BREVET DE TECHNICIEN SUPÉRIEUR
SERVICES INFORMATIQUES AUX ORGANISATIONS
Option : Solutions logicielles et applications métiers

# U7 – CYBERSÉCURITÉ DES SERVICES INFORMATIQUES

SESSION 2026

_____

Durée : 4 heures
Coefficient : 4

_____

Matériel autorisé :
Aucun matériel ni document est autorisé.

Dès que le sujet vous est remis, assurez-vous qu'il est complet.

Le sujet comporte 19 pages, numérotées de 1/19 à 19/19.

# LES OLINS

Ce sujet comporte 19 pages dont un dossier documentaire de 11 pages.

## Barème

| | | |
|---|---|---|
| DOSSIER A | **Analyse des risques liés à l'application GESSYCA** | 33 points |
| DOSSIER B | **Sécurisation des accès à l'application GESSYCA** | 33 points |
| DOSSIER C | **Sécurisation de l'application de gestion des commandes** | 14 points |
| | TOTAL | 80 points |

## Dossier documentaire


**Documents associés au dossier commun ..... 9**
Document 1 : Extrait de la base de données DB_GESSYCA ..... 9
Document 2 : Diagramme état-transition du processus de traitement des commandes ..... 10
**Documents associés au dossier A ..... 11**
Document A1 : Les récits utilisateurs ..... 11
Document A2 : Fonction verifierDateMotDePasse() ..... 11
Document A3 : Fonction verifierStructMotDePasse() ..... 11
Document A4 : Déclencheur before_update_utilisateur ..... 12
Document A5 : Déclencheur after_update_utilisateur ..... 12
Document A6 : Fonction inscrireUtilisateur() ..... 13
**Documents associés au dossier B ..... 14**
Document B1 : Extrait des journaux d'accès du serveur de base de données ..... 14
Document B2 : Fichier de génération du jeton CSRF – csrf_token.php ..... 14
Document B3 : Extrait du fichier - formDesactivationUtilisateur.php ..... 14
Document B4 : Extrait du fichier - desactivationDateUtilisateur.php ..... 15
Document B5 : Extrait du code du script - desactivationUtilisateur.php ..... 16
Document B6 : Extrait du code du script - f_bdd.php ..... 17
**Documents associés au dossier C ..... 18**
Document C1 : Présentation et utilisation de SIRENE *open data* ..... 18
Document C2 : Processus d'obtention d'une clé pour Sirene *open data* ..... 18
Document C3 : Test de Sirene *open data* ..... 19
Document C4 : Classement OWASP des risques de sécurité concernant les API (extrait) ..... 19

# Présentation du contexte

## L'association LES OLINS

Le Collège Paul SOBRE, situé en Occitanie a ouvert en 1965. Il est connu pour ses initiatives dans le domaine de l'inclusion scolaire, avec son unité localisée pour l'inclusion scolaire (ULIS) qui accueille en moyenne une dizaine d'élèves en situation de handicap, ayant des besoins similaires mais pas obligatoirement le même type de handicap. Il est coordonné par une enseignante, Valérie LE BRUZ qui, en plus d'assurer les enseignements, a également la charge d'organiser le travail des AESH (accompagnants des élèves en situation de handicap).

Afin de développer et favoriser l'acquisition des compétences de ces élèves, Mme LE BRUZ, coordinatrice du dispositif ULIS, en lien avec le chef d'établissement, a créé en 2020 une association, LES OLINS, dont l'objectif est la commercialisation d'objets personnalisables : portes clés, *magnets* (en français des aimantins, petits objets décoratifs ou publicitaires aimantés), décapsuleurs, éventails, etc. À travers cette association, les élèves, aidés des AESH, sont confrontés, concrètement, à certains aspects de la réalité d'une entreprise, depuis la prise de commande des clients jusqu'à la livraison et l'encaissement, en passant par la confection des produits.

L'idée de départ était de commercialiser les produits uniquement auprès des acteurs gravitant autour du collège (élèves, parents d'élèves, enseignants, etc.), mais succès aidant, la structure s'est développée vers les entreprises de la région. Aujourd'hui, LES OLINS, ce sont une dizaine de membres (élèves, enseignants, AESH, personnels de direction, etc.) impliqués dans la structure, une vingtaine de produits personnalisables, un choix de plus de deux cents visuels libres de droits et plus de trois cents clients différents qui ont déjà passé commande.

## Le prestataire et la solution informatique

Pour automatiser le traitement des commandes, Mme LE BRUZ avait dans un premier temps fait appel à un stagiaire en informatique pour développer l'application GESSYCA, une application *web* permettant de gérer le cycle de vie complet d'une commande client. Afin de créer la commande, le client contacte l'association ou vient sur place, et l'un des membres de l'association s'occupe de la saisir dans l'application.

Si ce choix initial s'est avéré pertinent, le développement de la structure, mais aussi les menaces grandissantes liées à la cybersécurité ont amené LES OLINS à confier le support, la maintenance et l'évolution de cette application à un prestataire plus aguerri.

Mme MARAIX, cheffe de projet de l'ESN retenue (CyberDev Solutions certifiée ISO 27001), vous accueille en tant que développeur(se), spécialiste en cybersécurité, vous devrez :

- participer à l'atelier d'analyse de risque ;
- sécuriser les accès à l'application GESSYCA ;
- sécuriser l'application de gestion des commandes.

**Vous vous appuierez sur le dossier documentaire mis à votre disposition.**

## Dossier A – Analyse des risques liés à l'application GESSYCA

Dans un premier temps, Mme MARAIX va prendre la main sur les réalisations passées, afin d'identifier les éventuelles faiblesses applicatives d'un point de vue sécurité, mais également de collecter les nouveaux besoins identifiés par l'association LES OLINS, et pas encore couverts par l'existant.

### Mission A1 – Récits utilisateurs et conformité avec le règlement général sur la protection des données (RGPD)

Les utilisateurs de l'application GESSYCA, membres de l'association, ont chacun un rôle qui peut être :

- administrateur : a la possibilité d'ajouter de nouveaux membres (élèves, enseignants, AESH, administratifs) ;
- gestionnaire : s'occupe de la gestion des commandes (création, suivi, etc.) et gère le catalogue des produits (ajout, modification, suppression) ;
- lecteur : peut consulter le catalogue, les stocks et l'état des commandes.

Tous les accès utilisateurs nécessitent la création d'un compte et la définition des droits correspondants.

Un extrait des récits utilisateurs est présenté dans le dossier documentaire.

**Question A1.1**
Justifier l'évaluation du récit utilisateur 28 pour chacun des critères de sécurité.

Si LES OLINS proposent un choix de plus de deux cents visuels libres de droit, il arrive régulièrement que les clients viennent avec leur propre visuel. Par exemple, commander un porte-clé avec la photo d'une personne ou d'un groupe de personnes (parents, grands-parents, frères, sœurs, etc.).

Mme MARAIX est présente lorsqu'une cliente est venue commander :

- des tasses avec la photo de ses enfants, qu'elle souhaite offrir aux grands-parents ;
- ainsi qu'un tapis de souris avec la photo d'un paysage qu'elle a prise lors de ses dernières vacances.

Mme MARAIX alerte alors Mme LE BRUZ sur la nécessité de se prémunir par rapport à la gestion de ces visuels personnalisés. Pour ce faire, elle propose de faire signer un document de consentement du droit à l'image aux clients concernés.

**Question A1.2**
Justifier, pour chaque photo de cette commande, de l'utilité ou non d'un tel document, selon le RGPD.

**Question A.1.3**
Indiquer le traitement à réaliser sur les images à caractère personnel une fois la commande livrée.

**Mission A2 - Prise en charge d'une authentification incorrecte**

Pour le moment, en cas de saisie incorrecte au niveau du formulaire d'authentification de l'application GESSYCA, un message d'erreur apparait, différent suivant le type d'erreur : "identifiant non reconnu" ou, si l'identifiant est reconnu et que le mot de passe est erroné, "mot de passe incorrect".

Les tentatives de connexion ne sont pas limitées.

Ce mode opératoire pose différents problèmes de sécurité. Vous avez la responsabilité de déterminer lesquels et de proposer des solutions.

**Question A2.1**
a) Expliquer pourquoi le système actuel des messages d'erreur pose problème.
b) Proposer une autre solution plus sécurisée.
c) Donner deux exemples de types d'attaque pouvant profiter du fait que les tentatives de connexion ne sont pas limitées.

**Mission A3 – Évènements redoutés et mesures mises en place**

En complément des récits utilisateurs, Mme MARAIX a identifié une liste des évènements redoutés auxquels l'association LES OLINS est susceptible d'être confrontée. Parmi ceux-ci, on note l'évènement : « À la suite d'une fuite de données, un attaquant récupère les mots de passe stockés dans la base de données ».

Pour contrer cet évènement redouté, des mesures de deux types ont été proposés et implémentés :

- des contraintes imposées aux utilisateurs, qui portent sur les règles de définition des mots de passe ;
- des dispositifs techniques, qui visent à renforcer la sécurité des développements.

**Question A3.1**
Décrire les contraintes imposées aux utilisateurs de l'application en matière de gestion des mots de passe.

De la même manière, la fonction d'inscription des utilisateurs a été modifiée afin de sécuriser au maximum l'application. On note l'utilisation du salage comme technique mise en œuvre.

**Question A3.2**
a) Décrire la façon dont le salage a été implémenté dans le code.
b) Définir la technique du salage.
c) Préciser le type d'attaque auquel le salage permet, entre autres, de faire face.

## Dossier B – Sécurisation des accès à l'application GESSYCA

Le 12 janvier, comme tous les matins, Mme LE BRUZ s'apprête à traiter les dernières commandes, c'est-à-dire à passer à l'état « Validée » les commandes « En cours » afin d'indiquer que les produits vont être confectionnés.

Mais elle constate, dans l'application GESSYCA, qu'il n'existe plus aucune commande ayant le statut « En cours ».

### Mission B1 – Analyse de l'incident

Mme LE BRUZ craint d'avoir fait l'objet d'une cyber-attaque et demande donc à Mme MARAIX d'investiguer le plus rapidement possible.

Mme MARAIX va commencer par analyser les journaux disponibles (*logs*) et vous charge d'étudier ceux de la base de données.

**Question B1.1**
Repérer dans les journaux d'accès du serveur de base de données de l'application GESSYCA les éléments qui pourraient laisser penser que l'association a pu faire l'objet d'une cyber-attaque.

**Question B1.2**
Détailler les impacts de cet acte malveillant pour l'association LES OLINS.

En comparant avec les journaux (*logs*) de connexion à l'application GESSYCA, Mme MARAIX retrouve le nom du membre qui a réalisé ces modifications de statut et le communique à Mme LE BRUZ.

En contactant ce dernier, celui-ci lui avoue avoir noté ses identifiants de connexion dans un carnet au sein duquel il notait toutes les informations en lien avec l'association LES OLINS, carnet qu'il a malheureusement perdu quelques jours auparavant.

**Question B1.3**
Citer l'une des infractions qui peut être retenue contre la personne qui a modifié le statut des commandes de façon malveillante.

**Question B1.4**
Citer le document qui précise les risques encourus par le membre qui a perdu son carnet.

**Mission B2 – Désactivation des membres de l'association**

Lors de l'analyse de cet incident de sécurité, Mme MARAIX a constaté qu'un compte utilisateur d'un membre ayant quitté l'association depuis plusieurs mois était toujours actif dans le système et disposait de privilèges élevés. En effet, il n'est pas rare qu'un élève s'investisse dans l'association en sachant qu'à la fin de l'année scolaire, il la quittera pour diverses raisons comme son départ au lycée par exemple.

Bien que ce compte n'ait pas été directement impliqué dans l'incident, son maintien actif présentait un risque de sécurité. Cela a mis en évidence une lacune dans la gestion des comptes utilisateurs. Mme LE BRUZ demande alors à Mme MARAIX de mettre en place un système de désactivation des utilisateurs.

L'identifiant (*login*) de l'utilisateur à désactiver, sera saisi via un formulaire ainsi qu'une date de désactivation qui sera stockée dans la base de données DB_GESSYCA.

À cet effet un champ *dateDesactivation* est ajouté à la table utilisateur :

> **Utilisateur** (id, login, mdp, nom, prenom, fonction, role, estActif, *dateDesactivation*)
> Clé primaire : id
> estActif : booléen qui indique si l'utilisateur peut se connecter.

Quotidiennement à 1 heure du matin, une tâche CRON lancera l'exécution d'un script PHP dont l'objectif sera de désactiver les utilisateurs lorsque la date de désactivation prévue est atteinte.

**Question B2.1**
Justifier, en matière de preuve, le choix de désactiver les utilisateurs plutôt que de les effacer.

Une partie de cette fonctionnalité a été réalisée, Mme MARAIX vous demande de la finaliser ainsi que de procéder à une évaluation de la sécurité, en identifiant les forces et les faiblesses du code.

**Question B2.2**
Identifier les trois éléments de sécurité mis en place dans le traitement de la désactivation d'un membre.

**Question B2.3**
Compléter le code de la fonction *desactiverUtilisateurs()* pour désactiver les utilisateurs appropriés.

En analysant le code fourni, vous constatez plusieurs vulnérabilités potentielles au niveau de la connexion à la base de données.

**Question B2.4**
Identifier la faiblesse principale du script *f_bdd.php*.

Par ailleurs, stocker directement les informations de connexion à la base de données dans un script PHP au niveau des sources de l'application l'expose à des risques de sécurité et de confidentialité, car ces données peuvent être plus facilement compromises en cas d'accès non autorisé au code source ou de vulnérabilité dans le code.

**Question B2.5**
Proposer une alternative plus sécurisée.

Peu après la mise en production de cette nouvelle fonctionnalité, une panne de courant électrique de plusieurs heures s'est produite. Mme MARAIX souhaite alors s'assurer que les comptes qui devaient être désactivés l'ont bien été après la remise en route des services.

**Question B2.6**
Proposer une requête qui permette de s'assurer du bon fonctionnement du traitement.

## Dossier C – Sécurisation de l'application de gestion des commandes

### Mission C1 – Risque de malveillance

Dans la base de données, on ne conserve que l'état courant de la commande, une commande ne pouvant pas revenir à un état précédent.

L'ordre des états possibles est le suivant :

En cours – Validée – En Production – Terminée – Réglée – Clôturée.

Mme MARAIX conseille à Mme LE BRUZ de conserver un historique des modifications d'état des commandes, ainsi que l'utilisateur responsable de ce changement.

**Question C1.1**
Indiquer quel critère de la sécurité de l'information la base de données ne respecte pas concernant l'état de la commande.

Pour toutes les commandes passées, quelles qu'elles soient, et en réponse à ce manque, Mme LE BRUZ souhaite pouvoir suivre l'historique des changements d'état :

- Quel collaborateur a changé le statut ? À quelle date ?
- Un commentaire spécifique peut être aussi renseigné à chaque changement d'état.

**Question C1.2**
Proposer les modifications à réaliser pour répondre à ces exigences. Seuls les éléments du schéma existant qui sont concernés par l'évolution seront repris dans le formalisme de votre choix.

### Mission C2 - Consommation (utilisation) d'une interface de programmation (API) en données ouvertes (*open data*)

Au regard des nombreuses sollicitations d'entreprise, Mme LE BRUZ est préoccupée par le temps que prend l'ajout des nouveaux clients professionnels et la vérification des données fournies.

En effet, Mme LE BRUZ, après avoir récupéré les informations, vérifie manuellement sur le site « https://sirene.fr/sirene/public/accueil » que les données de l'entreprise fournies sont cohérentes et exactes (numéro de SIRET, raison sociale, adresse, nom du dirigeant, etc.).

Mme MARAIX vous demande d'étudier l'interface de programmation, nommée Sirene open data, qui permet d'interroger le répertoire (Sirene) des entreprises et des établissements gérés par l'Insee. Cette solution technique permettrait d'automatiser cette vérification à partir de l'interface d'ajout d'un nouveau client de type entreprise.

Le dossier documentaire présente le résultat d'un test effectué avec un outil de test d'interface de programmation.

**Question C2.1**
a) Donner la cause probable de l'erreur obtenue.
b) Préciser le risque de sécurité OWASP auquel ce mécanisme est censé répondre.

Après avoir résolu le problème précédent et toujours lors de cette phase d'essai, l'erreur 429 a été provoquée de façon volontaire.

**Question C2.2**
Nommer le type d'attaque qui provoque cette erreur.

## Documents associés au dossier commun

***Document 1 : Extrait de la base de données DB_GESSYCA***

Schéma entité-association :

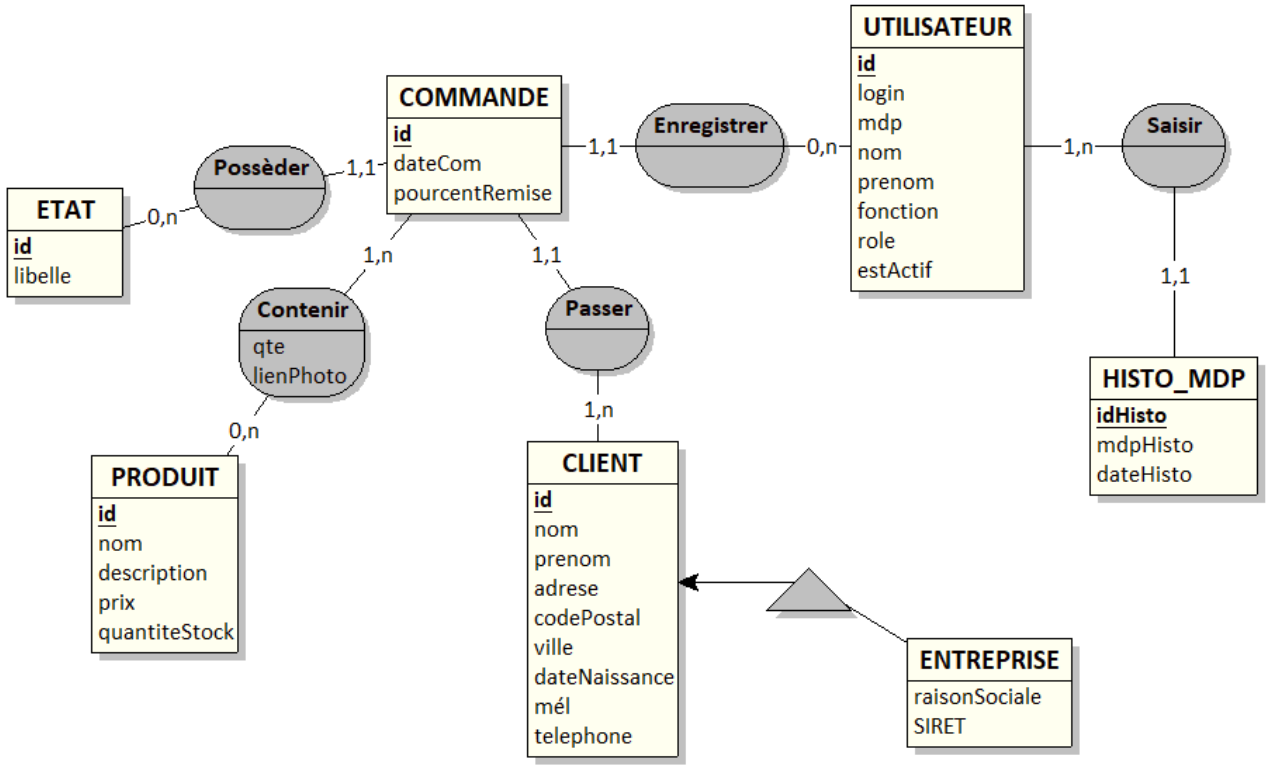
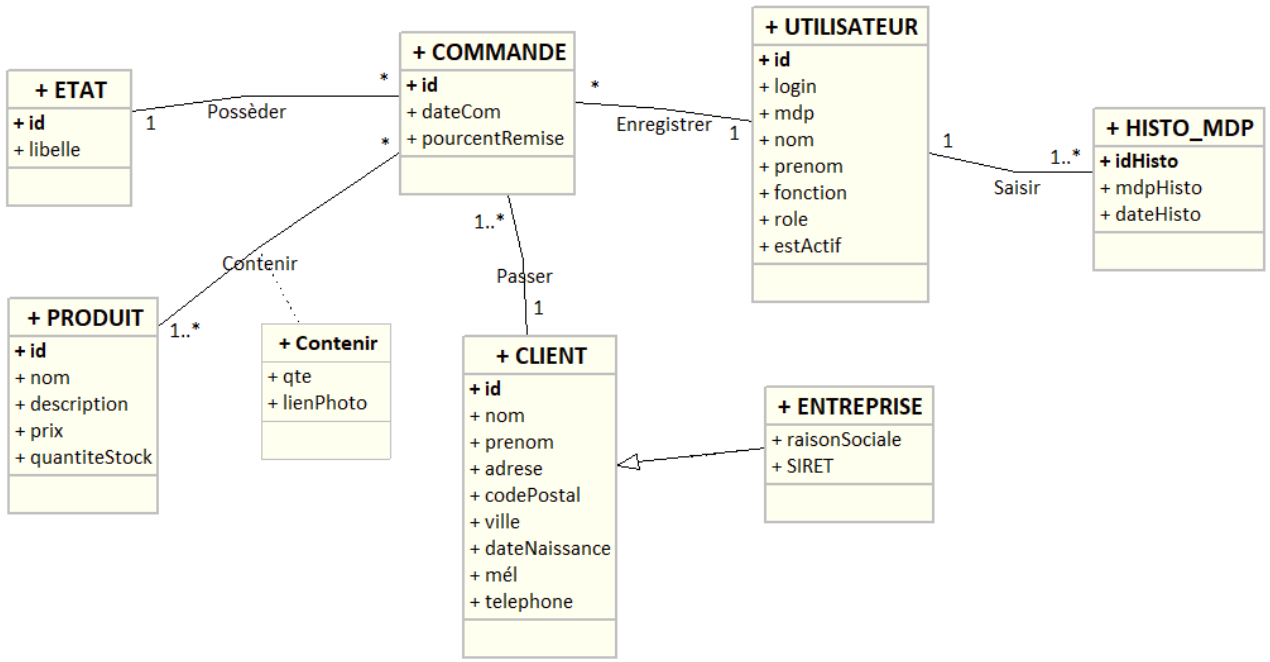



Diagramme de classe :




…/…

Schéma relationnel :

**Etat** (id, libelle)
Clé primaire : id
Libelle peut prendre les valeurs : En cours, Validée, En Production, Terminée, Réglée, Clôturée

**Produit** (id, nom, description, prix, quantiteStock)
Clé primaire : id

**Utilisateur** (id, login, mdp, nom, prenom, fonction, role, estActif)
Clé primaire : id
Le champ role peut prendre les valeurs : administrateur, gestionnaire, lecteur
Le champ fonction peut prendre les valeurs : Collégien, Personnel ou Direction

**Histo_Mdp** (idHisto, mdpHisto, dateHisto, idUtilisateur)
Clé primaire : idHisto
Clé étrangère : idUtilisateur en référence à id de Utilisateur

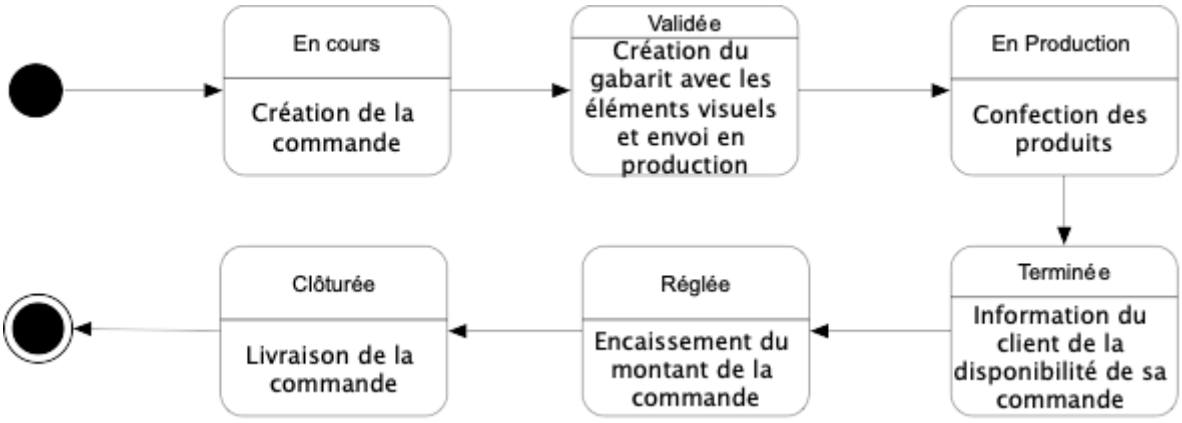
**Client** (id, nom, prenom, adresse, codePostal, ville, dateNaissance, mél, telephone)
Clé primaire : id

**Entreprise** (idClient, raisonSociale, SIRET)
Clé primaire : idClient
Clé étrangère : idClient en référence à id de Client

**Commande** (id, dateCom, pourcentRemise, idUtilisateur, idClient, idEtat)
Clé primaire: id
Clés étrangères : idUtilisateur en référence à id de Utilisateur
idClient en référence à id de Client
idEtat en référence à id de Etat

**Contenir** (idCommande, idProduit, qte, lienPhoto)
Clé primaire : idCommande, idProduit
Clés étrangères : idCommande en référence à id de Commande
idProduit en référence à id de Produit

***Document 2 : Diagramme état-transition du processus de traitement des commandes***




Un gabarit d'impression est un modèle préconçu qui sert de guide pour la mise en page des créations.

## Documents associés au dossier A

### *Document A1 : Les récits utilisateurs*

| ID | Cas d'Utilisation | Critères de sécurité | | | |
|---|---|---|---|---|---|
| | | D | I | C | P |
| 4 | Un lecteur veut consulter le catalogue des produits. | * | - | - | - |
| 11 | Un utilisateur veut s'authentifier. | * | ** | ** | * |
| 13 | Un gestionnaire veut enregistrer une commande. | ** | ** | * | * |
| 17 | Un gestionnaire veut modifier le statut d'une commande. | * | * | * | ** |
| 28 | Un gestionnaire, qui s'occupe de la comptabilité, veut télécharger les données des ventes du mois en cours contenant les informations client. | * | ** | ** | - |
| 36 | L'application GESSYCA veut mettre à jour la liste des entreprises. | - | ** | - | - |

D : disponibilité I : intégrité C : confidentialité P : preuve
- : sans objet * : modéré ** : important

### *Document A2 : Fonction verifierDateMotDePasse()*

Cette fonction est appelée lors de la tentative de connexion d'un utilisateur ayant un rôle administrateur.

```
function verifierDateMotDePasse(string $date) : bool {

    $dateMotDePasse = new DateTime($date); //date à laquelle on modifie le mot de passe
    $dateLimite = new DateTime();  //initialisation avec la date du jour

    $dateLimite->modify('-3 months');

    $retour = false;
    if ($dateMotDePasse < $dateLimite) {

        $retour = true;
    }
    return $retour;
}
```

La méthode modify modifie la variable de type DateTime en l'incrémentant ou en la décrémentant selon le paramètre donné.

### *Document A3 : Fonction verifierStructMotDePasse()*

```
function verifierStructMotDePasse(string $motDePasse) : bool {

    $retour = false;

    if (strlen($motDePasse) >= 8) {

        $chiffre = preg_match('/\d/', $motDePasse);
        $minuscule = preg_match('/[a-z]/', $motDePasse);
        $majuscule = preg_match('/[A-Z]/', $motDePasse);
        $carSpecial = preg_match('/[^a-zA-Z0-9]/', $motDePasse);

        $retour = $chiffre && $minuscule && $majuscule && $carSpecial;
    }
    return $retour;
}
```

***Document A4 : Déclencheur before_update_utilisateur***

```
DELIMITER |
CREATE TRIGGER before_update_utilisateur BEFORE UPDATE ON Utilisateur
FOR EACH ROW
BEGIN
    DECLARE v_nbFois INT DEFAULT 0;

    SELECT COUNT(*) INTO v_nbFois
    FROM (SELECT mdpHisto
          FROM Histo_Mdp
          WHERE idUtilisateur = NEW.id
          ORDER BY dateHisto DESC LIMIT 3) AS M
    WHERE M.mdpHisto = NEW.mdp;

    IF v_nbFois > 0 THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Votre mot de passe n\'est pas
                                                    conforme aux règles en vigueur...';
    END IF;
END |
```

***Document A5 : Déclencheur after_update_utilisateur***

```
DELIMITER |
CREATE TRIGGER after_update_utilisateur AFTER UPDATE ON Utilisateur
FOR EACH ROW
BEGIN
    IF(NEW.mdp != OLD.mdp) THEN
        INSERT INTO Histo_Mdp (idUtilisateur, dateHisto, mdpHisto)
            VALUES (NEW.id, NOW(), OLD.mdp);
    END IF;
END |
```

***Document A6 : Fonction inscrireUtilisateur()***

Cette fonction est appelée lors de la création d'un nouvel utilisateur.

```
function inscrireUtilisateur(string $login, string $password, string $nom,
                              string $prenom, string $fonction, string $role) : void {
    $pdo = get_db_connection();
    if ($pdo  !== null) {
        if (!verifierStructMotDePasse($password)) {
            echo "Le mot de passe ne répond pas aux critères requis.";
        }
        else
        {
            $hashed_password = password_hash($password, PASSWORD_DEFAULT);

            $stmt = $pdo->prepare(
                        "INSERT INTO Utilisateur (login,mdp,nom,prenom,fonction,role)
                                 VALUES (:login,:mdp,:nom,:prenom,:fonction,:role)");
            $stmt->bindParam(':login', $login);
            $stmt->bindParam(':mdp', $hashed_password);
            $stmt->bindParam(':nom', $nom);
            $stmt->bindParam(':prenom', $prenom);
            $stmt->bindParam(':fonction', $fonction);
            $stmt->bindParam(':role', $role);

            try {
                $stmt->execute();
                echo "Utilisateur inscrit avec succès !";
            } catch (PDOException $e) {
                echo "Erreur lors de l'inscription de l'utilisateur : " .
                                                                    $e->getMessage();
            }
        }
    }
}
```

Remarque : la fonction password_hash() génère un salage (*salt*) sécurisé automatiquement et crée une nouvelle empreinte (*hash*) en utilisant un algorithme de hachage fort et irréversible.

La constante PASSWORD_DEFAULT, représentant l'algorithme à utiliser lors du hachage du mot de passe, est conçue pour changer dans le temps, au fur et à mesure que des algorithmes plus récents et plus forts sont ajoutés à PHP.

## Documents associés au dossier B

### *Document B1 : Extrait des journaux d'accès du serveur de base de données*

Extrait du fichier /var/log/mysql/mysql.log

```
…
260111 23:37:45   294 Connect  root@localhost on DB_GESSYCA using Socket
           294 Query    SELECT * FROM Commande C JOIN Etat E ON C.idEtat = E.id
                        WHERE libelle = 'En cours'
           294 Query    UPDATE Commande
                        SET idEtat = (SELECT id FROM Etat WHERE libelle = 'Clôturée')
                        WHERE libelle = 'En cours'
           294 Quit
```

### *Document B2 : Fichier de génération du jeton CSRF – csrf_token.php*

L'emplacement de ce fichier est : /var/www/gessyca/

```
<?php

    session_start();

    if (empty($_SESSION['csrf_token'])) {
        $_SESSION['csrf_token'] = bin2hex(random_bytes(32));
    }

    function getCsrfToken(): string {
        return $_SESSION['csrf_token'];
    }

    function validerCsrfToken(string $token): bool {
        return hash_equals($_SESSION['csrf_token'], $token);
    }
```

### *Document B3 : Extrait du fichier - formDesactivationUtilisateur.php*

L'emplacement de ce fichier est : /var/www/gessyca/

```
<?php
    require 'csrf_token.php';
    $csrfToken = getCsrfToken();
?>
<form action="desactivationDateUtilisateur.php" method="post">
    <input type="hidden" name="csrf_token" value="<?php echo $csrfToken; ?>">
    <label for="login">Login :</label>
    <input type="text" id="login" name="login" required>
    <br>
    <label for="dateDesactivation">Date de désactivation :</label>
    <input type="date" id="dateDesactivation" name="dateDesactivation" required>
    <br>
    <button type="submit">Enregistrer</button>
</form>
```

***Document B4 : Extrait du fichier - desactivationDateUtilisateur.php***

L'emplacement de ce fichier est : /var/www/gessyca/

```
<?php
require 'csrf_token.php';
require_once 'f_bdd.php';
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
   if (!validerCsrfToken($_POST['csrf_token'])) {
      die("Échec de la validation du jeton CSRF");
   }
   $login = htmlspecialchars(trim($_POST['login']));
   $dateDesactivation = htmlspecialchars(trim($_POST['dateDesactivation']));
   if (!empty($login) && !empty($dateDesactivation)) {
      if (validerDate($dateDesactivation)) {
         if (sauvegarderDesactivationDate($login, $dateDesactivation)) {
            echo "La date de désactivation a été enregistrée avec succès pour
                                                                    le login $login";
         }
         else {
            echo "Erreur lors de l'enregistrement de la date de désactivation du
                                                                       login $login";
         }
      }
      else {
         echo "Date de désactivation $dateDesactivation invalide.";
      }
   }
   else {
      echo "Tous les champs sont obligatoires.";
   }
}




function validerDate(string $date, $format = 'Y-m-d'): bool {
    $d = DateTime::createFromFormat($format, $date);
    // Retourne `true` si la date fournie est valide et correspond
    // exactement au format donné. Sinon, retourne `false`.
    return $d && $d->format($format) === $date;
}
```

**Le document continue sur la page suivante.**

```
function sauvegarderDesactivationDate(string $login, string $dateDesactivation): bool
{
   $resultat = false;
   try {
      $cnx = connexionPDO();
      $sql = "UPDATE Utilisateur SET dateDesactivation = :dateDesactivation
              WHERE login = :login";
      $req = $cnx->prepare($sql);
      $req->bindValue(':dateDesactivation', $dateDesactivation, PDO::PARAM_STR);
      $req->bindValue(':login, $login, PDO::PARAM_STR);

      // renvoie true si la requête a été correctement exécutée, false sinon
      $resultat = $req->execute();
   } catch (PDOException $e) {
      print "Erreur !: " . $e->getMessage();
      die();
   }
   return $resultat;
}
...
?>
```

***Document B5 : Extrait du code du script - desactivationUtilisateur.php***

L'emplacement de ce fichier est : /var/www/gessyca/

```
<?php
    require_once 'f_bdd.php';
/**
 * Permet de désactiver les utilisateurs au niveau
 * de la base de données lorsque la date de désactivation stockée (champ
 * dateDesactivation) est égale ou antérieure à la date actuelle.
 * Renvoie true si la requête a été correctement exécutée, false dans le cas contraire.
*/
function desactiverUtilisateurs(): bool {

    $dateActuelle = date('Y-m-d');

    // ***** À COMPLÉTER *****
}
...
?>
```

***Document B6 : Extrait du code du script – f_bdd.php***

L'emplacement de ce fichier est : /var/www/gessyca/

```
<?php
const DB_HOST = 'localhost';
const DB_NAME = 'DB_GESSYCA';
const DB_USER = 'root';
const DB_PASS = 'A?(bGvX2[[bU';
/**
 * Fonction de connexion à la base de données
 * Utilise PDO pour établir une connexion sécurisée
*/
function connexionPDO() {
    try {
        // DSN (Data Source Name) pour la connexion
        $dsn = 'mysql:host=' . DB_HOST . ';dbname=' . DB_NAME . ';charset=utf8mb4';
        // Création de l'instance PDO
        $pdo = new PDO($dsn, DB_USER, DB_PASS);
        // Configuration des options de PDO
        $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
        $pdo->setAttribute(PDO::ATTR_DEFAULT_FETCH_MODE, PDO::FETCH_ASSOC);
        return $pdo;
    } catch (PDOException $e) {
        // Gestion de l'erreur de connexion
        echo "Erreur de connexion à la base de données : " . $e->getMessage();
        return null;
    }
}
```

## Documents associés au dossier C

### *Document C1 : Présentation et utilisation de SIRENE open data*

Sirene open data permet d'interroger le répertoire Sirene des entreprises et des établissements, géré par l'Institut national de la statistique et des études économiques (Insee), et principalement, d'obtenir :

- des informations sur une entreprise ou un établissement du répertoire ;
- des informations sur plusieurs entreprises ou établissements sélectionnées selon un ou plusieurs critères (activité, géographie, etc.) ;
- des informations sur les mises à jour quotidiennes du répertoire.

Pour accéder à Sirene open data, il est nécessaire d'avoir un compte sur le portail de l'Insee et de souscrire à cette interface de programmation (API).

**Utilisation du service**

Il s'agit d'un service *web* de type REST, qui s'appuie donc uniquement sur les protocoles et standards utilisés sur le *Web*. L'invocation du service se fait par envoi d'une requête HTTPS (de type GET ou POST) sur une adresse réticulaire (URL) publique, par exemple :

https://api.insee.fr/api-sirene/3.11/siren/123456789

Le résultat est fourni au format Json.

**Codes de retour**

Les codes renvoyés par les services sont conformes aux spécifications HTTP :

- 200-*OK* : entreprise trouvée, les caractéristiques sont fournies dans le contenu de la réponse ;
- 400-*Bad Request* : nombre incorrect de paramètres ou les paramètres sont mal formatés ;
- 401-*Unauthorized* : clé d'API manquante ou invalide ;
- 403-*Forbidden* : droits insuffisants pour consulter les données sur cette entreprise ;
- 404-*Not Found* : entreprise non trouvée dans la base Sirene (cela signifie que le numéro de 9 chiffres ne correspond pas à un Siren présent dans la base) ;
- 406-*Not acceptable* : le paramètre « Accept » de l'en-tête http contient une valeur non prévue
- 429-*Too Many Requests* : quota d'interrogations de l'API dépassé ;
- 500-*Internal Server Error* : erreur interne du serveur ;
- 503-*Service Unavailable* : service indisponible.

*Source : https://www.data.gouv.fr/dataservices/api-sirene-open-data/*

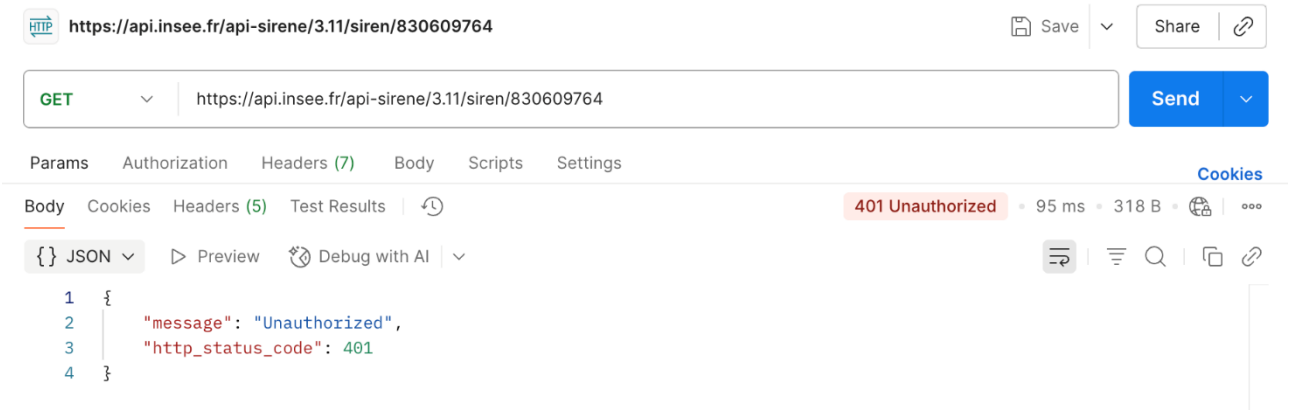
### *Document C2 : Processus d'obtention d'une clé pour Sirene open data*

Mme MARAIX devra créer un compte, sur le portail en ligne de l'Insee, en renseignant : nom, prénom, adresse de courriel, identifiant et mot de passe. Un courriel contenant un lien d'activation lui sera envoyé.

Après s'être authentifiée, elle aura accès à un tableau de bord qui lui permettra d'indiquer l'adresse réticulaire (URL) de l'application web GESSYCA qui utilisera le service, cela consiste à créer une « application ».

Cette « application » déclarée pourra alors souscrire à Sirene open data et obtenir une clé (API KEY) générée, qui devra être transmise dans l'en-tête (Header) des requêtes HTTPS.

### *Document C3 : Test de Sirene open data*




### *Document C4 : Classement OWASP des risques de sécurité concernant les API (extrait)*

| | | |
|---|---|---|
| API01 | Autorisation au niveau des objets rompue | Les API ont tendance à exposer des points d'accès (***endpoints***) qui manipulent des identifiants d'objets (OID), créant ainsi une large surface d'attaque sur les contrôles d'accès aux objets. Des contrôles d'autorisation doivent être effectués dans chaque fonction qui accède à une source de données en utilisant un ID fourni par l'utilisateur. |
| API02 | Authentification rompue | Les mécanismes d'authentification sont souvent implémentés de manière incorrecte, permettant aux attaquants de compromettre les jetons (token) d'authentification ou d'exploiter des failles d'implémentation afin d'usurper temporairement ou définitivement l'identité d'autres utilisateurs. L'incapacité d'un système à identifier le client/l'utilisateur compromet la sécurité de l'API dans son ensemble. |
| API05 | Violation des Autorisations au niveau des fonctions | Les politiques de contrôle d'accès complexes avec différentes hiérarchies, groupes et rôles, et une séparation floue entre les fonctions normales et d'administration, tendent à entraîner des failles d'autorisation. En exploitant ces problèmes, les attaquants peuvent accéder aux ressources d'autres utilisateurs et/ou aux fonctions d'administration. |
| API07 | Falsification de requête côté serveur | Les failles de type SSRF (Server-Side Request Forgery) peuvent survenir lorsqu'une API récupère une ressource distante sans valider l'URI fourni par l'utilisateur. Cela permet à un attaquant de forcer l'application à envoyer une requête forgée vers une destination inattendue, même si elle est protégée par un pare-feu ou un VPN. |
| API09 | Gestion incorrecte des ressources | Les API ont tendance à exposer plus de points d'accès que les applications web traditionnelles, ce qui rend une documentation propre et à jour très importante. Un inventaire correct des hôtes et des versions d'API déployées est également important pour atténuer les problèmes tels que les versions d'API obsolètes et les points d'accès de débogage exposés. |

*Source : https://owasp.org/API-Security/editions/2023/en/0x11-t10/*