aHR0cHM6Ly9jZG4uY3Nnb3NraW5zLm
2ludGVyX3Nwb3J0X2xpZ2h0LmJjOTM2ZTNhMDIyNWQwNzViNThiY2VlMTJiZmQyMmViZTJhNjNiZmUucG5n/auto/auto/85/notrim/bdbfc7f6137f2e8bc41dffcb70622491.webp
```

```
bakimidb=# select count(*) AS nb_skins from skin;
 nb_skins
----------
     1355
(1 ligne)
```

```
CREATE TABLE skin (
    id SERIAL PRIMARY KEY,
    weapon_id INTEGER NOT NULL REFERENCES weapon(id),
    name TEXT NOT NULL,
    max_price REAL,
    image_url TEXT,
    UNIQUE (weapon_id, name)
);
```

Table Auth_user

```
 id |                                       password                                        |          last_login           | is_superuser | username  | first_name | last_name |        email         | is_staff | is_active |          da
te_joined
----+---------------------------------------------------------------------------------------+-------------------------------+--------------+-----------+------------+-----------+----------------------+----------+-----------+------------
  1 | pbkdf2_sha256$600000$VCjEmUKZqu31V7ZygFjnrj$SqoY5ELZyWy5hmJ4rTnhPrjADAA6WBmRuP2DvamaZYg= | 2026-01-05 19:40:03.597918+01 | f            | kameeelol |            |           | kboy9562@gmail.com   | f        | t         | 2025-12-29
16:29:47.508517+01
(1 ligne)
```

Table user_favorite

```
bakimidb=# select * from user_favorite;
 user_id | skin_id
---------+---------
(0 ligne)
```

```
CREATE TABLE user_favorite (
    user_id INTEGER NOT NULL,
    skin_id INTEGER NOT NULL REFERENCES skin(id),
    PRIMARY KEY (user_id, skin_id)
);
```

Contribution des membres du groupe

Au sein du projet, nous avons assuré la mise en place de l’environnement de développement ainsi que l’architecture initiale de l’application. Nous avons développé la première version fonctionnelle, incluant la page d’accueil et le système de connexion, qui a ensuite été rapidement améliorée et enrichie grâce aux échanges et aux contributions des autres membres du groupe. La conception de l’interface utilisateur a également été réalisée de manière collaborative, incluant le développement des différentes views ainsi que l’intégration de boutons interactifs, sur lesquels nous avons travaillé conjointement. Nous avons également participé activement au travail collaboratif autour de la collecte et de la structuration des données, notamment par la création et la normalisation de fichiers CSV à l’aide de Thunderbit. Ce travail en équipe a permis d’itérer efficacement sur une base technique stable et évolutive.