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 18 pages, numérotées de 1/18 à 18/18.

Cas NDV

Ce sujet comporte 18 pages dont un dossier documentaire de 10 pages.

Barème

DOSSIER A	**Évolutions de l'intranet de l'agence NDV en matière de sécurité**	42 points
DOSSIER B	**Reprise et sécurisation du site internet du client Architect.60**	38 points
	TOTAL	80 points

Dossier documentaire

Présentation du contexte

NDV (Numérique, Développement et Valeur) est une agence digitale qui propose à des clients très divers - parfois de renom - conseil, conception de solutions ergonomiques (selon la démarche *UX Design* - expérience de l'utilisateur final), développement et optimisation de solutions. Les projets concernent des sites de marque ou d'organisation, des boutiques en ligne ou sites marchands, des outils numériques et des plateformes métier.

Depuis plusieurs années, l'entreprise NDV s'engage à fournir des solutions numériques sécurisées, conformes aux normes réglementaires, et à accompagner ses clients dans un environnement numérique en constante évolution.

L'intranet de la société NDV, développé en interne, couvre divers aspects de ses activités : la gestion commerciale, le suivi des projets, la tierce maintenance applicative (TMA), etc.

Au cœur de l'agence NDV, l'équipe « Audit et sécurité » assure la qualité et la sécurité des solutions développées. Par exemple, elle inspecte minutieusement les codes sources produits en interne, garantissant leur conformité aux meilleures pratiques en matière de sécurité.

Depuis peu, l'agence a étendu les activités de son équipe « Audit et sécurité » en lui confiant d'une part des évolutions de l'intranet et d'autre part des missions pour des clients externes portant sur des sites *web* non maintenus par l'entreprise NDV.

Évolutions de l'intranet en matière de sécurité

Face aux évolutions réglementaires en matière de protection des données personnelles, l'agence engage actuellement une révision approfondie de son intranet.

De plus, NDV, en tant qu'agence digitale, reconnaît l'importance cruciale de la confidentialité des échanges de données entre les clients et les équipes de développement. Afin de renforcer cette sécurité, elle prévoit une évolution significative de son intranet de gestion vers la mise en place de canaux de communication sécurisés avec identification dédiée des interlocuteurs sur chaque contrat.

Activités de l'équipe Audit et Sécurité auprès de clients externes

- **audits de sécurité et de conformité** : réalisation d'audits de sécurité sur des sites *web* de clients externes non gérés par l'agence NDV, mettant l'accent sur la sécurité, la protection des données, la conformité aux normes et derniers standards de sécurité ;
- **gestion d'incidents de sécurité :** intervention après des incidents de sécurité sur des sites *web* provenant d'autres prestataires, analyse approfondie des vulnérabilités et mise en place de mesures de sécurité pour assurer une transition en douceur lors de la reprise par l'agence.

Actuellement stagiaire dans l'équipe « Audit et Sécurité » de l'agence NDV, votre responsable, Nathalie Delcati-Vémini, vous demande de participer à deux missions :

- évolutions de l'intranet de l'agence NDV en matière de sécurité ;
- reprise et sécurisation du site internet du client Architect.60 .

Dossier A – Évolutions de l'intranet de l'agence NDV en matière de sécurité

Votre responsable vous demande de participer aux évolutions majeures de l'intranet qui sont concentrées sur trois axes cruciaux :

- la gestion des journaux d'activité (*logs*) du système de gestion de base de données (SGBD) ;
- le renforcement de la protection des données personnelles ;
- l'évolution de l'application de gestion des contrats pour assurer une sécurité optimale des échanges.

Mission A1 – Évolution de la gestion des journaux d'activité (*logs*) du SGBD de l'intranet

La journalisation des accès aux bases de données est considérée comme une bonne pratique pour démontrer la conformité aux principes du RGPD, notamment en ce qui concerne la transparence, la responsabilité et la gestion des incidents de sécurité.

Les informations relatives aux accès et aux opérations effectuées sur la base de données de l'intranet (base *ndv*) sont enregistrées dans une vue SQL appelée *t_logs*. Cette vue est générée en interrogeant les journaux d'activité (*logs*) du SGBD MySQL, spécifiquement la table *mysql.general_log* qui enregistre toutes les requêtes et opérations qui ont lieu sur le serveur MySQL.

La consultation de ces données par les personnes accréditées est rendue possible via une page *web*, alimentée par un fichier *select_logs.php*, qui permet de visualiser et d'analyser le contenu de la vue *t_logs* de la base de données.

Votre responsable vous explique que, lors de l'analyse de ces journaux d'activité, il est difficile d'isoler des informations précises car la page contient des centaines de lignes. Elle aimerait donc une évolution du code actuel pour pouvoir filtrer ces lignes selon la date et leur nature : REQUETE, CONNEXION, ERREUR, DECONNEXION.

L'évolution du code de l'interface graphique, permettant de sélectionner (filtrer) la nature et la date d'opérations, a été réalisée (document A1b). Vous devez terminer les modifications.

Question A1.1

Compléter le code de la fonction *print_logs()* pour aussi tenir compte du filtre par nature. Indiquer seulement le code ajouté et/ou modifié en renseignant les numéros de ligne concernés.

Votre responsable a constaté des anomalies dans les journaux du 19 novembre 2024. L'utilisateur ndv_64784, qui correspond à l'utilisateur du pilote (*driver)* de connexion utilisée pour le site *web* de l'intranet, ne devrait pas pouvoir se connecter depuis une autre adresse IP que celle du serveur local et ne devrait disposer que des droits minimaux requis pour effectuer des opérations de création, lecture, mise à jour et suppression sur les données des tables de la base *ndv*.

Elle vous demande de documenter son analyse et de corriger les droits. Pour cela elle vous fournit un extrait des habilitations actuelles sur la base *ndv*.

Question A1.2

a) Indiquer les éléments dans les journaux qui confirment que l'utilisateur '*ndv_64784*' peut se connecter depuis une autre adresse IP que le serveur local et avec plus de droits que nécessaire.

b) Indiquer les éléments dans les habilitations qui confirment que l'utilisateur '*ndv_64784*' peut se connecter depuis une autre adresse IP que le serveur local et avec plus de droits que nécessaire.

c) Écrire les requêtes permettant de modifier les droits pour que l'utilisateur '*ndv_64784*' ait seulement les bonnes habilitations.

Mission A2 – Renforcement de la protection des données personnelles de l'intranet

Votre responsable souhaiterait tout d'abord avoir une vue d'ensemble des données à caractère personnel et des données sensibles stockées sur l'intranet de l'agence NDV dans le cadre de la gestion des contrats avec les clients.

Question A2.1

a) Rappeler les critères qui caractérisent une donnée à caractère personnel et une donnée sensible au sens de la CNIL.

b) Lister, s'il y en a, les données à caractère personnel et les données sensibles contenues dans les tables de la base de données de l'intranet.

Les données à caractère personnel actuellement stockées dans la base de données de l'intranet semblent insuffisamment protégées. Votre responsable identifie le scénario de risque suivant : « En cas de fuite des données, des attaquants pourraient exploiter les coordonnées des contacts de nos contrats pour conduire une opération d'hameçonnage (*phishing)* contre nos clients ».

Pour limiter ce risque et être à jour en matière de réglementation, votre responsable pense qu'il faudrait stocker les adresses de courrier électronique des contacts externes dans la base de données sous une forme chiffrée. Elle propose d'exploiter les fonctions de chiffrement du SGBD MySQL `AES_ENCRYPT()` et `AES_DECRYPT()` et elle vous confie la tâche de transformer l'actuelle base de données.

Question A2.2

Indiquer si le chiffrement des données à caractère personnel relève d'une obligation légale ou d'une recommandation (bonne pratique).

Question A2.3

Proposer un autre scénario de risque, impliquant un acteur interne dont vous préciserez le rôle, contre lequel le chiffrement des adresses de courrier électronique stockées dans la base serait une protection.

L'utilisation de la fonction `AES_ENCRYPT()` nécessite de modifier le type actuel de la colonne au format VARBINARY(200). Votre responsable vous demande d'opérer les modifications nécessaires pour adapter la structure de la base de données et chiffrer les données actuelles en utilisant la clé de chiffrement « lhh4&abt2! ».

Question A2.4

Écrire les requêtes permettant d'adapter la base de données et de chiffrer les adresses de courrier électronique des contacts externes actuellement stockées dans la base de données.

Conformément aux recommandations, la clé de chiffrement a été stockée dans le fichier de configuration de l'application intranet dans une variable d'environnement `$_ENV['AES_key']`.

À la suite de vos modifications, la fiche contact d'un contrat dans l'application affiche la version chiffrée de l'adresse de courrier électronique. Votre responsable vous charge de corriger le code de l'actuelle fonction getContact() qui fournit les données afin de restaurer l'affichage en clair.

Question A2.5

Corriger la requête préparée dans la fonction getContact() pour rétablir l'affichage de l'adresse de courrier électronique en clair.

Mission A3 – Évolution de l'application de gestion des contrats pour sécuriser les échanges

La société NDV gère ses contrats de prestation à l'aide d'une application qui exploite une base de données. Actuellement, lorsqu'un contrat est souscrit par une organisation cliente, celle-ci fournit les coordonnées d'un contact référent unique puis l'agence NDV nomme un responsable interne pour ce contrat parmi ses collaborateurs. Ces deux personnes assurent un rôle de chef de projet sur le contrat et tous les échanges entre les deux organisations passent par eux.

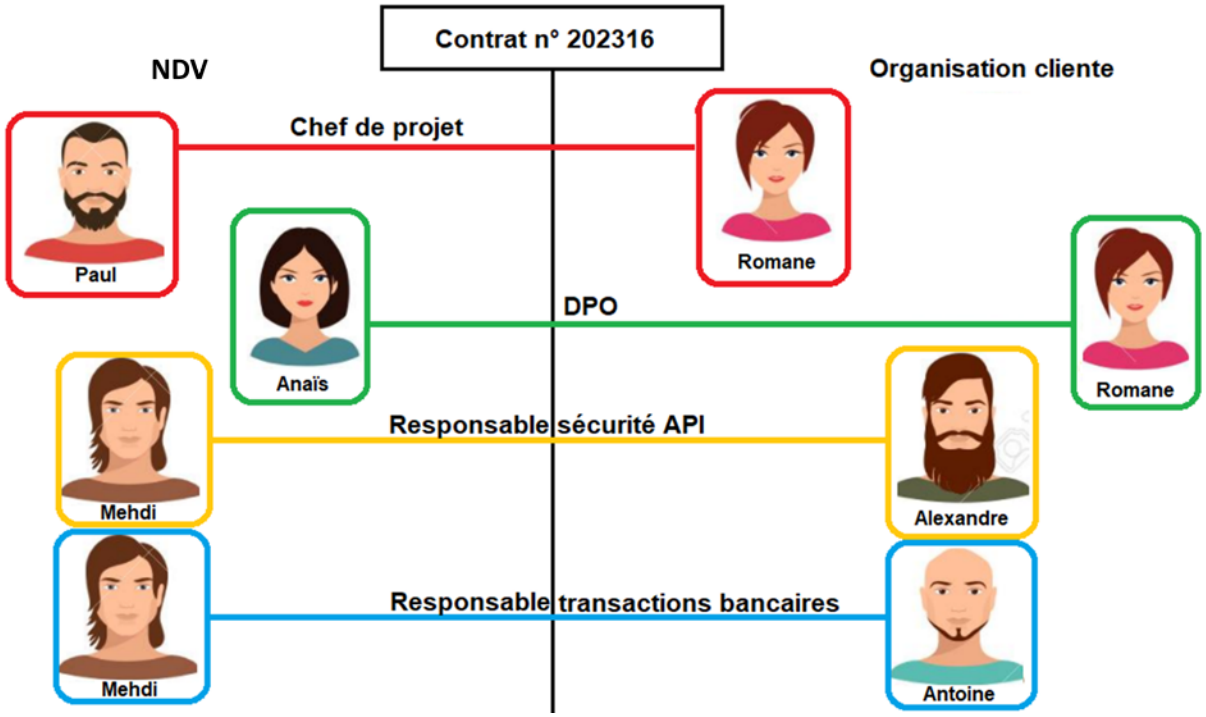
Les évolutions à prendre en compte

L'agence souhaite faire évoluer la gestion des contrats pour fluidifier les échanges entre les membres de l'agence et les interlocuteurs de l'organisation cliente tout en garantissant leur sécurité. Pour cela elle a décidé de mettre en place le concept d'organisation fonctionnelle « miroir » qui consiste, pour un projet, à aligner les équipes du client et celles du prestataire de services pour communiquer de façon rapide et confidentielle sur des aspects spécifiques.

Lors de la création d'un contrat, l'organisation cliente et l'agence devront :

- identifier parmi un ensemble de rôles génériques (chef de projet, développeur, délégué à la protection des données - DPD -, expert, etc.) ceux nécessaires à la réalisation du contrat ;
- nommer pour chaque rôle identifié une personne de l'agence et son homologue chez le client, une même personne pouvant occuper plusieurs rôles.

Voici un exemple de projet de site *web* marchand (contrat n° 202316) où des interlocuteurs doivent pouvoir échanger de façon confidentielle. Le client et le prestataire ont identifié les rôles nécessaires suivants : chef de projet, délégué à la protection des données (DPD ou DPO), responsable de la sécurité des interfaces de programmation applicatives (API) et responsable des transactions bancaires. Ils ont ensuite affecté pour chacun de ces rôles une personne des deux parties :



L'agence NDV doit faire évoluer sa base de données actuelle afin d'intégrer le concept d'organisation fonctionnelle « miroir ».

Question A3.1

Réaliser les modifications nécessaires du schéma de la base ndv afin d'intégrer les nouveaux besoins.

Seuls les éléments du schéma existant concernés par l'évolution seront repris, dans le formalisme de votre choix.

Dossier B – Reprise et sécurisation du site internet du client Architect.60

Le client Architect.60 : Ce cabinet d'architectes s'occupe de projets de construction, de réhabilitation, de rénovation, d'aménagement intérieur et extérieur, de décoration et d'ameublement, pour les particuliers et les professionnels.

Afin d'avoir une visibilité sur Internet, le cabinet Architect.60 dispose d'un site *web* qui présente les différents projets sur lesquels il a travaillé. Ce site permet également aux clients de transmettre aux architectes d'Architect.60 des documents (plans, croquis, etc.), via un espace dédié à chaque client.

Son site *web* étant vieillissant, le cabinet Architect.60 en a confié la maintenance corrective à l'agence NDV qui va le soumettre à son équipe « Audit et Sécurité » avant d'en reprendre la gestion complète.

Mission B1 – Affichage des documents clients sur leur espace dédié

Des clients du cabinet Architect.60 ont indiqué que la liste de leurs documents n'apparaissait plus sur leur espace dédié. En se connectant à la base de données, votre responsable dans l'équipe « Audit et Sécurité » de l'agence NDV a constaté qu'effectivement, la table contenant les documents des clients était vide. Heureusement, il a pu la reconstituer en partant d'une sauvegarde.

Soupçonnant une attaque par injection SQL (A03:2021 - Injection, dans le Top Ten OWASP 2021), il a effectué des tests d'intrusion et a recueilli deux preuves d'exploitation de la vulnérabilité aux injections en testant les charges utiles (*payloads*) suivantes dans le champ `username` du formulaire de connexion à l'espace des clients :

payload1	`sbegard' OR 1=1 #`
payload2	`sbegard' AND 1=0 UNION SELECT null, username, password, null, null FROM T_Users #`

Information : En langage SQL les marques -- et # indiquent que ce qui suit est en commentaire.

Question B1.1

a) Écrire chaque requête exécutée par le serveur de base de données suite à l'injection des charges utiles payload1 et payload2. (Pour alléger votre réponse, ne pas recopier les clauses SELECT et FROM de la requête initiale.)

b) Indiquer les affichages attendus dans le navigateur lors des injections des charges utiles payload1 et payload2.

c) Expliquer pourquoi il n'est pas nécessaire que l'utilisateur *'sbegard'* existe pour que ces deux charges utiles (*payloads*) fonctionnent.

Votre responsable vous charge de retrouver le mode opératoire par lequel les attaquants ont pu vider la table en utilisant la vulnérabilité aux injections.

Question B1.2

a) Proposer une charge utile (*payload)* qui a pu être utilisée par les attaquants pour vider la table des documents.

b) Indiquer quelle faille de sécurité dans le code du formulaire de connexion *form_connexion.php* a pu permettre aux attaquants de connaitre le nom de la table visée par l'attaque.

Votre responsable vous charge enfin d'implémenter les contre-mesures nécessaires dans le code du fichier *espace_client.php* pour empêcher les injections SQL.

Question B1.3

a) Indiquer ce qui rend possible les injections SQL dans ce code.

b) Modifier le code exécuté pour empêcher les injections SQL. (Indiquer seulement le code ajouté et/ou modifié en renseignant les numéros de ligne concernés).

Mission B2 – Modification des documents clients sur leur espace dédié

En inspectant les fichiers liés à la gestion des documents clients, votre responsable a découvert une autre faille de sécurité dans le code permettant la modification des informations des documents (à savoir nom et description du document).

En effet, en changeant l'identifiant du document dans l'adresse *URL* de la page de modification (par exemple : **`…/modif_doc.php?doc_id=2`**), cela ne provoque pas d'erreur et permet la modification du nom et la description du document correspondant dans la base de données.

Cette manipulation fonctionne également avec des documents n'appartenant pas au client connecté car le traitement actuel ne vérifie pas que le document à modifier appartient bien à l'utilisateur connecté. Il vous demande de qualifier cette faille et de proposer une solution pour la corriger.

Votre responsable vous signale que le tableau associatif *$lesDocuments* contient les documents de l'utilisateur connecté.

Question B2.1

a) Identifier, dans le projet OWASP TOP Ten 2021, la faille de sécurité décelée.

b) Modifier le code source du fichier *espace_client.php* pour corriger la faille de sécurité. (Indiquer seulement le code ajouté et/ou modifié en renseignant les numéros de ligne concernés).

Mission B3 – Création des comptes clients pour l'accès à leur espace dédié

À la suite de cette suspicion de cyberattaque, certains clients ont signalé avoir perdu l'accès à leur espace dédié malgré la saisie correcte de leur mot de passe. Cette situation résulte d'une usurpation d'identité réussie : l'attaquant, après avoir exploité des faiblesses dans le mécanisme de hachage des mots de passe, a pu modifier les identifiants de connexion, rendant les accès légitimes inopérants.

Votre responsable vous demande donc d'analyser le code source des fichiers permettant aux clients de créer leur compte dédié sur le site d'Architect.60, dans lesquels la création du mot de passe intervient.

Question B3.1

a) Identifier l'algorithme de hachage utilisé dans le processus de création des comptes clients et indiquer pourquoi il n'est pas efficace.

b) Identifier, dans le projet OWASP TOP Ten 2021, la faille de sécurité décelée liée à cet algorithme de hachage.

Pour renforcer la sécurité, votre responsable vous demande de remplacer l'algorithme de hachage actuellement utilisé, par l'algorithme Argon2i qui est plus robuste.

Cette évolution implique de faire évoluer le code des programmes *traitement_inscription.php* et *espace_client.php*.

Question B3.2

a) Modifier le code source lié à la création des comptes clients (fichier *traitement_inscription.php*) pour implémenter cette solution. (Indiquer seulement les lignes ajoutées et/ou modifiées.)

b) Modifier le code source lié à l'affichage des documents du client dans le fichier *espace_client.php* pour prendre en compte ces changements. (Indiquer seulement les lignes ajoutées et/ou modifiées.)

Documents associés au dossier A

Document A1a : Affichage actuel des journaux d'activité sur le site

Affichage des logs

Type de recherche: Date: 19 / 11 / 2024 Rechercher

1 2024-11-19 16:00:33 **[CONNEXION]** [ndv_64784]@[localhost] **ndv_64784@localhost on ndv using TCP/IP**
2 2024-11-19 16:01:33 **[REQUETE]** [ndv_64784]@[localhost] **SELECT * FROM Contrat WHERE id=202316**
3 2024-11-19 16:04:33 **[REQUETE]** [ndv_64784]@[localhost] **UPDATE Contrat SET idResponsable='00246' WHERE contrat_id=202316**
4 2024-11-19 16:15:33 **[REQUETE]** [ndv_64784]@[localhost] **SELECT * FROM Collaborateur WHERE matricule='00246'**
5 2024-11-19 16:17:33 **[DECONNEXION]** [ndv_64784]@[localhost]
6 2024-11-19 16:28:07 **[ERREUR]** [admin]@[172.17.12.45] **Access denied for user 'admin'@'172.17.12.45' (using password: YES)**
7 2024-11-19 16:30:11 **[ERREUR]** [admin]@[172.17.12.45] **Access denied for user 'admin'@'172.17.12.45' (using password: YES)**
8 2024-11-19 18:10:36 **[CONNEXION]** [ndv_64784]@[172.17.12.45] **ndv_64784@172.17.12.45 on ndv using SSL/TLS**
9 2024-11-19 18:11:14 **[REQUETE]** [ndv_64784]@[172.17.12.45] **CREATE USER 'test'@'%' IDENTIFIED WITH 'mysql_native_password' AS ''**
10 2024-11-19 18:11:27 **[REQUETE]** [ndv_64784]@[172.17.12.45] **GRANT ALL ON *.* TO 'test'@'%'**
11 2024-11-19 18:11:34 **[DECONNEXION]** [ndv_64784]@[172.17.12.45]
12 2024-11-19 18:16:36 **[CONNEXION]** [test]@[172.17.12.45] **test@172.17.12.45 on ndv using SSL/TLS**
13 2024-11-19 18:17:14 **[REQUETE]** [test]@[172.17.12.45] **SHOW TABLES**
14 2024-11-19 18:19:27 **[REQUETE]** [test]@[172.17.12.45] **DROP TABLE Contrat**
15 2024-11-19 18:21:34 **[DECONNEXION]** [test]@[172.17.12.45]

Document A1b : Code HTML du nouvel écran souhaité

```
1.    <h2>Affichage des logs</h2>
2.    <form method="post" action="select_logs.php">
3.          <label for="type_de_logs">Nature de logs:</label>
4.          <select name="type_de_logs" id="type_de_logs">
5.                <option value="">TOUS</option>
6.                <option value="CONNEXION">CONNEXION</option>
7.                <option value="ERREUR">ERREUR</option>
8.                <option value="DECONNEXION">DECONNEXION</option>
9.                <option value="REQUETE">REQUETE</option>
10.         </select>
11.         <label for="date_choisie">Date:</label>
12.         <input type="date" name="date_choisie" id="date_choisie" value="<?php echo
      $_POST['date_choisie'] ?? '' ?>" /> <!-- Affiche la date choisie si elle est
      transmise ou une chaine vide sinon -->
13.         <button type="submit">Rechercher</button>
14.   </form>
15.   <br/>
```

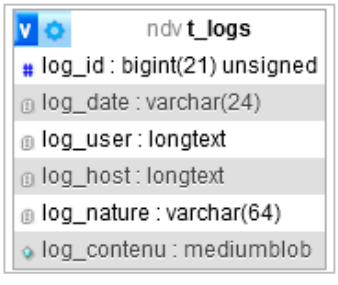
Document A2 : Extrait du fichier select_logs.php avec appel de l'affichage

```
// retourne une instance PDO permettant de se connecter à la base de données ndv
1.   function db_connexion() : PDO {
2.       $dsn = 'mysql:dbname=ndv;host=localhost';
3.       $user = 'ndv_64784';
4.       $password = 'xgHF45!3rF@th';
5.       return new PDO($dsn, $user, $password);
6.   }

7.   // renvoie l'ensemble des logs formatés pour l'affichage
8.   // ** fonction à faire évoluer pour répondre au besoin **
9.   function print_logs() : array
10.  {
11.      $logs = array(); // Tableau qui contiendra tous les logs
12.      // La fonction isset($var) vérifie si $var est déclarée et différente de null
13.      $date = null;
14.      if (isset($_POST['date_choisie']) AND $_POST['date_choisie'] != ''){
15.          $date = $_POST['date_choisie'];
16.      }
17.      $query = "SELECT * FROM t_logs WHERE (:date IS NULL OR DATE(log_date) = :date)
             ORDER BY log_date ASC";
18.      $conn = db_connexion();
19.      $stmt = $conn->prepare($query);
20.      $stmt->bindParam(':date', $date, PDO::PARAM_STR);
21.      $stmt->execute();
22.      $result = $stmt->fetchAll(PDO::FETCH_ASSOC);
23.      foreach ($result as $row) {
24.      // syntaxe php : "Bonjour {$nom}" inclut la variable $nom dans la chaine.
25.          $logs[] = "{$row['log_date']} <b>[{$row['log_nature']}]</b>
                      [{$row['log_user']}]@[{$row['log_host']}]
                      <b>{$row['log_contenu']}</b>";
26.      }
27.      return $logs;
28.  }

29.  // Affichage des logs
30.  $logs = print_logs();
31.  foreach ($logs as $log) { echo $log."<br>"; }
```

Document A3 : Structure de la vue t_logs, alimentée par le système de journalisation du SGBD MySQL et extrait des habilitations actuelles sur la base de l'intranet ndv



Utilisateurs ayant accès à « ndv »

	Nom d'utilisateur	Nom d'hôte	Type	Privilèges	« Grant »
☐	ndv_64784	%	spécifique à cette base de données	ALL PRIVILEGES	Oui
☐	root	localhost	global	ALL PRIVILEGES	Oui
☐	test	%	spécifique à cette base de données	ALL PRIVILEGES	Non

Document A4 : Documentation technique PDO, MySQL et SQL

Utilisation de PDO en PHP

```
1.  <?php
2.  /* Connexion à une base de données MySQL testdb située sur 127.0.0.1 */
3.  $dsn = 'mysql:dbname=testdb;host=127.0.0.1';
4.  $user = 'usertest';
5.  $password = 'passtest';
6.  $bdd = new PDO($dsn, $user, $password);
7.  /* Exécute une requête préparée multiple d'insertion et de suppression */
8.  $sql = "INSERT INTO histo_user VALUES (:id, :nom, NOW()); DELETE FROM user WHERE
    id=:id";
9.  $reqPrep = $bdd->prepare($sql); // $bdd étant la base déclarée plus haut
10. /* Valorisation des éléments composants la requête */
11. $reqPrep->bindParam(':id', 12, PDO::PARAM_INT);
12. $reqPrep->bindParam(':nom', 'bob', PDO::PARAM_STR);
13. $reqPrep->execute();
14. /* Exécute une requête préparée de lecture */
15. $sql = 'SELECT * FROM histo_user WHERE dateHisto < :date ';
16. $reqPrep = $bdd->prepare($sql);
17. $reqPrep->bindParam(':date', '2023-11-19', PDO::PARAM_STR);
18. $reqPrep->execute();
19. /* Récupération des données de la requête dans la variable $reqFetch */
20. $reqFetch = $reqPrep->fetchAll(PDO::FETCH_ASSOC);
21. ?>
```

Gestion des droits utilisateurs en MySQL

Créer un utilisateur *test* se connectant depuis le serveur local	`CREATE USER 'test'@'localhost' IDENTIFIED BY 'password';`
Modifier l'adresse IP de connexion autorisée pour l'utilisateur *test* de 172*.17.46.4* à *10.8.45.234*	`RENAME USER 'test'@'172.17.46.4' TO 'test'@'10.8.45.234';`
Supprimer un utilisateur *test* se connectant depuis l'adresse IP *10.8.45.234*	`DROP USER 'test'@'10.8.45.234'`
Changer le mot de passe d'un utilisateur *test* se connectant en local	`SET PASSWORD FOR 'test'@'localhost' = PASSWORD('newpassword');`
Attribuer les droits d'insertion et de suppression sur la table *fruit* de la base *database* à un utilisateur *test* se connectant depuis n'importe où	`GRANT INSERT, DELETE ON database.fruit TO 'test'@'%';`
Retirer tous les droits sur toutes les tables de la base database à l'utilisateur *test* se connectant en local	`REVOKE ALL ON database.* FROM 'test'@'localhost';`

…/…

Documentation sur le format VARBINARY dans MySQL

Le type de données VARBINARY dans *MySQL* est utilisé pour stocker des données binaires variables. Il est similaire au type VARCHAR, mais il traite les données comme des octets plutôt que des caractères. Une donnée en VARCHAR peut être convertie sans perte en VARBINARY.

Syntaxe de déclaration : `VARBINARY(M)`

`M` représente la longueur maximale des données binaires.

Fonctions de chiffrement/déchiffrement AES en MySQL

AES_ENCRYPT et **AES_DECRYPT** sont des fonctions qui permettent de chiffrer et de déchiffrer des données à l'aide de l'algorithme AES (*Advanced Encryption Standard*).

AES_ENCRYPT permet de chiffrer une chaîne de caractères avec l'algorithme AES et la renvoie sous la forme d'une chaîne binaire :

`AES_ENCRYPT(`*`str, key_str`*`)`

str : La chaîne de caractères à chiffrer.

key_str : La clé de chiffrement utilisée pour chiffrer et déchiffrer les données.

AES_DECRYPT permet de déchiffrer une chaîne de caractères chiffrée avec l'algorithme AES :

`AES_DECRYPT(`*`crypt_str, key_str`*`)`

crypt_str : La chaîne de caractères chiffrée au format binaire à déchiffrer.

key_str : La même clé de chiffrement utilisée lors du chiffrement.

Syntaxe de la commande SQL ALTER TABLE

La commande ALTER TABLE en langage SQL permet de modifier une table existante pour ajouter une colonne, supprimer une colonne ou modifier une colonne existante, par exemple pour changer le type.

Syntaxe de base :

```
ALTER TABLE nom_table instruction
```

Le mot-clé « instruction » ici sert à désigner une commande supplémentaire, qui sera détaillée ci-dessous :

Ajouter une colonne	`ALTER TABLE nom_table` `ADD nom_colonne type_donnees`
Supprimer une colonne	`ALTER TABLE nom_table` `DROP COLUMN nom_colonne`
Modifier une colonne	`ALTER TABLE nom_table` `MODIFY nom_colonne type_donnees`
Renommer une colonne	`ALTER TABLE nom_table` `CHANGE colonne_ancien_nom colonne_nouveau_nom type_donnees`

Document A5 : Extrait du RGPD – Section 2 - Sécurité des données à caractère personnel

Article 32 - Sécurité du traitement

1. Compte tenu de l'état des connaissances, des coûts de mise en œuvre et de la nature, de la portée, du contexte et des finalités du traitement ainsi que des risques, dont le degré de probabilité et de gravité varie, pour les droits et libertés des personnes physiques, le responsable du traitement et le sous-traitant mettent en œuvre les mesures techniques et organisationnelles appropriées afin de garantir un niveau de sécurité adapté au risque, y compris entre autres, selon les besoins :

    a. la pseudonymisation et le chiffrement des données à caractère personnel ;

    b. des moyens permettant de garantir la confidentialité, l'intégrité, la disponibilité et la résilience constantes des systèmes et des services de traitement ;

    c. des moyens permettant de rétablir la disponibilité des données à caractère personnel et l'accès à celles-ci dans des délais appropriés en cas d'incident physique ou technique ;

    d. une procédure visant à tester, à analyser et à évaluer régulièrement l'efficacité des mesures techniques et organisationnelles pour assurer la sécurité du traitement.

2. Lors de l'évaluation du niveau de sécurité approprié, il est tenu compte en particulier des risques que présente le traitement, résultant notamment de la destruction, de la perte, de l'altération, de la divulgation non autorisée de données à caractère personnel transmises, conservées ou traitées d'une autre manière, ou de l'accès non autorisé à de telles données, de manière accidentelle ou illicite.

3. L'application d'un code de conduite approuvé comme le prévoit l'article 40 ou d'un mécanisme de certification approuvé comme le prévoit l'article 42 peut servir d'élément pour démontrer le respect des exigences prévues au paragraphe 1 du présent article.

4. Le responsable du traitement et le sous-traitant prennent des mesures afin de garantir que toute personne physique agissant sous l'autorité du responsable du traitement ou sous celle du sous-traitant, qui a accès à des données à caractère personnel, ne les traite pas, excepté sur instruction du responsable du traitement, à moins d'y être obligée par le droit de l'Union ou le droit d'un État membre.

Document A6 : Code de la fonction getContact qui retourne les données d'un contact

```
1.   public function getContact(int $idContrat): array
2.   {
3.       $contact = array();
4.       $sql = "SELECT ce.nom, ce.prenom, ce.adresseMel, ce.telephone
.                   FROM Contrat c
.                   JOIN ContactExterne ce ON c.idReferent = ce.id
.                   WHERE c.id = :id";
5.       $stmt = $this->db->prepare($sql);
6.       $stmt->bindValue(':id', $idContrat, PDO::PARAM_INT);
7.       $stmt->execute();
8.       $contact = $stmt->fetch(PDO::FETCH_ASSOC);
9.       return $contact;
10.  }
```

Document A7 : Extrait de la base de données ndv

Schéma relationnel

OrganisationCliente(id, siret , denomination, adresseMel, nomPrenomDirigeant, adresseSiegeSocial, codePostal, telephone, ville)
Clé primaire : id

ContactExterne (id, nom, prenom, adresseMel, telephone, idOrganisation)
Clé primaire : id
Clé étrangère : idOrganisation en référence à id de OrganisationCliente

Collaborateur (matricule, nom, prenom, adresseMel, telephone)
Clé primaire : matricule

Contrat (id, reference, label, dateCreation, idClient, idReferent, idResponsable)
Clé primaire : id
Clés étrangères :
idClient en référence à id de OrganisationCliente
idReferent en référence à id de ContactExterne
idResponsable en référence à matricule de Collaborateur

Contrainte : Il faut vérifier que le référent associé à un contrat travaille bien pour l'organisation cliente associée à ce contrat.

Diagramme de classes

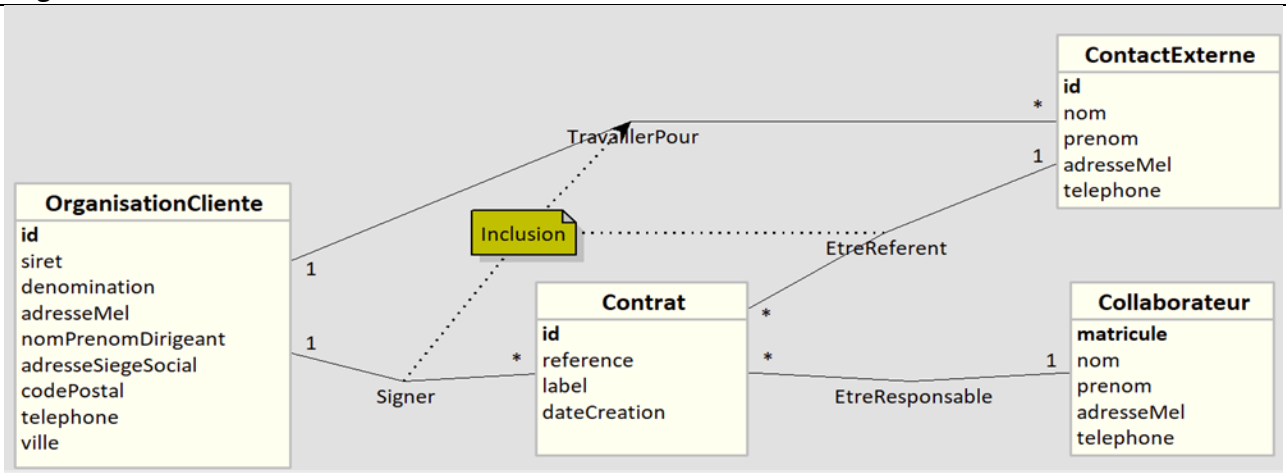
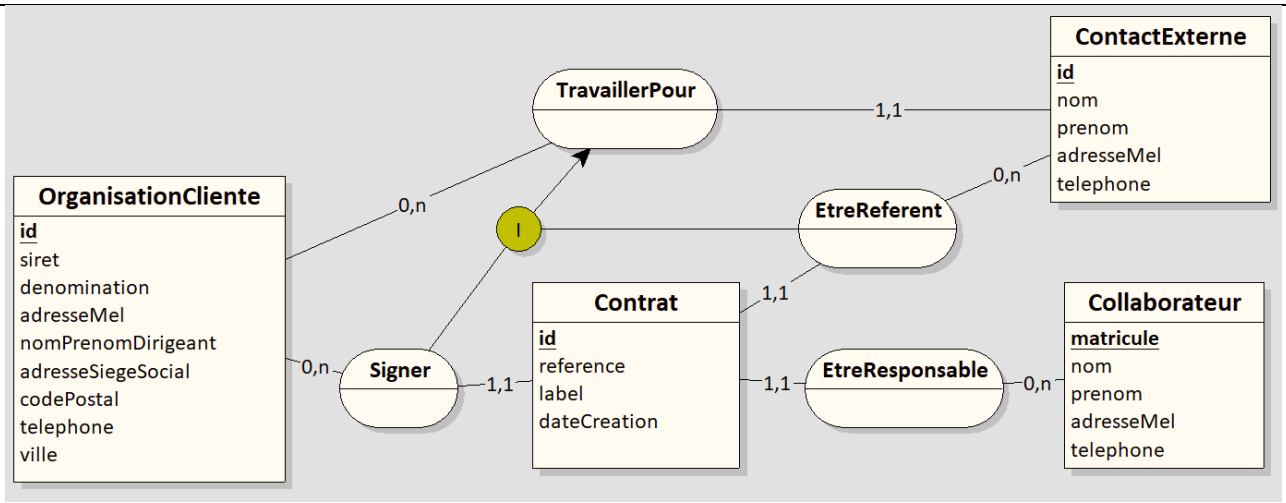


Schéma entité-association



Documents associés au dossier B

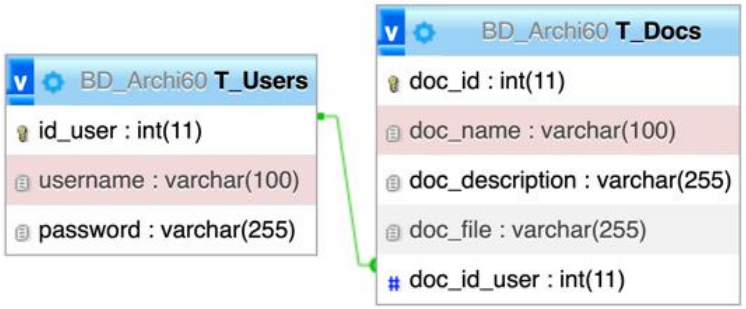
Document B1 : Présentation de la fondation OWASP

L'organisation Open Worldwide Application Security Project® (OWASP) est une fondation à but non lucratif qui travaille à améliorer la sécurité des logiciels. Grâce à des projets de logiciels *open source* réalisés par la communauté, des centaines de chapitres locaux dans le monde, des dizaines de milliers de membres et des conférences éducatives et de formation de premier plan, la fondation OWASP est la source pour les développeurs pour sécuriser le Web.

Document B2 : Extrait du projet OWASP Top Ten 2021

Rang	Nom de la faille de sécurité	Description de la faille de sécurité
A01:2021	Contrôles d'accès défaillants	L'utilisateur peut effectuer des actions sur une donnée alors qu'il ne possède pas les droits requis (privilèges).
A02:2021	Défaillances cryptographiques	Utilisation d'un algorithme de chiffrement obsolète, exposition de données sensibles.
A03:2021	Injection	Utilisation de données récupérées par un formulaire pour exécuter du code SQL à l'insu du créateur.
A04:2021	Conception non sécurisée	La conception de l'application présente des défauts au niveau métier.
A05:2021	Mauvaise configuration de sécurité	La configuration de l'application est mal/pas effectuée.
A06:2021	Composants vulnérables et obsolètes	Les dépendances de composants ne sont pas à jour, voire obsolètes.
A07:2021	Identification et authentification de mauvaise qualité	Faiblesse, problème au niveau de l'authentification.
A08:2021	Manque d'intégrité des données et du logiciel	Application utilisant des dépendances non fiables.
A09:2021	Carence des systèmes de contrôle et de journalisation	Aucune journalisation mise en place (ou avec des données peu exploitables)
A10:2021	Falsification de requête coté serveur (SSRF)	Le serveur *web* récupère une ressource distante sans valider l'URL fournie par l'utilisateur.

Document B3 : Extraits de la base de données BD_Archi60



id_user	username	password
1	MonsieurT	2ff5d5a240ba80d07ce08c9
2	LambertM	aaf4c61ddcc5e8a2dabede(

doc_id	doc_name	doc_description	doc_file	doc_id_user
1	Croquis_rdc	Croquis qui présente la décoration du rdc	rdc.jpg	1
2	Croquis_etage	Croquis qui présente la décoration de l'étage	etage.jpg	1
3	Plan_maison	Plan total de la maison	maison.jpg	2

Document B4 : Extraits du code source lié à l'espace dédié des clients du site

Extrait du formulaire d'inscription : form_inscription.php

```
1.   <!-- Saisie des informations de profil -->
2.   <h1>Saisir vos informations de profil</h1>
3.   <form method="post" name="form" action="traitement_inscription.php">
4.       <!-- Zone de saisie de l'identifiant -->
5.       <input type="text" name="username" placeholder="Identifiant">
6.       <!-- Zone de saisie du mot de passe -->
7.       <input type="password" name="password" placeholder="Mot de passe">
8.       <!-- Bouton de validation du formulaire -->
9.       <input type="submit" name="signin" value="S'inscrire">
10.  </form>
```

Extrait du code source contenant l'inscription des clients : traitement_inscription.php

```
1.    // Inclusion du script contenant la fonction de connexion à la base de données
2.    require "connexionBdd.php";

3.    // Appel de la fonction de connexion à la base de données
4.    $bdd = connexionBdd();

5.    if (isset($_POST['signin'])) {
6.        // On récupère les données saisies dans le formulaire d'inscription
7.        $newusername = $_POST['username'];
8.        $newpass = $_POST['password'];

9.        // On hache le mot de passe avec l'algorithme SHA-1
10.       $newpasshash = SHA1($newpass);

11.       // On vérifie ici que l'utilisateur n'est pas déjà présent dans la table
12.       // le cas échéant, on arrête le traitement
13.       // ...

14.       // Préparation de la requete pour insérer un nouvel utilisateur dans la bdd
15.       $sql = "INSERT INTO T_Users (username, password) VALUES (:username,
             :password)";
16.       $insertAccount = $bdd->prepare($sql);
17.       $insertAccount->bindParam(':username', $newusername, PDO::PARAM_STR);
18.       $insertAccount->bindParam(':password', $newpasshash, PDO::PARAM_STR);
19.       $insertAccount->execute();
20.       header("Location: form_connexion.php");
21.   }
```

Extrait du formulaire de connexion à l'espace client : form_connexion.php

```
1.   <!-- Accès à l'espace client et aux documents de la table T_Docs -->
2.   <h1>Authentification</h1>
3.   <form method="post" name="form" action="espace_client.php">
4.        <!-- Zone de saisie de l'identifiant -->
5.        <input type="text" name="username" placeholder="Identifiant">
6.        <!-- Zone de saisie du mot de passe -->
7.        <input type="password" name="password" placeholder="Mot de passe">
8.        <!-- Bouton de validation du formulaire -->
9.        <input type="submit" name="login" value="Se connecter">
10.  </form>
```

Suite du code page suivante

Extrait de code de la page affichant l'espace dédié d'un client : espace_client.php

```
1.  <?php
2.  /* Inclusion du script contenant la fonction de connexion à la base de données, appel
    de la fonction de connexion à la base de données et démarrage de la session */
3.  require "connexionBdd.php";
4.  $bdd = connexionBdd();
5.  session_start();

6.  if (isset($_POST['login'])) {
7.      $_SESSION['password'] = SHA1($_POST['password']);
8.      $_SESSION['username'] = $_POST['username'];
9.  }

10. $password = $_SESSION['password'];
11. $username = $_SESSION['username'];

12. $sql = "SELECT id_user, doc_name, doc_description, doc_id, doc_file
13.         FROM T_Users LEFT JOIN T_Docs ON T_Users.id_user = T_Docs.doc_id_user
14.         WHERE username ='$username'
15.         AND password ='$password'";
16. $select = $bdd->prepare($sql);
17. $select->execute();
18. $lesDocuments = $select->fetchAll(PDO::FETCH_ASSOC);

19. if ((count($lesDocuments) > 0) && ($lesDocuments[0][doc_id] != null) {
20. ?>
21.     <!-- Affichage du username -->
22.     <h1>Bienvenue sur votre espace <?php echo $username ?> !</h1>
23.     <!-- Affichage des documents liés -->
24.     <p>Voici la liste des documents que vous avez transmis :</p>
25.     <table>
26.         <thead>
27.             <tr>
28.                 <th>Nom doc.</th>
29.                 <th>Description doc.</th>
30.                 <th>Fichier</th>
31.                 <th>Modification du document</th>
32.             </tr>
33.         </thead>
34.         <tbody>
35.         <?php foreach ($lesDocuments as $element) { ?>
36.             <tr>
37.                 <td><?php echo $element['doc_name'] ?></td>
38.                 <td><?php echo $element['doc_description'] ?></td>
39.                 <td><?php echo $element['doc_file'] ?></td>
40.                 <!-- Lien vers la page de modification du document -->
41.                 <td><a href="modif_doc.php?doc_id=
                            <?php echo $element['doc_id'] ?>">Modifier</a></td>
42.             </tr>
43.         <?php } ?>
44.         </tbody>
45.     </table>
46.     <?php
47. } else {
48.     echo '<script>alert("Identifiant et/ou mot de passe invalide !");
49.     window.location="form_connexion.php"</script>';
50. }
```

Suite du code page suivante

```
51.  // Traitement de la modification d'un document
52.  if (isset($_POST['update'])) {
53.      $docid = $_POST['docid'];
54.      $docname = $_POST['docname'];
55.      $docdescription = $_POST['docdescription'];
56.      $sql = "UPDATE T_Docs SET doc_name = '$docname',
                   doc_description = '$docdescription' WHERE doc_id = '$docid'";
57.      $updateDoc = $bdd->query($sql);
58.      header("Location: espace_client.php");
59.  }
60.  ?>
```

Extrait du formulaire de modification des informations d'un document : modif_doc.php

```
1.   <!-- Formulaire de modification des informations d'un document -->
2.   <h1> Modification du document</h1>
3.   <form method="post" name="form" action="espace_client.php">
4.      <!-- Champs identifiant le document à modifier -->
5.      <input type="hidden" name="docid" value=<?php echo $_GET['doc_id'] ?>>
6.      <!-- Zone de modification du nom du document -->
7.      <input type="text" name="docname" placeholder="Nom du document">
8.      <!-- Zone de modification de la description du document -->
9.      <input type="text" name="docdescription" placeholder="Description du document">
10.     <!-- Bouton de validation de la modification -->
11.     <input type="submit" name="update" value="Effectuer la modification">
12.  </form>
```

Document B5 : Liste non exhaustive des algorithmes de hachage

Algorithme	Date de sortie	Statut actuel	Utilisation $clair : chaine à hacher ou à vérifier $hash : hash récupéré en BDD
MD5	1991	Sévèrement compromis, largement déconseillé pour le stockage de mots de passe.	`md5($clair) : str`
SHA-1	1993	Vulnérable aux attaques par collision, déconseillé pour le stockage sécurisé des mots de passe.	`sha1($clair) :str`
SHA-256	2001	Robuste et largement utilisé pour le stockage sécurisé des mots de passe.	`hash('sha256', $clair) : str`
Bcrypt	1999	Conçu spécifiquement pour le stockage de mots de passe, résistant aux attaques par force brute, recommandé.	`password_hash($clair, PASSWORD_BCRYPT) : str` `password_verify($clair, $hash) : bool`
Argon2i	2015	Conçu pour être résistant aux attaques par force brute et parallèles, recommandé pour le stockage sécurisé des mots de passe.	`password_hash($clair, PASSWORD_ARGON2I) : str` `password_verify($clair, $hash) : bool`

Il est fortement recommandé d'utiliser des algorithmes de hachage spécifiquement conçus pour le stockage de mots de passe qui intègrent des mécanismes de coût et de salage pour renforcer la sécurité.